TESIS - IF - 185401

PEMODELAN KLASIFIKASI NON-KEBUTUHAN DENGAN FITUR SEMANTIK PERNYATAAN KEBUTUHAN PERANGKAT LUNAK

Achmad An'im Fahmi
NRP. 05111650010060

Dosen Pembimbing
Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.

Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
2020

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN TESIS

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar
Magister Komputer (M.Kom)
di
Institut Teknologi Sepuluh Nopember
Oleh:
ACHMAD AN'IM FAHMI
NRP. 05111650010060

Tanggal Ujian: 10 Juli 2020
Periode Wisuda: September 2020

Disetujui oleh:
Pembimbing:

1. Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng. .....
   NIP. 19741123 200604 1 001

Penguji:

1. Dr. Ir. Siti Rochimah, M.T. .....
   NIP. 196810021994032001

2. Dr. Umi Laili Yuhana S.Kom., M.Sc. .....
   NIP. 197906262005012002

3. Dr. Agus Budi Raharjo, S.Kom., M.Kom. .....
   NIP. 1990202011022

Kepala Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas

Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.
NIP. 197512202001122002

(Halaman ini sengaja dikosongkan)

PEMODELAN KLASIFIKASI NON-KEBUTUHAN DENGAN FITUR SEMANTIK PERNYATAAN KEBUTUHAN PERANGKAT LUNAK

Nama	:	Achmad An'im Fahmi
NRP	:	05111650010060
Pembimbing	:	Daniel Oranova Siahaan, S.Kom, M.Sc, PD.Eng.

ABSTRAK

Noise atau derau dalam Spesifikasi Kebutuhan Perangkat Lunak (SKPL) adalah pernyataan kebutuhan tidak relevan atau pernyataan non-kebutuhan. Hal tersebut dapat membingungkan pembaca dan dapat berakibat buruk pada tahap-tahap pengembangan perangkat lunak selanjutnya. Deteksi derau pada penelitian sebelumnya berfokus pada jenis derau yang diakibatkan oleh pernyataan kebutuhan yang tidak relevan. Metode yang dikembangkan gagal mendeteksi jenis derau yang kedua, yaitu pernyataan non-kebutuhan. Hasil dari penelitian sebelumnya belum dapat dengan baik mendeteksi derau yang muncul dalam sebuah dokumen spesifikasi kebutuhan perangkat lunak.

Penelitian ini mengajukan sebuah model klasifikasi untuk mendeteksi jenis derau yang kedua, yaitu pernyataan non-kebutuhan. Model klasfikasi yang dibangun didasarkan kepada fitur semantic dari pernyataan non-kebutuhan. Penelitian ini juga membanding lima metode pembelajaran mesin tersupervisi terbaik saat ini, yaitu *support vector machine* (*SVM*), *naïve bayes* (*NB*), *random forest* (*RF*), *k-nearest neighbor* (*kNN*), dan *Decision Tree*. Perbandingan ini bertujuan untuk mengetahui metode mana yang dapat menghasilkan model klasifikasi non-kebutuhan terbaik.

Pengujian atas kelima metode tersebut menggunakan 14 dataset spesifikasi kebutuhan perangkat lunak. Hasil dari perbandingan menunjukkan model terbaik dihasilkan oleh metode *SVM* dengan rata-rata akurasi 0,96. Adapun fitur yang paling signifikan dalam model klasifikasi non-kebutuhan ini adalah pernyataan kebutuhan atau non-kebutuhan, id pernyataan, nilai normalisasi mean, nilai standar deviasi, nilai varian kesamaan, nilai normalisasi standar deviasi, nilai normalisasi maksimal, nilai normalisasi varian kesamaan, nilai NN yang buruk, nilai mean, jumlah kalimat, nilai VB yang buruk, dan id projek.

Kata kunci: *Derau, Klasifikasi SVM, Pernyataan Kebutuhan Tidak Relevan, Pernyataan Non-Kebutuhan, Spectral Clustering, Spesifikasi Kebutuhan*

(Halaman ini sengaja dikosongkan)

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Pada bab ini akan dijelaskan mengenai beberapa hal dasar dalam pembuatan tesis penelitian yang meliputi latar belakang, perumusan masalah, tujuan, manfaat, kontribusi penelitian, dan batasan masalah.

1.1. Latar Belakang

Penjaminan Kualitas Perangkat Lunak (PKPL) merupakan sebuah proses yang memantau seluruh proses pengembangan perangkat lunak untuk memastikan bahwa proses-proses tersebut dapat menghasilkan perangkat lunak yang berkualitas. PKPL mencakup seluruh tahap pada Daur Hidup Perangkat Lunak (DHPL). Proses PKPL, khususnya pada tahap spesifikasi kebutuhan, dinilai sangat penting menurut Boehm dan Basili (Meyer, 1985), untuk mencegah peningkatan biaya yang diperlukan saat pembangunan perangkat lunak. Perangkat lunak yang telah selesai diperbaiki memerlukan biaya lebih besar daripada pada saat tahap spesifikasi kebutuhan dan desain perangkat lunak.

Noise atau derau adalah salah satu dari jenis kesalahan yang sering dilakukan oleh perekayasa kebutuhan (Meyer, 1985). Derau dapat muncul pada proses spesifikasi kebutuhan di dalam Spesifikasi Kebutuhan Perangkat Lunak (SKPL). Derau terjadi jika dalam spesifikasi kebutuhan, pengembang perangkat lunak menyertakan suatu informasi yang tidak relevan dengan keseluruhan kebutuhan perangkat lunak atau menyertakan suatu pernyataan *non-requirement* atau non-kebutuhan. Hal tersebut dapat membingungkan pembaca dan dapat berakibat buruk pada tahap-tahap pengembangan perangkat lunak selanjutnya.

Derau dibagi menjadi dua tipe, yaitu kebutuhan yang tidak relevan dan pernyataan non-kebutuhan. Kebutuhan yang tidak relevan lebih membahas topik di luar topik umum yang sedang dibahas oleh seluruh kebutuhan yang relevan. Topik-topik tersebut secara semantik jauh dari topik-topik umum. Selain itu, topik umum dari SKPL akan memiliki jarak semantik lebih jauh dari topik umum spesifikasi perangkat lunak lainnya. Pernyataan non-kebutuhan cenderung berisi topik yang juga terdapat pada pernyataan kebutuhan dari SKPL yang sama. Meskipun derau

dalam berbagai SKPL memiliki berbagai bagian urutan ucapan, mereka cenderung memiliki istilah tertentu (seperti proyek, jadwal, berorientasi objek, dan penerimaan pengguna).

Deteksi derau pada penelitian sebelumnya menghasilkan nilai Koefisien Kappa yang kurang memuaskan, yaitu 0,4426. Hal tersebut disebabkan karena pada penelitian tersebut metode yang dikembangkan hanya mendeteksi derau tipe pertama, yaitu pernyataan kebutuhan yang tidak relevan (Manek and Siahaan, 2019). Penulis akan memperbaiki metode sebelumnya dengan mengembangkan model klasifikasi untuk mendeteksi derau tipe kedua, yaitu pernyataan non-kebutuhan.

(Liu *et al.*, 2010) membandingkan tiga metode pembelajaran mesin tersupervisi pada data lingkungan, olahraga, politik, dan seni. Keempat metode tersebut dipilih dengan alasan metode tersebut adalah teknik yang paling umum untuk klasifikasi teks. Dari hasil pengujian terhadap empat dataset, terlihat bahwa tiga metode teratas dalam klasifikasi adalah *kNN*, *Naive Bayes,* dan *SVM*. Dilihat dari nilai F1, metode klasifikasi *SVM* memperoleh nilai tertinggi pada semua data dibandingkan dua metode klasifikasi lainnya.

Penelitian (Colditz, 2015) membandingkan berbagai strategi alokasi data pelatihan untuk pemetaan tutupan lahan yang terpisah dan berkesinambungan menggunakan algoritma klasifikasi dan pohon regresi. Metode klasifikasi yang digunakan adalah *Decision Tree* dan *Random Forest.* Dataset yang digunakan adalah *The National Land Cover Data set (NLCD)*. Dari algoritma yang diuji, metode *Random Forest* menghasilkan kesalahan yang lebih sedikit dibandingkan *Decision Tree.*

Model klasifikasi yang akan dikembangkan untuk mendeteksi informasi yang non-kebutuhan akan dibangun berdasarkan fitur semantik dari pernyataan kebutuhan. Penelitian ini juga akan membandingkan lima metode klasifikasi berbasis fitur yang terbaik saat ini, yaitu *support vector machine* (*SVM*), *naïve bayes* (*NB*), *random forest* (*RF*), *k-nearest neighbor* (*kNN*), dan *Decision Tree* (*DT*). Perbandingan ini bertujuan untuk menemukan metode pembelajaran mesin tersupervisi yang dapat menghasilkan model klasifikasi paling akurat dalam mendeteksi pernyataan non-kebutuhan.

1.2. Perumusan Masalah

Rumusan masalah yang diangkat dalam penelitian ini adalah sebagai berikut.

1. Fitur semantic apa dalam kalimat pernyataan yang dapat mengklasifikasi sebuah pernyataan sebagai pernyataan kebutuhan atau pernyataan non-kebutuhan?
2. Metode supervisi klasifikasi apa yang dapat menghasilkan model klasifikasi terbaik?

1.3. Tujuan

Tujuan yang akan dicapai dalam penelitian ini adalah memperbaiki akurasi dari metode pendeteksian derau pada SKPL. Perbaikan metode dilakukan dengan mengembangkan model klasifikasi pernyataan non-kebutuhan pada SKPL menggunakan metode pembelajaran mesin tersupervisi.

1.4. Manfaat

Manfaat dari penelitian ini adalah menyediakan suatu metode mendeteksi derau bagi pengembang perangkat lunak yang dapat mengklasifikasi pernyataan non-kebutuhan berdasarkan fitur semantik yang terdapat dalam SKPL. Hasil dari penelitan ini diharapkan dapat meningkatkan kualitas SKPL.

1.5. Kontribusi Penelitian

Kontribusi penelitian ini adalah membangun model klasifikasi pernyataan non-kebutuhan untuk mendeteksi derau. Selain itu lima metode klasifikasi, yaitu *SVM, NB, RF, kNN*, dan *DT* dibandingkan untuk menentukan metode klasifikasi terbaik untuk mendeteksi pernyataan non-kebutuhan.

1.6. Batasan Masalah

1. Data dokumen SKPL yang digunakan berbahasa inggris.
2. Pernyataan kebutuhan dinyatakan dalam bahasa alamiah.

(Halaman ini sengaja dikosongkan)

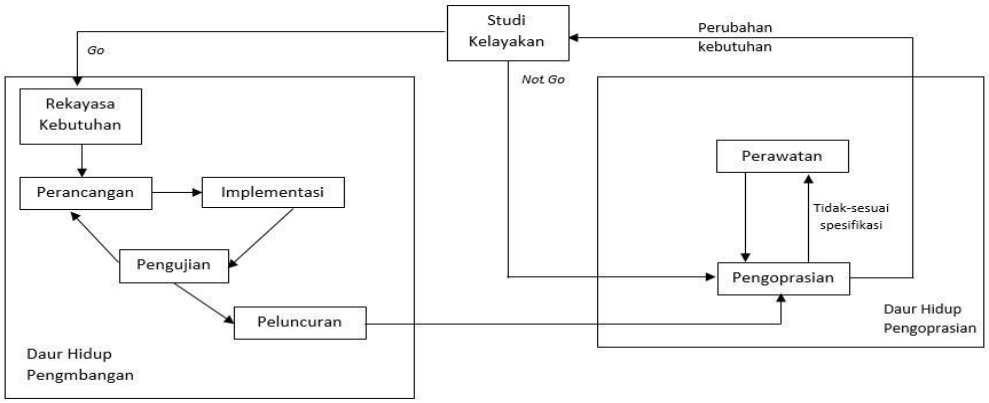
BAB 2
KAJIAN PUSTAKA

Pada bab ini akan dijelaskan tentang pustaka yang terkait dengan landasan penelitian. Pustaka yang terkait dianalisa dan dirangkum dalam bentuk studi komparasi.

2.1. Kebutuhan Perangkat Lunak

Kebutuhan dalam perangkat lunak dapat diartikan sebagai sekumpulan deskripsi tentang bagaimana suatu sistem berperilaku sehingga dapat memiliki nilai guna bagi penggunanya (Siahaan, 2012). Kebutuhan tersebut dispesifikasikan dalam tahap rekayasa kebutuhan.

Daur Hidup Perangkat Lunak dapat dilihat pada Gambar 2.1 yang menunjukkan bahwa dalam kebutuhan perangkat lunak terdapat dua siklus kehidupan utama yaitu daur hidup pengembangan dan daur hidup pengoperasian perangkat lunak, yang dihubungkan oleh studi kelayakan dan proses peluncuran.

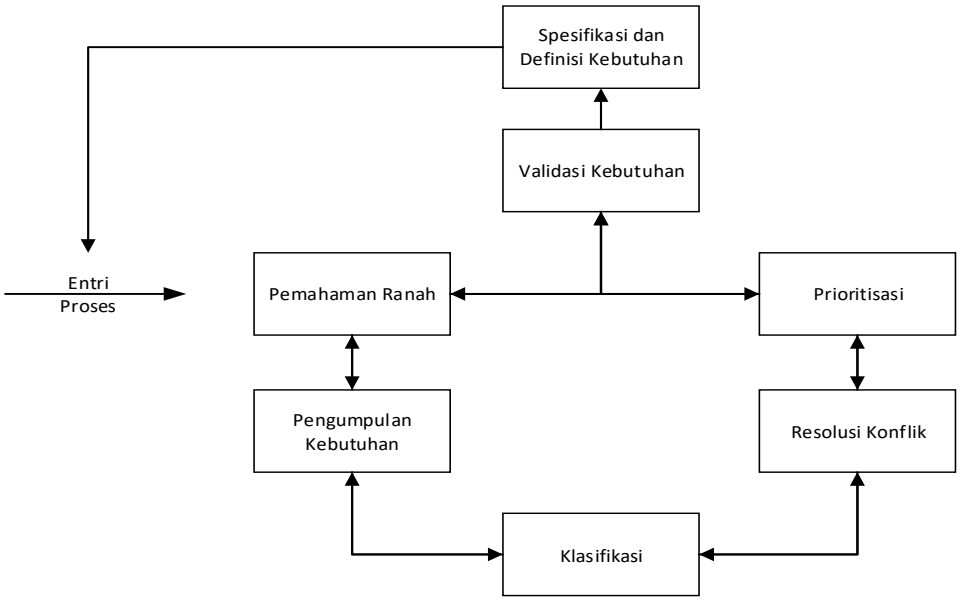


Gambar 2.1. Daur Hidup Perangkat Lunak

Terdapat dua siklus kehidupan dari suatu perangkat lunak, yaitu daur hidup pengembangan dan daur hidup pengoperasian. Dalam membuat sebuah aplikasi kita harus mengetahui kebutuhan dari aplikasi tersebut. Setelah mengetahui kebutuhannya, kita dapat merancang aplikasi tersebut. Dari perancangan kita dapat mengimplementasikan. Hasil dari implementasi selanjutnya diuji sampai benar-benar bisa digunakan sebelum pada akhirnya dilucurkan. Setelah daur hidup

pengembangan dilakukan sampai peluncuran, kita masuk ke daur hidup pengoperasian yaitu pengoperasian dan perawatan (Siahaan, 2012).

Proses rekayasa kebutuhan dilakukan dalam beberapa tahap seperti yang ditunjukkan pada Gambar 2.2 Proses dimulai dengan mempelajari tentang ranah lingkup permasalahan pada ranah pemahaman. Dari pemahaman ranah tersebut dikumpulkan kebutuhan perangkat lunak dan masuk pada klasifikasi dan diakhiri dengan pemeriksaan kembali kebutuhan-kebutuhan yang telah didapatkan terhadap relevansi dengan permasalahan utama. Kebutuhan-kebutuhan perangkat lunak yang telah berhasil dikumpulkan dirumuskan dalam sebuah dokumen SKPL (Siahaan, 2012).



Gambar 2.2 Tahapan Rekayasa Kebutuhan

2.2. Kebutuhan Pengguna

Kebutuhan pengguna adalah pernyataan atau gambaran pelayanan yang disediakan oleh sistem dan batasan-batasan bagi sistem yang akan dibangun agar pengguna dapat memenuhi tujuan penggunaan sistem.

Secara umum, kebutuhan perangkat lunak dapat dibagi ke dalam dua jenis yaitu: fungsional dan non-fungsional. Kebutuhan fungsional adalah kebutuhan yang dilihat dari sudut pandang pengguna. Kebutuhan non-fungsional adalah kebutuhan yang dilihat dari sudut pandang sistem. Selain dapat dikelompokkan ke

dalam kebutuhan fungsional dan non-funsional, kebutuhan juga dapat dikelompokkan lebih rinci lagi berdasarkan tingkat dari *stakeholder* (Siahaan, 2012).

Tabel 2.1 Contoh Kebutuhan Sistem Serta Kebutuhan Penggunaanya

Sistem	Kebutuhan Pengguna
SIM Akademik	- Mahasiswa hendak menyusun rencana kuliah semester ini - Orangtua hendak melihat prestasi belajar anak didik

2.3. Spesifikasi Kebutuhan Perangkat Lunak (SKPL)

SKPL merupakan keluaran dari tahap rekayasa kebutuhan di dalam Daur Hidup Perangkat Lunak seperti pada Gambar 2.1. Dokumen tersebut berisi tentang kumpulan kebutuhan-kebutuhan perangkat lunak yang akan dibuat. SKPL berfungsi sebagai landasan proses pengembangan perangkat lunak yang telah disetujui oleh pihak pengembang dan klien.

Terdapat empat pendekatan dalam menyusun dokumen SKPL yaitu:

1. Bahasa Alamiah

SKPL disusun dengan menggunakan bahasa sehari-hari.

2. Bahasa Alamiah Terstruktur

SKPL disusun dengan menggunakan bahasa sehari-hari yang ditata sesuai dengan struktur tertentu.

3. Bahasa Semi-formal

SKPL disusun dengan menggunakan bahasa grafikal yang dilengkapi dengan penjelasan teks.

4. Bahasa Formal

SKPL disusun sepenuhnya dengan notasi-notasi matematika seperti *finite state machine*.

Menurut survei yang telah dilakukan sebelumnya, pendekatan bahasa alamiah paling banyak digunakan untuk menyusun dokumen SKPL sebesar 71,8% (Rossi, 1999). Hal tersebut disebabkan karena pendekatan bahasa alamiah lebih mudah dipahami dan diimplementasi. Namun karena sifatnya yang subjektif, penggunaan bahasa alamiah dapat menimbulkan kerancuan dan kesalahan-

kesalahan karena merupakan salah satu dari jenis kesalahan yang sering dilakukan oleh perekayasa kebutuhan (Meyer, 1985).

Mayer mendeskripsikan secara mendalam tujuh kesalahan umum yang sering dilakukan pengembang perangkat lunak sebagai "*The seven sins of specifier*" salah satunya adalah *noise* atau derau, derau dapat muncul pada proses spesifikasi kebutuhan yaitu menyertakan suatu pernyataan yang tidak relevan dan suatu pernyataan non-kebutuham dengan keseluruhan kebutuhan perangkat lunak.

2.4. Derau pada SKPL

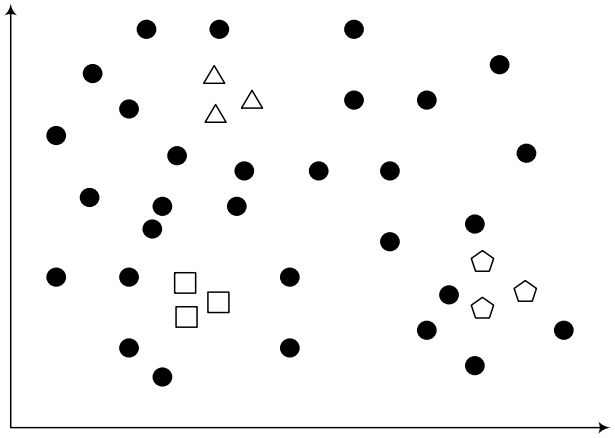
Tabel 2.2 Komparasi Penelitian Derau Sebelumnya

No	Judul	Masalah	Metode	Hasil
1	Simultaneous Clustering and Noise Detection for Theme-based Summarization (Cai *et al.*, 2011)	Mendeteksi derau berupa kalimat dalam dokumen teks di internet	Menggunakan *spectral clustering*	Berupa derau pada teks
2.	Detecting Deviations in Text Collection : An Approach Using Conceptual Graphs (Montes-y-Gómez, Gelbukh and López-López, 2002)	Mendeteksi penyimpangan teks (paper) "computer science".	Menggunakan Graphs Conceptual	Berupa visualisasi graph terhadap penyimpangan yang terjadi pada suatu dokumen teks.

Spesifikasi dalam rekayasa perangkat lunak adalah suatu aktivitas untuk mengumpulkan kebutuhan dalam pembuatan perangkat lunak. Kumpulan kebutuhan tersebut dirumuskan dalam SKPL. Namun terkadang terdapat suatu kebutuhan yang tidak relevan dengan kebutuhan lainnya. Kebutuhan tersebut dapat mengurangi kualitas informasi yang diberikan oleh SKPL dan dapat dianggap sebagai derau, seperti yang telah dijelaskan oleh Meyer (Meyer, 1985). Terdapat

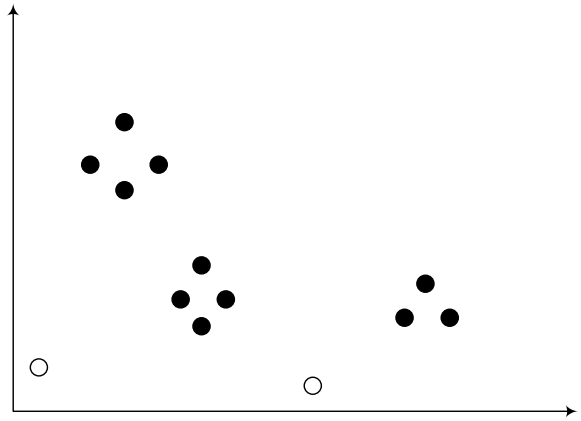
beberapa penelitian terkait yang telah dikembangkan tentang deteksi derau, seperti yang terdapat pada Tabel 2.2.

Derau adalah pernyataan non-kebutuhan dari sistem (yaitu, manajemen proyek, detail desain, rincian pelaksanaan, atau rencana pengujian) atau pernyataan yang berisi kebutuhan yang tidak relevan dengan proyek yang dijelaskan (Manek and Siahaan, 2019). Pada Gambar 2.3 titik-titik hitam mewakili pernyataan kebutuhan. Bentuk lainnya mewakili pernyataan tentang manajemen proyek, detail desain, detail implementasi, atau pernyataan non-kebutuhan lainnya. Derau ini cenderung lebih dekat dengan derau lainnya. Oleh karena itu digunakan pendekatan *supervised machine learning* untuk mendeteksi tipe derau ini.



Gambar 2.3 Ilustrasi Pernyataan Non-kebutuhan.

Derau juga terdeteksi sebagai pernyataan kebutuhan yang tidak relevan cenderung memiliki jarak semantik terhadap kelompok kebutuhan lain yang terkait dalam SKPL yang sama (Manek and Siahaan, 2019). Gambar 2.4 menggambarkan distribusi pernyataan dalam SKPL. Titik-titik hitam mewakili pernyataan kebutuhan. Titik-titik putih mewakili pernyataan derau. Dalam SKPL pernyataan kebutuhan yang relevan biasanya membahas topik dalam sistem yang dijelaskan. Kebutuhan yang tidak relevan akan membahas topik di luar topik umum yang sedang dibahas oleh seluruh kebutuhan yang relevan. Topik-topik itu secara semantik jauh dari topik-topik umum. Oleh karena itu, pendekatan yang akan digunakan adalah *unsupervised machine learning* untuk mendeteksi tipe derau ini.



Gambar 2.4 Ilustrasi Pernyataan yang Tidak Relevan.

2.5. Pengolahan Bahasa Alamiah

Salah satu teknik pemrosesan bahasa alami yang seringkali digunakan pada pra proses sebagai berikut:

1. *Tokenisasi*

Pada proses tokenisasi, teks dokumen dipecah menjadi bagian-bagian berupa kalimat, paragraf, atau dokumen.

2. *Lowercase*

Merubah semua huruf menjadi huruf kecil.

3. *Stopword Removal*

Proses untuk menghilangkan angka, simbol, dan kata ganti

4. *Lemmatizing*

Menghilangkan imbuhan dari kata-kata yang berimbuhan sehingga menjadi kata dasar.

2.6. Klasifikasi

Klasifikasi adalah pengelompokkan yang sistematis pada sejumlah objek, gagasan, buku atau benda-benda lain ke dalam kelas atau golongan tertentu berdasarkan ciri-ciri yang sama (Hamakonda, 1978).

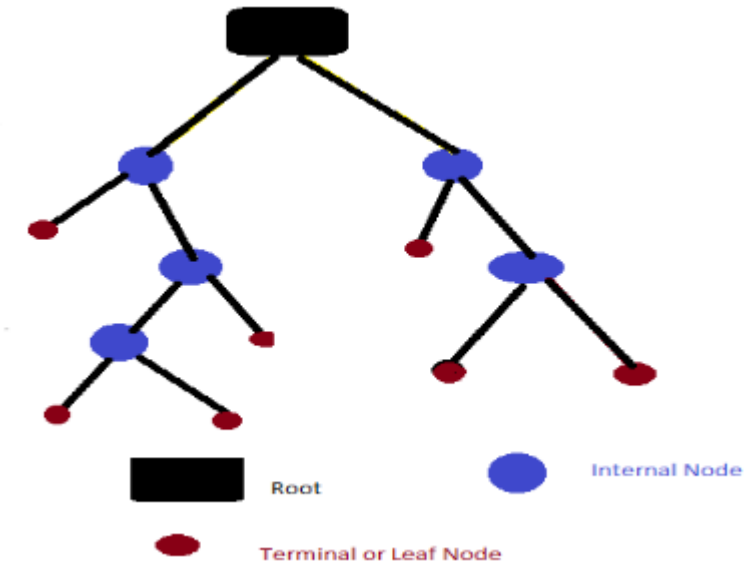
(Basuki, 1991) mengatakan bahwa klasifikasi berasal dari kata Latin “classis”. Klasifikasi adalah proses pengelompokan, artinya mengumpulkan benda/entitas yang sama serta memisahkan benda/entitas yang tidak sama. Secara

umum dapat dikatakan bahwa batasan klasifikasi adalah usaha menata alam pengetahuan ke dalam tata urutan sistematis.

Menurut (Liu *et al.*, 2010) Teknologi utama analisis informasi teks adalah klasifikasi teks. Melalui sistem klasifikasi teks otomatis, data teks dapat diklasifikasikan. Menemukan data yang lebih baik, menyaring dan menganalisa sumber informasi teks sangat diperlukan untuk membangun sistem klasifikasi teks yang efektif. Beberapa metode dalam klasifikasi teks adalah metode *Decision Tree*, metode *kNN*, metode *Naive Bayesian*, *Random Forest*, metode *SVM* dan lainnya.

2.6.1. Decision Tree

Metode *Decision Tree* adalah proses yang menggunakan informasi gain dalam teori informasi untuk menemukan sifat-sifat bidang dengan jumlah terbesar informasi dalam database sampel untuk membangun simpul pohon keputusan, dan membangun cabang pohon sesuai dengan sifat-sifat nilai bidang yang berbeda; dan kemudian ulangi membangun node dan cabang berikutnya untuk setiap cabang. Metode ini memiliki karakteristik klasifikasi cepat dan akurat. Namun, masih ada beberapa masalah yang belum terselesaikan, misalnya, kekosongan dalam pemrosesan data dan diskritisasi atribut berkelanjutan, dan lainnya. (Liu *et al.*, 2010)



Gambar 2.5 Struktur *Tree* (Ali *et al.*, 2012)

Pengelompokan data dalam *Decision Tree* didasarkan pada nilai atribut dari data yang diberikan. *Decision Tree* dibuat dari data pra-klasifikasi. Pembagian

menjadi beberapa kelas ditentukan oleh fitur-fitur yang paling baik dalam membagi data. Item data dipecah sesuai dengan nilai-nilai fitur. Proses ini diterapkan untuk setiap bagian yang terpisah dari item data secara rekursif. Proses berakhir ketika semua item data dalam subset memiliki kelas yang sama. (Ali *et al.*, 2012)

2.6.2. k-Nearest Neighbour (kNN)

Metode *kNN* adalah metode yang secara teori lebih matang. Misalkan setiap kelas berisi beberapa data sampel, dan setiap data memiliki tipe standar data yang unik. *kNN* adalah untuk mendapatkan data sampel k- terdekat dari data yang akan diklasifikasikan dengan menghitung jarak dari masing-masing data sampel ke data yang akan diklasifikasikan, dan jenis data sampel k yang menempati mayoritas, maka data yang akan diklasifikasikan menjadi milik ke kategori itu. Namun, karena sejumlah besar perhitungan, efisiensi metode *kNN* sangat berkurang ketika melatih sampel dengan lebih banyak atribut untuk diklasifikasikan. (Liu *et al.*, 2010)

2.6.3. Naive Bayesian

Metode *Naive Bayesian* adalah sejenis metode pembelajaran yang didasarkan pada teorema *Bayesian*, dan dapat digunakan untuk memprediksi kemungkinan keanggotaan kelas, dan itu memberikan kemungkinan teks milik kelas tertentu. Menurut hasil perkiraan, sampel ditugaskan ke probabilitas kategori tertinggi ketika diklasifikasikan. Secara teori, premis penerapan klasifikasi *Naive Bayesian* adalah bahwa nilai atribut sampel tidak tergantung pada sifat klasifikasi sampel, tetapi asumsi independensi atributnya mempengaruhi kinerja klasifikasinya. (Liu *et al.*, 2010)

2.6.4. Random Forest

Random Forest adalah salah satu metode non-parametrik terbaru. Ini adalah klasifikasi yang sangat akurat dan kuat terhadap derau. Dibandingkan dengan metode non-parametrik lainnya, pengaturan *Random Forest* sederhana, karena tidak diperlukan penyetelan parameter yang canggih. (Immitzer, Atzberger and Koukal, 2012)

2.6.5. Support Vector Machine (SVM)

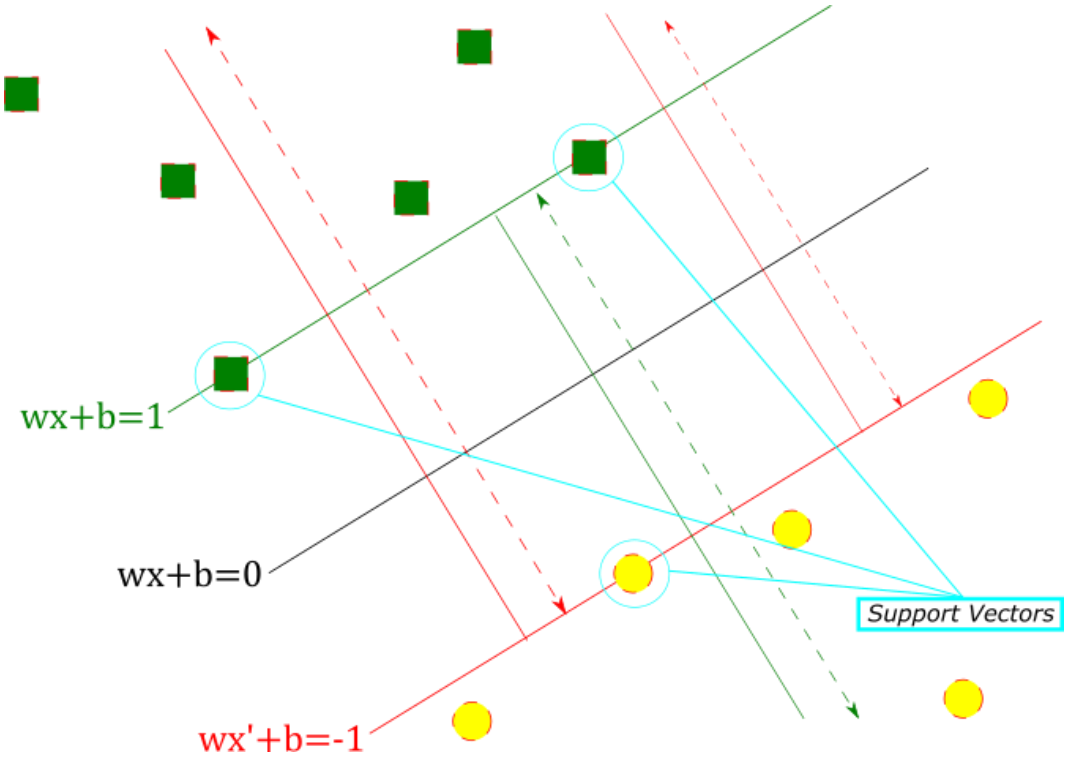
Tujuan dari menghitung *SVM* adalah untuk mengklasifikasikan semua data dengan benar. Untuk perhitungan matematis seperti pada persamaan berikut:

[a]. Jika $y_i = +1;\ wx_i + b \geq 1$

[b]. Jika $y_i = -1;\ wx_i + b \leq 1$

[c]. Untuk semua $i;\ y_i(w_i + b) \geq 1$

Dalam persamaan tersebut x adalah titik vektor dan w adalah berat dan juga sebuah vektor. Jadi untuk memisahkan data [a] harus selalu lebih besar dari nol. Di antara semua kemungkinan hyperplane, *SVM* memilih salah satu yang memiliki jarak hyperplane sebesar mungkin. Jika data pelatihan baik dan setiap vektor pengujian terletak dalam radius r dari vektor pelatihan. Hyperplane yang terpilih membagi dua garis antara titik-titik terdekat pada *convex hull* dari kedua dataset.



Gambar 2.6 Representasi hyperplane dan *support vector*

Jarak titik terdekat pada hyperplane ke asal dapat ditemukan dengan memaksimalkan x seperti x pada hyperplane. Demikian pula untuk titik di sisi lain menggunakan skenario yang sama. Dengan memecahkan dan mengurangi dua jarak tersebut, didapatkan jarak penjumlahan dari hyperplane pemisah ke titik terdekat. Margin Maksimum $M = 2/\|w\|$

Memaksimalkan margin sama seperti minimum. Optimasi kuadratik dengan batasan linear diperlukan untuk menentukan w dan b. Persamaan untuk menentukan w dan b adalah $\Phi(w) = {}^{1}/_{2}\,|w' \parallel w|$;

Dan untuk semua $\{(x_i, y_i)\}$: $y_i(w * x_i + b) \geq 1$.

Untuk menyelesaikannya dengan persamaan $w = \sum a_i * x_i$; $b = y_k - w * x_k$untuk setiap x_k sedemikian hingga $ak \neq 0$

Bentuk fungsi klasifikasi adalah $f(x) = \sum a_i y_i x_i * x + b$

Beberapa masalah klasifikasi tidak dapat dipisahkan secara linear. Oleh karena itu, kapasitas hyperplane klasifikasi akan terbatas. Dalam hal ini, (Liu *et al.*, 2010) memperkenalkan variabel slack $\varepsilon_i, i = 1,2, \cdots n$. Kondisi konstrainnya adalah

$$y_i[(w \cdot x_i) + b] \geq 1 - \varepsilon, i = 1,2, \cdots n$$

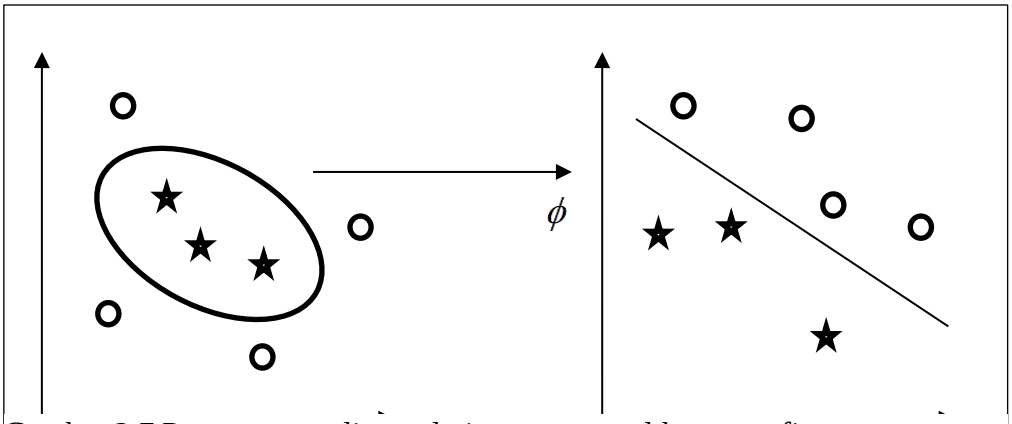
Kemudian fungsi objektifnya adalah

$$L(w, \varepsilon) = \frac{1}{2} \| w \|^2 + C| \sum_{i=1}^{n} \varepsilon_i |, i = 1,2, \cdots n$$

Oleh karena itu, permukaan pemisah optimal yang luas dapat diperoleh dengan pertimbangan komprehensif antara sampel kesalahan klasifikasi minimum dan interval kelas maksimum. Di mana C adalah konstanta dan lebih besar dari nol, dikenal sebagai koefisien penalti, yang mengontrol tingkat hukuman sampel benar atau salah. Maka fungsi hyper-plane optimal diperoleh sebagai berikut:

$$f(x) = sgn\left(\sum_{i=1}^{n} a_i y_i \phi(x) \cdot \phi(x_i) + b\right)$$

Nilai sebenarnya dari *SVM* digunakan untuk menyelesaikan masalah nonlinear. Metode ini melalui pemetaan nonlinier φ untuk memetakan ruang sampel ke ruang fitur berdimensi tinggi atau bahkan tak terbatas, sehingga metode *SVM* linier dalam ruang fitur dapat diterapkan untuk menyelesaikan masalah klasifikasi



Gambar 2.7 Pemetaan nonlinear dari ruang sampel ke ruang fitur

nonlinier di ruang sampel. Pemetaan nonlinear dari ruang sampel ke ruang fitur ditunjukkan pada Gambar 2.7 Pemetaan nonlinear dari ruang sampel ke ruang fitur.

Metode *support vector machine* (*SVM*), *naïve bayes* (*NB*), *random forest* (*RF*), *k-nearest neighbor* (*kNN*), dan *Decision Tree* akan digunakan untuk menentukan metode mana yang memiliki hasil terbaik (Liu *et al.*, 2010; Amancio *et al.*, 2014; Colditz, 2015). Pengukuran yang digunakan adalah tingkat akurasi, yang didefinisikan sebagai *true* positif ditambahkan *true* negatif, kemudian dibagi dengan total data.

2.7. Koefisien Korelasi *Pearson*

Koefisien Korelasi *Pearson* adalah metrik statistik yang mengukur kekuatan dan arah hubungan linear antara dua variabel acak. Cara telah banyak digunakan dalam banyak aplikasi seperti estimasi waktu tunda, pengenalan pola, analisis data, untuk beberapa nama. (Benesty, Chen and Huang, 2008)

Koefisien korelasi Pearson didefinisikan sebagai:

$$\rho(x, y) = \frac{E[xy]}{\sigma_x \sigma_y}$$

2.8. Kakas Bantu Pemrosesan Bahasa Alamiah

1. *Cosine Similarity*

   Cosine similarity merupakan salah satu metode pengukuran kemiripan antara dua dokumen atau teks. Pada *cosine similarity* dokumen atau teks dianggap sebagai vektor. Rumus *cosine similarity* (Garcia, 2015):

   $$\boldsymbol{similarity} = \cos(\boldsymbol{\theta}) = \frac{\boldsymbol{R_i} \times \boldsymbol{R_j}}{\|\boldsymbol{R_i}\| \times \|\boldsymbol{R_j}\|} = \frac{\sum_{k=1}^{n} (\boldsymbol{R_i} \times \boldsymbol{R_j})}{\sqrt{\sum_{k=1}^{n} \boldsymbol{R_i}^2} \times \sqrt{\sum_{k=1}^{n} \boldsymbol{R_j}^2}} \qquad (1.1)$$

   Di mana R_i dan R_j adalah vektor *term frequency* dari dokumen, dengan nilai *cosine similarity* berada pada *range* 0-1.

2. *Koefisien Kappa*

   Koefisien Kappa berfungsi untuk menghitung tingkat persetujuan antara dua penilai dalam memberikan penilaian pada suatu kasus (Cohen, 1960). Koefisien Kappa memperhitungkan probabilitas terjadinya persetujuan dan

penilaian di antara kedua penilai. Nilai koefisien Kappa selalu diantara 0 dan 1, dengan nilai 1 menunjukan kesempurnaan tingkat persetujuan.

Tabel 2.4. Masukan Perhitungan Nilai Koefisien *Kappa*

		Penilai 2		
	Nilai	A	B	**Total**
Penilai 1	A	n_aa	n_ab	n_a*
	B	n_ba	n_bb	n_b*
	Total	n_*a	n_*b	

Pada Tabel 2.4 dapat dilihat bahwa terdapat 2 penilai yaitu Penilai 1 dan Penilai 2. Setiap penilai dapat memberikan dua kategori penilaian yaitu nilai a atau nilai b pada suatu kasus. Elemen dari Tabel 2.4 merupakan jumlah kejadian Penilai 1 dan Penilai 2 memberikan penilaian suatu kasus. Penjelasan variabel pada elemen Tabel 2.4 adalah sebagai berikut :

- *n_aa* adalah jumlah kejadian Penilai 1 dan Penilai 2 memberikan nilai a pada suatu kasus.
- *n _ab* adalah jumlah kejadian *Penilai 1* memberikan nilai *a* dan *Penilai 2* memberikan nilai *b* pada suatu kasus.
- *n _ba* adalah jumlah kejadian *Penilai 1* memberikan nilai *b* dan *Penilai 2* memberikan nilai *a* pada suatu kasus.
- *n _bb* adalah jumlah kejadian *Penilai 1* dan *Penilai 2* memberikan nilai *b* pada suatu kasus.

Nilai koefisien *Kappa k* dapat dihitung jika jumlah kejadian penilaian *Penilai 1* dan *Penilai 2* telah diketahui. Pertama-tama probabilitas persetujuan yang telah diobservasi antara kedua penilai, atau P_o dihitung dengan Persamaan (2.1).

$$P_o = \frac{n\_aa + n\_bb}{n\_aa + n\_ab + n\_ba + n\_bb} \quad (2.1)$$

Lalu probabilitas kedua penilai memberikan nilai *a* secara acak, atau P_a, dihitung, begitu juga dengan nilai P_b dengan Persamaan (2.2) dan (2.3). Sehingga nilai probabilitas persetujuan secara acak, atau P_e dapat dihitung dengan Persamaan (2.4).

$$P_a = \frac{n\_a*}{n\_aa + n\_ab + n\_ba + n\_bb} \cdot \frac{n\_*a}{n\_aa + n\_ab + n\_ba + n\_bb} \quad (2.2)$$

$$P_b = \frac{n\_b*}{n\_aa + n\_ab + n\_ba + n\_bb} \cdot \frac{n\_*b}{n\_aa + n\_ab + n\_ba + n\_bb} \quad (2.3)$$

$$P_e = P_a + P_b \quad (2.4)$$

Nilai koefisien *Kappa k* dapat dihitung dengan Persamaan

$$k = \frac{P_o - P_e}{1 - P_e} \quad (2.5)$$

Hasil perhitungan nilai Kappa tersebut kemudian dapat diintrepretasikan secara kuantitaf dengan menggunakan tabel intrepretasi seperti pada Tabel 2.4

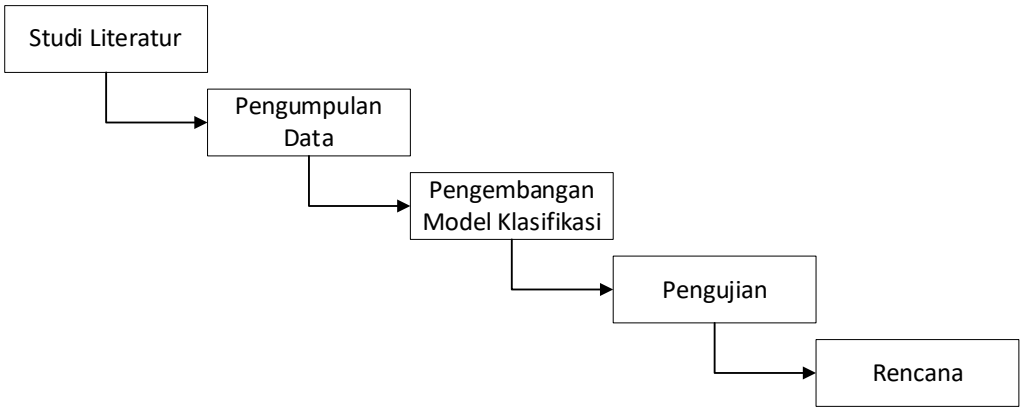
Tabel 2.3 Interpretasi Nilai Gwet AC1

Index Kappa	**Proporsi Kesepakatan**
< 0	Rendah
0.01 - 0.20	Sedikit
0.21 - 0.40	Cukup
0.41 - 0.60	Sedang
0.61 - 0.80	Substansial
0.81 – 1	Hampir sempurna

(Halaman ini sengaja dikosongkan)

BAB 3
METODE PENELITIAN

Bab ini akan memaparkan tentang metodologi penelitian yang digunakan pada penelitian ini, yang terdiri dari (1) studi literatur, (2) pengumpulan data, (3) pengembangan model klasifikasi non-kebutuhan, (4) pengujian, dan (5) rencana. Ilustrasi alur metolodogi penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1 Alur Metodologi Penelitian

3.1. Studi Literatur

Penelitian ini diawali dengan studi literatur terkait dengan topik yang akan dibahas dalam penelitian ini. Beberapa topik yang akan dibahas adalah derau pada SKPL, metode pengolahan bahasa ilmiah, dan metode klasifikasi. Literatur yang akan digunakan untuk topik derau pada SKPL adalah yang membahas berbagai macam derau yang ada pada kebutuhan SKPL. Dengan adanya literatur-literatur tersebut, diharapkan kerangka kerja untuk deteksi derau pada SKPL dapat dibuat dengan baik. Literatur yang digunakan untuk pengolahan bahasa ilmiah adalah literatur yang membahas penerapan prapemrosesan pada bahasa ilmiah. Literatur yang digunakan untuk metode klasifikasi adalah literatur yang membahas klasifikasi teks dan sebagian membahas tentang klasifikasi gambar. Sumber literatur yang akan digunakan adalah buku, jurnal dan konferensi internasional yang dipublikasi melalui sciencedirect, IEEE, atau springer. Literatur yang diambil menggunakan kata kunci *noise detection,* analisa kebutuhan, *text preprocessing, text classification, decision tree,* dan *random forest*.

Tabel 3.1 Dataset

IDE	Project Name	Original Req.	Non-Req.	Irrelevant Req.	Crossed Req.
DA-1	*Submit Job*	13	0	0	6
DA-2	*System Administrator*	17	0	1	6
DA-3	*Archive Administrator*	39	0	0	6
DA_4	SPG *Application*	6	1	3	2
DA-5	*System Administrator Staff*	33	0	0	7
DA-6	*Software Development*	65	28	26	7
DA-7	*Display System*	106	9	13	8
DA-8	*Internet Access*	64	4	4	8
DA-9	*Meeting Initiator*	33	0	0	6
DA-10	*Online System*	17	0	0	4
DA-11	*Library System*	86	11	12	9
DA-12	IMSETY *System*	70	0	0	6
DA-13	*Manage Student Information*	24	1	2	3
DA-14	PHP *Project*	75	3	2	8
Total		**648**			**86**

3.2. Pengumpulan Data

Pengumpulan data bertujuan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan penelitian. Data yang dikumpulkan akan menghasilkan daftar pernyataan dari beberapa dokumen SKPL. Tahap-tahap yang dilakukan adalah mengumpulkan dokumen SKPL, mengekstraksi secara manual pada csv, dan menandai pernyataan kebutuhan tersebut sebagai kebutuhan, non-kebutuhan, atau tidak relevan.

3.2.1. Dataset

Penelitian ini menggunakan 648 pernyataan kebutuhan perangkat lunak yang diekstraksi secara manual dari 14 dokumen SKPL. Tabel 3.1 menunjukkan dataset yang digunakan dalam penelitian ini. Setiap dokumen mengacu pada proyek yang berbeda dari berbagai domain masalah. Setiap pernyataan kebutuhan dilabeli oleh lima anotator dengan kebutuhan, non-kebutuhan, atau tidak relevan. Label adalah nilai tipe boolean. Anotator akan melabeli pernyataan dengan 1 (*true*) pada kolom derau, jika dan hanya jika dia berpikir bahwa pernyataan itu adalah derau. Anotator akan memberi label status dengan 0 (*false*) pada kolom derau, jika dan hanya jika dia berpikir bahwa pernyataan tersebut adalah pernyataan kebutuhan yang relevan. Hal yang sama berlaku untuk kolom kebutuhan/non-kebutuhan untuk setiap pernyataan kebutuhan. Jumlah derau dalam SKPL relatif kecil. Oleh karena itu, untuk meminimalkan ketidakseimbangan, ditambahkan dataset dengan

pernyataan kebutuhan dari dataset lintas domain. Setiap dataset lintas domain akan menyumbang hanya satu derau ke sebuah dataset.

Tabel 3.2 Statistik Dataset

Keterangan	Jumlah
Banyak Pernyataan	648
Rata-rata Jumlah Kata	18
Maksimal Jumlah Kata	123
Minimal Jumlah Kata	4

Tabel 3.2 menunjukkan statistik dari dataset yang akan digunakan pada penelitian ini. Statistik yang ditunjukkan antara lain berapa banyak pernyataan, rata-rata jumlah kata, maksimal dan minimal jumlah kata, berapa banyak kata benda, dan berapa banyak kata kerja.

Langkah terakhir dalam proses ini adalah membentuk standar untuk non-kebutuhan dan dataset. Nilai standar dari sebuah pernyataan ditentukan oleh suara mayoritas dari lima anotator. Nilai standar yang ditentukan adalah di mana setidaknya tiga anotator menyetujui anotasi yang sama.

3.3. Pengembangan Model Klasifikasi Pernyataan Non-Kebutuhan

Untuk mendeteksi pernyataan non-kebutuhan, yang dilakukan adalah dengan mengklasifikasinya dan mengukur model klasifikasi.

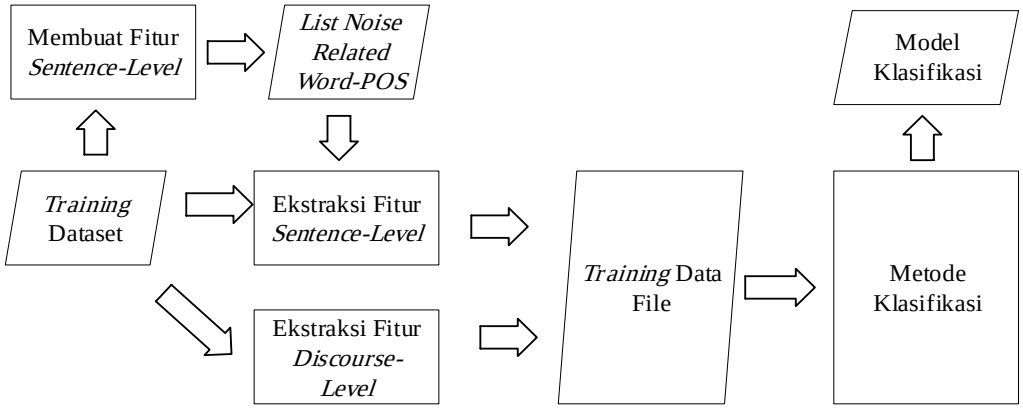
a) Klasifikasi

   Berdasarkan model klasifikasi yang dibangun dari proses sebelumnya, pernyataan kebutuhan dipisahkan dari pernyataan non-kebutuhan. Seperti yang telah disebutkan, metode yang digunakan ada lima, yaitu, *support vector machine* (*SVM*), *naïve bayes* (*NB*), *random forest* (*RF*), *k-nearest neighbor* (*kNN*), dan *Decision Tree*. Kinerja masing-masing metode diukur berdasarkan lima pengukuran yang berbeda, yaitu, *precision*, *recall*, *F1-score*, *sensitivity*, dan *specificity*. Metode terbaik yang memiliki kinerja tertinggi secara keseluruhan akan dipilih untuk proses selanjutnya, yaitu seleksi derau.

b) Mengukur Model Klasifikasi Non-Kebutuhan

Setelah mengekstrak fitur *sentence-level* dan *discourse-level* dari data training, selanjutnya akan dibangun model klasifikasi. Model dengan performa paling bagus akan digunakan untuk proses seleksi derau.

Langkah untuk membangun model klasifikasi pernyataan non-kebutuhan dapat dilihat pada Gambar 3.4. Langkah pertama, menggunakan fitur global dimana nilai-nilai fitur yang berlaku untuk semua pernyataan individu mengabaikan domain masalahnya. Langkah ini terdiri dari tiga proses utama, yaitu, membuat fitur *sentence-level*, membuat fitur *discourse-level*, dan membuat model klasifikasi derau.

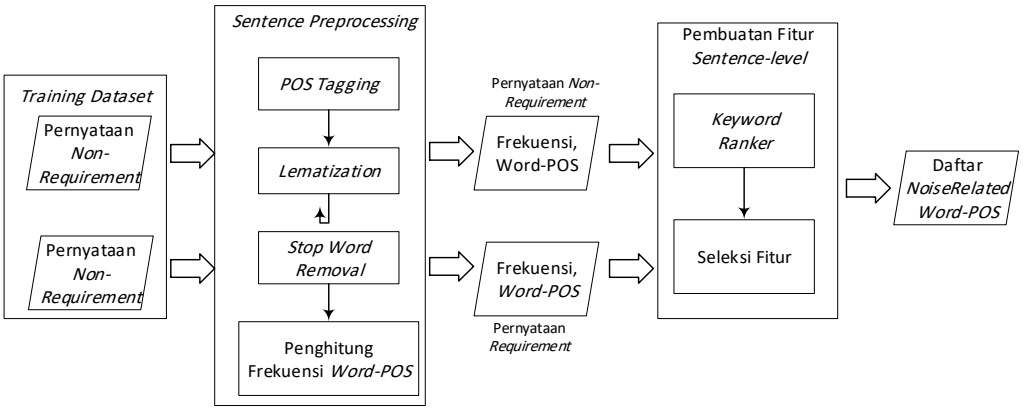


Gambar 3.2 Membangun Model Klasifikasi

Langkah pertama dalam mengidentifikasi derau pada SKPL adalah membuat model klasifikasi derau. Model *supervised* ini terletak pada sekumpulan fitur *sentence*, yaitu fitur *sentence-level* dan fitur *discourse-level*. Fitur *Sentence-level* diekstraksi dari dataset training, yang mengandung kalimat kebutuhan dan kalimat non-kebutuhan. Frekuensi setiap *word-POS* dalam daftar *noise-related word-POS* yang muncul dihitung pada setiap pernyataan. Fitur *Discourse-level* diekstraksi dari dataset training yang sama. Satu set dari data statistik setiap pernyataan sehubungan dengan sisa pernyataan dihitung dalam SKPL yang sama.

3.3.1. Membuat Fitur *Sentence-Level*

Gambar 3.5 menunjukkan pembuatan fitur *sentence-level* yang terdiri dari dua proses utama, yaitu prapemrosesan *sentence* dan pembuatan fitur. Pemrosesan *sentence* melibatkan penandaan *part-of-speech (POS), lemmatization, stopword removal,* dan penghitung frekuensi tag *word-POS*.

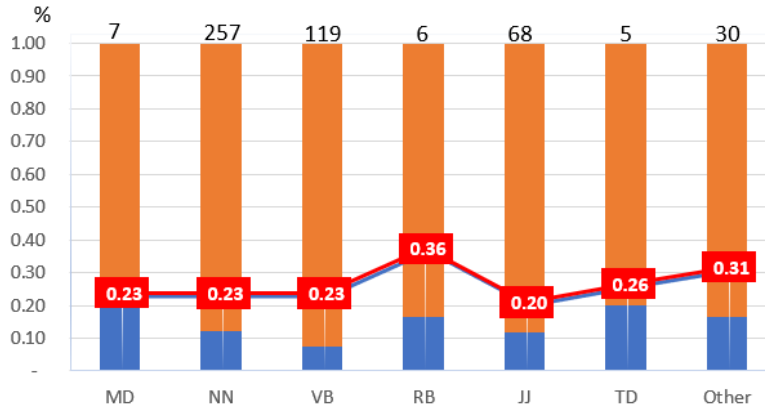


Gambar 3.3 Pembuatan Fitur *Noise-Related Sentence-Level* Secara Otomatis

Pembangunan Fitur *Sentence-level* terdiri dari dua proses utama, yaitu, *keyword ranker* dan seleksi fitur. *Keyword ranker* menggunakan output dari proses sebelumnya untuk menghitung kekuatan dari diskriminasi (dalam hal kemungkinan rasio atau LR) dari setiap tag *word-POS*. LR dari sebuah *word-POS* dapat dihitung menggunakan persamaan 3.1

$$LR(word - POS) = \frac{frequency\ of\ (word{-}POS) in\ Corpus\ Non{-}Requirements}{frequency\ of\ (word{-}POS) in\ both\ Corpus} \qquad (3.1)$$

Gambar 3.6 menunjukkan presentase dari *noise-related word-POS* dengan total *word-POS* yang ditemukan di kedua korpus. Bilah oranye mewakili jumlah *word-POS* unik yang ditemukan di kedua korpus. Bilah biru mewakili jumlah *word-POS* yang buruk. Angka di dalam blok merah-biru mewakili presentase kemunculan *word-POS* dalam korpus non-kebutuhan terhadap total kemunculan *word-POS* di kedua korpus.



Gambar 3.4 *BaselineLR* dari Setiap Tag dan Presentase dari *Word-POS* yang Buruk

Ini adalah nilai *BaselineLR* dari setiap tag *POS*. *BaselineLR* dari tag *POS* dihitung menggunakan persamaan 3.2

$$BaselineLR(POS_i) = \frac{\sum_{j=0}^{N} frequency\ of\ word_j{-}POS_i\ in\ Non{-}Requirements}{\sum_{j=1}^{N} frequency\ of\ word_j{-}POS_i\ in\ both\ Corpus} \quad (3.2)$$

Tabel 3.3 menunjukkan *LR* dari setiap *word-POS* yang dimiliki tag *VB.*

Tabel 3.3 *LR* dari Setiap *Word-POS* dari Tag VB

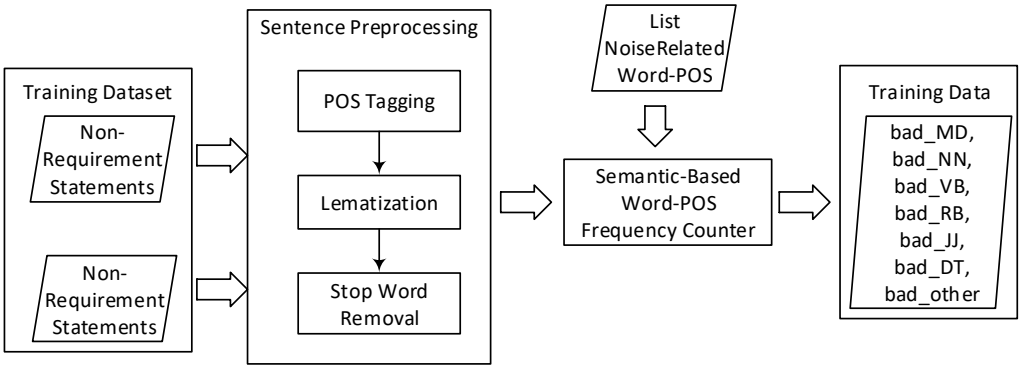
Word	LR
Develop	0.8
Describe	0.692
Start	0.667
Determine	0.667
Make	0.667
Work	0.619
Require	0.571
Need	0.571
Define	0.538
Monitor	0.5
Delegate	0.5
…	
Base	0.235

Tabel 3.4 Jumlah dari *Word-POS* dan Frekuensi pada setiap Korpus

POS Tag	All Words	Bad Words	Good Corpus	Bad Corpus
MD	7	2	462	140
NN	257	31	3065	919
VB	119	9	1014	305
RB	6	1	42	24
JJ	68	8	437	112
TD	5	1	83	29
Other	30	5	573	256

Pada Tabel 3.3, hanya sembilan *word-POS* tertinggi pada tag *VB* (*LR* > 0.5) yang akan dimasukkan pada daftar *noise-related word-POS.* Tabel 3.4 menunjukkan hasil dari proses ini untuk semua tag *POS.* Dari eksperimen, langkah proses memilih sekitar 12-29% dari *word-POS* yang ditemukan pada korpus non-kebutuhan sebagai *noise-related word-POS.*

3.3.2. Membuat Model Klasifikasi Non-Kebutuhan

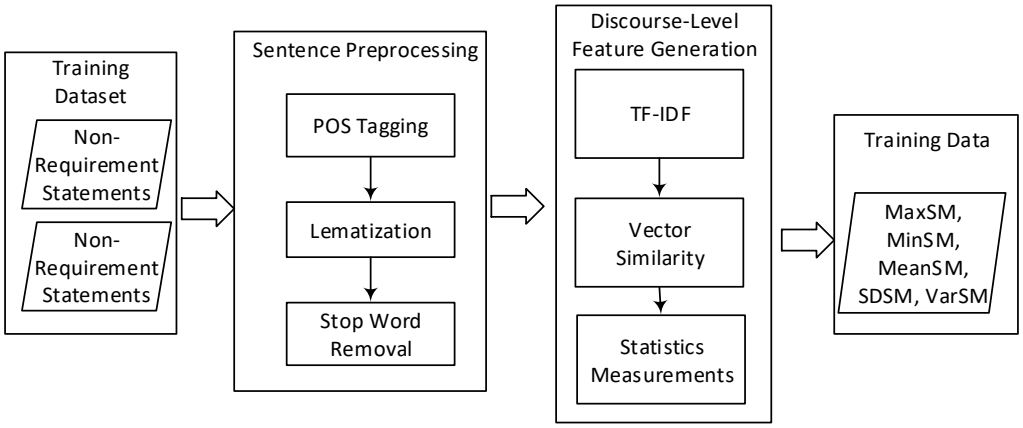


Gambar 3.5 Ekstraksi Fitur *Sentence-Level*

Proses selanjutnya adalah mengekstraksi fitur *sentence-level* dan fitur *discourse-level* dari data pelatihan. Data pelatihan digunakan untuk membangun model klasifikasi untuk mendeteksi pernyataan non-kebutuhan. Gambar 3.7 dan 3.8 menunjukkan diagram proses untuk mengekstrak fitur *sentence-level* dan *discourse-level*. korpus berisi sejumlah kosakata yang terbatas, sehingga perlu memperluas cakupan agar dapat menangani proyek lintas domain yang lebih luas. *WordNet* tesaurus digunakan untuk menemukan kata-kata yang mirip dan diberikan *treshold* kemiripan.

Tabel 3.5 Daftar Pernyataan Kebutuhan dari SKPL

ID	Statement	Label
F1	Register a new lecturer's account	0
F2	Delete lecturer's account	0
F3	Update lecturer's account information	0
F4	Add lecturer's availability status	0
F5	Update lecturer's availability status	0
F6	Delete lecturer's availability status	0
F7	Show lecturer's information and availability status	0
F8	Find lecturer's account	0
F9	The tester would be the lecturers	1



Gambar 3.6 Ekstraksi Fitur *Discourse-Level*

Dalam fitur ekstraksi *discourse-level*, setiap pernyataan dari *corpora* adalah prapemrosesan dan berubah menjadi vektor dalam ruang n-dimensi, di mana n adalah beberapa istilah penting sehubungan dengan SKPL seperti yang diilustrasikan persyaratan spesifikasi dalam Tabel 3.5. Ada sembilan pernyataan yang pernyataan terakhir yang diberi label derau. Dengan menggunakan TF-IDF (dengan frekuensi minimum adalah 2), dapat dibentuk sebuah vektor untuk setiap pernyataan seperti yang ditunjukkan pada Tabel 3.5. Proses menghitung kesamaan antara setiap vektor pada setiap dokumen menggunakan *cosine similarity*. Hal ini membentuk matriks dua dimensi V dari ukuran nxn. Mengingat ini *similarity* matriks V, proses ini menghitung lima fitur Statistik lokal, yaitu, nilai minimum ukuran *similarity* (MinSM), maksimum ukuran *similarity* (MaxSM), mean dari ukuran *similarity* (MeanSM), standar deviasi dari ukuran *similarity* (SDSM), dan varians dari ukuran *similarity* (VarSM). Tabel 3.6 menunjukkan contoh dari menghitung fitur lokal dari spesifikasi persyaratan perangkat lunak yang disebutkan dalam Tabel 3.4 dimana pernyataan F9 cenderung menghasilkan nilai yang berbeda dibandingkan dengan sisa pernyataan. Hal ini karena F9 membahas pernyataan non-kebutuhan (yaitu, pengujian terkait pernyataan), yang mungkin berisi istilah non-spesifik sehubungan dengan sisa pernyataan kebutuhan.

Tabel 3.6 Hasil dari Perhitungan *Cosine Similarity* dan Ekstraksi Fitur pada Matriks V

	F01	F02	F03	F04	F05	F06	F07	F08	F09	MinSM	MaxSM	MeanSM	SDSM	VarSM
F01		0.47	0.36	0.00	0.00	0.00	0.00	1.00	0.00	0.36	1.00	0.27	0.34	0.12
F02	0.47		0.17	0.00	0.00	0.70	0.00	0.47	0.00	0.17	0.70	0.25	0.26	0.07
F03	0.36	0.17		0.00	0.53	0.00	0.53	0.36	0.00	0.17	0.53	0.27	0.22	0.05
F04	0.00	0.00	0.00		0.61	0.61	0.61	0.00	0.00	0.61	0.61	0.31	0.32	0.10
F05	0.00	0.00	0.53	0.61		0.37	0.37	0.00	0.00	0.37	0.61	0.28	0.25	0.06
F06	0.00	0.70	0.00	0.61	0.37		0.37	0.00	0.00	0.37	0.70	0.30	0.28	0.08
F07	0.00	0.00	0.53	0.61	0.37	0.37		0.00	0.00	0.37	0.61	0.28	0.25	0.06
F08	1.00	0.47	0.36	0.00	0.00	0.00	0.00		0.00	0.36	1.00	0.27	0.34	0.12
F09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00		0.00	0.00	0.00	0.00	0.00

Tabel 3.7 Pernyataan Vektor berdasarkan TF-IDF

IDE	Lecture	Account	Delete	update	information	available	status
F01	0.00	0.81	0.00	0.00	0.00	0.00	0.00
F02	0.00	0.81	1.50	0.00	0.00	0.00	0.00
F03	0.00	0.81	0.00	1.50	1.50	0.00	0.00
F04	0.00	0.00	0.00	0.00	0.00	0.81	0.81
F05	0.00	0.00	0.00	1.50	0.00	0.81	0.81
F06	0.00	0.00	1.50	0.00	0.00	0.81	0.81
F07	0.00	0.00	0.00	0.00	1.50	0.81	0.81
F08	0.00	0.81	0.00	0.00	0.00	0.00	0.00
F09	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Pada setiap fitur statistik lokal, proses normalisasi sehubungan dengan nilai SKPL lain dalam dataset untuk mendapatkan fitur statistik global prosesnya menggunakan persamaan 3.4.

$$X' = a + \frac{(X - X_{\min})(b-a)}{X_{max} - X_{min}} \quad (3.4)$$

X', normalisasi fitur statistik lokal, akan memiliki nilai antara a dan b. Nilai a dan b masing-masing mewakili titik minimum dan maksimum dari fitur statistik lokal. Misalnya, rata-rata ukuran *similarity* (meanSM), keseluruhan domain dalam dataset. Titik minimum dan maksimum ditentukan menggunakan persamaan 3.5.

$$x_{max} \in X \quad is\ a\ global\ maximum\ of\ function\ f: X \to R, if\ (\forall x \in X)\ f(x_{max}) \leq f(x) \quad (3.5)$$

3.4. Pengujian

Data uji diperoleh dari dataset yang ada berupa kebutuhan di dalam SKPL. Sebelumnya, kebutuhan tersebut diberi label berupa nilai boolean 1 atau 0. Label 1 menandakan bahwa kebutuhan tersebut dianggap derau oleh penulis. Pelabelan data uji tersebut merepresentasikan penilaian derau oleh penulis.

Pada pengujian, pendeteksi derau akan menghasilkan prediksi kebutuhan yang dianggap sebagai derau. Kedua hasil derau dari pendeteksi derau dan penilaian penulis akan dibandingkan menggunakan kakas bantu yaitu menghitung nilai indeks koefisien Kappa, seperti yang telah dijelaskan pada subbab sebelumnya. Nilai dari koefisien Kappa dapat merepresentasikan tingkat konsistensi hasil derau yang diberikan oleh pendeteksi derau dengan penilaian penulis.

Adapun skenario pengujian uji coba yang dilakukan sebagai berikut:

a) Mengekstraksi pernyataan kebutuhan dari setiap dokumen SKPL yang ada ke dalam format csv, seperti contoh: KODE; PERNYATAAN_ KEBUTUHAN; LABEL. Setiap SKPL akan disimpan ke dalam satu buah berkas csv. Setiap berkas csv merupakan satu buah data set pengujian. Berikut contoh berkas csv dari daftar kebutuhan pada sebuah SKPL.

   R01; Operator dapat membuka berkas.; 0

   R02; Mahasiswa dapat memilih mata kuliah yang hendak diambil.; 0

   R03; Pemegang kartu dapat melihat saldo tabungannya saat ini.; 0

   NR01; Sistem melakukan backup data minimal setiap satu minggu sekali.; 0

b) Memberi label pada setiap dataset. Proses pelabelan ini melibatkan ahli dalam bidang rekayasa perangkat lunak. Untuk setiap pernyataan dalam sebuah dataset, ahli akan diminta untuk memberikan label 0 jika pernyataan kebutuhan tersebut bukan derau, dan label 1 pernyataan kebutuhan tersebut derau. Ada tiga orang ahli yang akan dilibatkan dalam pelabelan pernyataan kebutuhan ini. Dari setiap dataset, maka akan dibentuk dataset dengan dua kesepakatan dan dataset dengan tiga kesepakatan. Dataset dengan dua kesepakatan adalah dataset dengan label akhir merupakan label mayoritas diantara label yang diberikan oleh ketiga ahli. Dataset dengan tiga kesepakatan adalah dataset dengan aturan sebagai berikut: (a) jika salah satu ahli memberi label 0, maka label akhir dari pernyataan kebutuhan tersebut adalah 0, dan (b) jika semua ahli memberi label 1, maka label akhir dari pernyataan kebutuhan tersebut adalah 1.

c) Melakukan klasifikasi menggunakan beberapa metode yang ada pada aplikasi weka. *SVM* dikonfigurasi dengan *batchSize* 100, nilai *c* 1.0, dan kernel *PolyKernel*. *Naive Bayes* dikonfigurasi dengan *batchSize* 100. *Random Forest* dikonfigurasi dengan *batchSize* 100 dan *maxDepth* tak terbatas. *Decision Tree*

dikonfigurasi dengan *batchSize* 100. *kNN* dikonfigurasi dengan *batchSize* 100 dan *nearestNeighbourSearchAlgoritm LinearNNearch*. Klasifikasi dilakukan sebanyak jumlah dataset secara memutar, dengan komposisi satu projek sebagai data uji dan sisanya sebagai data latih. Sebagai contohnya adalah pertama-tama, id proyek DA-1 sebagai data uji, id proyek DA-2 sampai dengan id proyek DA-14 sebagai data latih. Kedua, id proyek DA-2 sebagai data uji, id proyek DA-1, DA-3 sampai dengan id proyek DA-14 sebagai data latih. Ketiga, id proyek DA-3 sebagai data uji, id proyek DA-1, DA-2, DA-4 sampai dengan id proyek DA-14 sebagai data latih, dan seterusnya sampai dengan id proyek DA-14 sebagai data uji, dan id proyek DA-1 sampai dengan id proyek DA-13 sebagai data latih. Sebelum data uji diatur manual, proses pelatihan pada setiap dataset dilakukan dengan menggunakan validasi silang 10 kali lipat.

d) Melakukan seleksi fitur untuk mendapatkan fitur mana yang berpengaruh dan tidak berpengaruh. Fitur yang telah diseleksi akan diuji ulang pada dataset dan metode yang dipilih secara acak untuk menguji validitasnya.

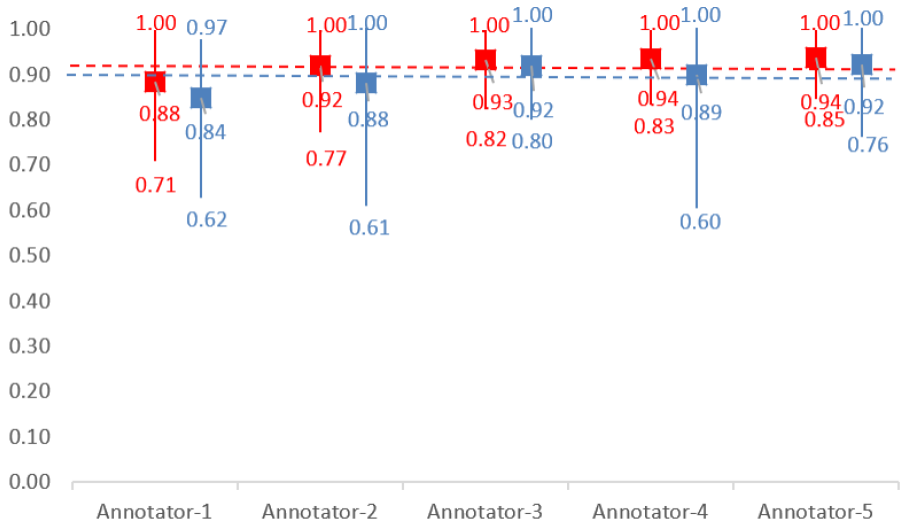
(Halaman ini sengaja dikosongkan)

BAB 4
HASIL DAN PEMBAHASAN

Pada bab ini dijelaskan tentang hasil pengujian terhadap penelitian yang dilakukan. Selanjutnya dilakukan analisa terhadap hasil pengujian untuk mendapatkan informasi yang akan menjadi kesimpulan penelitian.

4.1. Keandalan antar Anotator

Keandalan setiap anotator dihitung untuk menghasilkan anotasi standar pada dataset, dengan cara mengukur *pairwise agreement* antar anotator.



Gambar 4.1 Reliabilitas antar Anotator pada Pernyataan Non-Kebuthan

Mengukur reliabilitas antar anotator pada pernyataan non-kebutuhan yang ditunjukkan oleh diagram merah pada Gambar 4.1 Reliabilitas antar Anotator pada Pernyataan Non-Kebuthan, rata-rata reliabilitas antar anotator adalah 0,87. Hal tersebut menunjukkan bahwa lima anotator memiliki tingkat keandalan yang sangat baik.

4.2. Lingkungan Pengujian

Perangkat lunak yang digunakan adalah WEKA 3.8.4 dan sistem operasi Windows 10 64-bit.

4.3. Proses Dataset Awal

Dari dataset yang telah dikumpulkan langkah selanjutnya melabeli masing-masing kebutuhan oleh ahli. Penelitian ini menggunakan tiga orang ahli. Masing-masing ahli akan memberikan label berupa nilai *boolean* 1 (*true*) atau 0 (*false*). Di mana 1 menandakan kebutuhan tersebut derau dan 0 bukan derau. Tabel 4.1 merupakan contoh dataset asli yang sudah dilabeli oleh ahli. Data yang digunakan dalam pengujian adalah data derau mayoritas.

Tabel 4.1 Contoh Dataset Asli

ID	Pernyataan Kebutuhan	Ahli-1	Ahli-2	Ahli-3	Mayoritas
1	submit jobs with the associated deadline, cost, and execution time	0	0	0	0
2	query the cluster to establish the current cost per unit time for submitting new jobs	0	0	0	0
3	monitor the status of submitted jobs	0	0	0	0
4	cancel jobs submitted by him	0	0	0	0
5	check his credit balance	1	1	1	1
6	check his usage history	0	0	0	0
7	check the status of each node of the cluster	0	0	0	0
8	check the usage pattern history of the cluster	0	0	0	0
9	check the status of all submitted jobs	0	0	0	0
10	check the load on each node of the cluster	0	0	0	0
11	alter the cost structure of the cluster	0	0	0	0
12	alter the scheduling policy of the cluster	0	0	0	0
13	cancel, suspend, and resume any job	0	0	0	0

4.4. Hasil Pengujian

Detail hasil pelatihan dapat dilihat pada Lampiran A. Detail hasil pengujian dapat dilihat pada Lampiran B.

Tabel 4.2 Akurasi model klasifikasi dalam proses pelatihan

Dataset	SVM	Naive Bayes	Random Forest	kNN	Decision Tree
DA-1	0.880	0.712	0.869	0.820	0.847
DA-2	0.880	0.719	0.872	0.815	0.861
DA-3	0.877	0.700	0.865	0.813	0.860
DA-4	0.882	0.707	0.863	0.812	0.855
DA-5	0.880	0.741	0.875	0.834	0.868
DA-6	0.916	0.794	0.904	0.856	0.902
DA-7	0.884	0.731	0.865	0.815	0.865
DA-8	0.884	0.693	0.872	0.822	0.846
DA-9	0.878	0.750	0.863	0.813	0.855
DA-10	0.883	0.705	0.870	0.818	0.856
DA-11	0.881	0.705	0.861	0.827	0.849
DA-12	0.870	0.690	0.851	0.806	0.836
DA-13	0.883	0.713	0.865	0.825	0.865

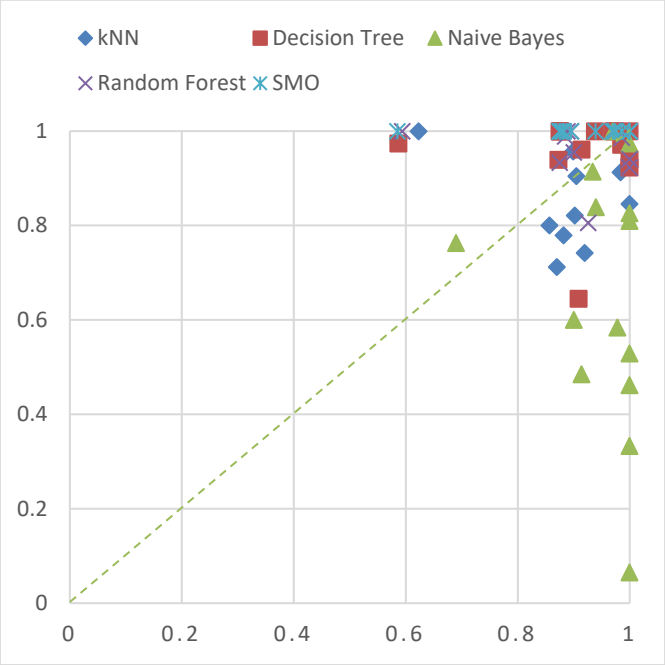
DA-14	0.883	0.691	0.869	0.839	0.867
μ	**0.883**	0.718	0.869	0.823	0.860

Tabel 4.3 Akurasi model klasifikasi dalam proses pengujian

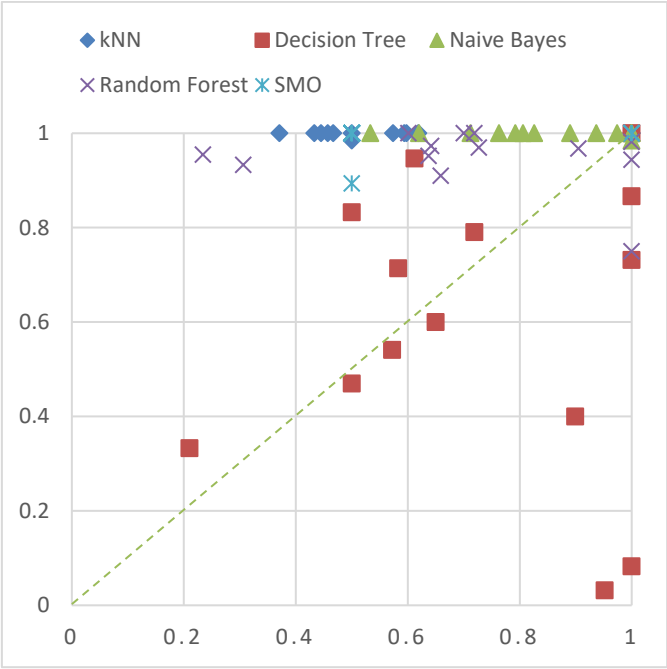
Dataset	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
DA-1	1,000	0,462	0,923	0,846	0,923
DA-2	1,000	0,529	0,941	0,941	0,941
DA-3	0,974	0,974	0,974	0,974	0,974
DA-4	1,000	0,333	1,000	1,000	1,000
DA-5	0,939	0,121	0,758	0,697	0,606
DA-6	0,585	0,662	0,600	0,646	0,585
DA-7	0,877	0,868	0,877	0,868	0,877
DA-8	0,875	0,813	0,891	0,766	0,875
DA-9	0,970	0,970	0,970	0,970	0,970
DA-10	0,882	0,588	0,824	0,706	0,941
DA-11	0,895	0,616	0,895	0,709	0,884
DA-12	0,986	0,829	0,971	0,900	0,957
DA-13	0,875	0,833	0,875	0,833	0,875
DA-14	0,880	0,507	0,867	0,653	0,827
μ	**0,910**	0,650	0,883	0,822	0,874
σ^2	0,012	0,062	**0,011**	0,016	0,016
σ_X	0,107	0,248	**0,105**	0,125	0,128

Tabel 4.3 Akurasi model klasifikasi dalam proses pengujian menunjukkan kinerja masing-masing metode dalam membangun model klasifikasi untuk pernyataan non-kebutuhan dan menguji setiap dataset. Dari proses pelatihan dan pengujian, dapat dilihat bahwa *SVM* dan *Random Forest* hampir sebanding dalam akurasi (dengan varians 0,012 dan 0,011). Akurasi *kNN* dan *Decision Tree* berada di bawahnya dengan varians 0,016. Sedangkan *Naive Bayes* memiliki nilai varians tertnggi, yang berarti nilai akurasinya adalah terendah daripada empat metode lainnya. Hasil menunjukkan bahwa model pelatihan yang dihasilkan oleh *SVM* memiliki nilai akurasi rata-rata lebih tinggi (0,883) daripada empat metode lainnya. Dari model klasifikasi yang diproduksi oleh *SVM* juga memiliki nilai akurasi rata-rata tertinggi (0,910). Hasil ini sesuai dengan penelitian oleh (Liu *et al.*, 2010; Amancio *et al.*, 2014), di mana dalam hal akurasi, *SVM* dapat menghasilkan hasil klasifikasi yang lebih baik daripada metode klasifikasi lainnya.

Tabel 4.3 Akurasi model klasifikasi dalam proses pengujian juga menunjukkan informasi statistik dari hasil klasifikasi masing-masing metode. Hal tersebut mengkonfirmasi fakta bahwa, secara umum, model klasifikasi yang diproduksi oleh *SVM* memiliki kinerja yang lebih stabil sehubungan dengan ukuran populasi dataset dan rasio antara pernyataan kebutuhan dan non-kebutuhan.



Gambar 4.2 Plot *Precision-Recall* dari lima metode klasifikasi



Gambar 4.3 Plot *ROC-Recall* dari lima metode

Berdasarkan plot *precision-recall* (Gambar 4.2 Plot *Precision-Recall* dari lima metode klasifikasi dan plot *ROC-recall* (Gambar 4.3 Plot *ROC-Recall* dari lima metode dari masing-masing metode dalam ruang dua dimensi, dapat digukur kemampuan klasifikasi untuk membedakan dua hasil yang mungkin. Model

klasifikasi yang dapat menghasilkan jumlah hasil tertinggi yang terletak di atas garis diagonal dianggap sebagai model terbaik. Dua plot grafis menunjukkan bahwa model klasifikasi yang diproduksi oleh *SVM* memiliki potensi kegunaan yang lebih tinggi daripada yang lain. Selain itu, untuk varian hasil klasifikasi di antara masalah domain, penelitian ini juga menunjukkan bahwa ada varians yang tidak signifikan untuk semua metode.

4.5. Model Klasifikasi *SVM*

Representasi *SVM* secara sederhana adalah:

$\min_{f\xi_i} \|f\|_K^2 + C\sum_{i=l}^{l}\xi_i \qquad y_i f(x_i) \geq 1 - \xi_i$, untuk semua $i\ \xi_i \geq 0$

Variabel $\xi_i$disebut sebagai variabel *slack* untuk mengukur kesalahan yang dibuat pada titik (x_i, y_i).

Fungsi kernel memiliki peran penting pada performa SVM. Hal ini didasarkan pada mereproduksi ruang *Kernel Hilbert*.

$$K(x, x') = \langle \phi(x), \phi(x') \rangle$$

Jika *K* adalah fungsi pasti positif simetris, yang memenuhi Ketentuan *Mercer*,

$$K(x, x') = \sum_{m}^{\infty} a_m \emptyset_m(x)\emptyset_m(x'),\ a_m \geq 0,$$

$$\iint K(x, x')g(x)g(x')dxdx' > 0,\ g \in L_2$$

Kernel polinomial adalah metode yang populer untuk pemodelan non-linear. Kernel kedua biasanya lebih disukai karena menghindari masalah dengan *hessian* menjadi nol.

$$K(x, x') = \langle x, x' \rangle^d$$

$$K(x, x') = (\langle x, x' \rangle + 1)^d$$

4.6. Seleksi Atribut

Seleksi atribut dilakukan untuk mengetahui atribut mana yang paling signifikan. Evaluator atribut menggunakan *InfoGainAttributeEval* dan metode pencarian dengan *Ranker* dari aplikasi weka. Tabel keseluruhan seleksi atribut dapat dilihat pada lampiran C.

Tabel 4.4 Keterangan kode judul terhadap nama atribut

Kode judul	**Nama Atribut/Fitur**	**Keterangan**
a	maxValNorm	Nilai normalisasi maksimal
b	meanValNorm	Nilai normalisasi mean
c	STDValNorm	Nilai normalisasi sandar deviasi
d	varSMNorm	Nilai normalisasi varian
e	maxVal	Nilai maksimal
f	meanVal	Nilai mean
g	STDVal	Nilai standar deviasi
h	varSM	Nilai varian
i	bad_NN	Nilai NN yang buruk
j	bad_VB	Nilai VB yang buruk
k	bad_punctuation	Nilai kata lain yang buruk

Tabel 4.5 Hasil seleksi fitur dengan perlakuan fitur bernilai 0 semua pada setiap dataset dihapus

Dataset	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**
DA-1	0.072	0.022	0.020	0.021	0.005	0.057	0.075	0.071	0.000	0.010	0.011
DA-2	0.071	0.021	0.019	0.020	0.005	0.056	0.074	0.071	0.000	0.010	0.000
DA-3	0.066	0.020	0.021	0.019	0.005	0.050	0.069	0.065	0.000	0.010	0.013
DA-4	0.071	0.019	0.017	0.018	0.005	0.054	0.075	0.072	0.000	0.010	0.012
DA-5	0.087	0.021	0.019	0.020	0.005	0.060	0.092	0.088	0.000	0.010	0.013
DA-6	0.044	0.000	0.000	0.000	0.006	0.036	0.051	0.047	0.016	0.012	0.015
DA-7	0.064	0.000	0.000	0.019	0.006	0.039	0.071	0.067	0.000	0.012	0.015
DA-8	0.071	0.000	0.019	0.016	0.005	0.050	0.076	0.072	0.000	0.011	0.000
DA-9	0.070	0.021	0.021	0.018	0.005	0.051	0.073	0.070	0.000	0.010	0.013
DA-10	0.073	0.022	0.020	0.021	0.005	0.054	0.072	0.069	0.000	0.010	0.000
DA-11	0.074	0.022	0.021	0.014	0.005	0.058	0.081	0.078	0.000	0.011	0.000
DA-12	0.069	0.020	0.018	0.018	0.005	0.055	0.072	0.069	0.000	0.010	0.000
DA-13	0.064	0.000	0.018	0.016	0.005	0.050	0.069	0.065	0.000	0.010	0.000
DA-14	0.086	0.022	0.023	0.020	0.000	0.067	0.083	0.077	0.000	0.000	0.015
μ	0.070	0.015	0.017	0.017	0.005	0.053	0.074	0.070	0.001	0.010	0.008
σ^2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
σ_X	0.010	0.010	0.007	0.005	0.001	0.008	0.009	0.009	0.004	0.003	0.007

Pada Tabel 4.4 Keterangan kode judul terhadap nama atribut menjelaskan kode judul pada Tabel 4.5 Hasil seleksi fitur dengan perlakuan fitur bernilai 0 semua pada setiap dataset dihapus. Pada Tabel 4.5 Hasil seleksi fitur dengan perlakuan fitur bernilai 0 semua pada setiap dataset dihapus dapat dilihat bahwa semua fitur dapat mempengaruhi akurasi (dengan varians paling besar 0.0001). Dari hasil seleksi atribut tersebut dapat dilihat bahwa dari sembilan belas fitur yang digunakan untuk membangun model, hanya sebelas fitur yang dianggap signifikan

mengklasifikasi pernyataan non-kebutuhan. Sebelas fitur tersebut merepresentasikan fitur semantik dan statistik.

Tabel 4.6 Koefisian Korelasi *Pearson*

	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**
a	1,000	0,628	0,716	0,751	-0,606	0,899	0,958	0,942	-0,742	-0,607	-0,061
b	0,628	1,000	0,708	0,548	-0,379	0,759	0,541	0,537	-0,437	-0,379	0,015
c	0,716	0,708	1,000	0,613	-0,451	0,855	0,630	0,629	-0,665	-0,452	-0,366
d	0,751	0,548	0,613	1,000	-0,343	0,614	0,678	0,691	-0,927	-0,344	-0,133
e	-0,606	-0,379	-0,451	-0,343	1,000	-0,690	-0,410	-0,352	0,272	1,000	-0,224
f	0,899	0,759	0,855	0,614	-0,690	1,000	0,811	0,791	-0,610	-0,691	-0,170
g	0,958	0,541	0,630	0,678	-0,410	0,811	1,000	0,997	-0,724	-0,411	-0,070
h	0,942	0,537	0,629	0,691	-0,352	0,791	0,997	1,000	-0,752	-0,353	-0,112
i	-0,742	-0,437	-0,665	-0,927	0,272	-0,610	-0,724	-0,752	1,000	0,274	0,298
j	-0,607	-0,379	-0,452	-0,344	1,000	-0,691	-0,411	-0,353	0,274	1,000	-0,225
k	-0,061	0,015	-0,366	-0,133	-0,224	-0,170	-0,070	-0,112	0,298	-0,225	1,000

Tabel 4.6 Koefisian Korelasi *Pearson* menunjukkan korelasi antar variabel setelah dilakukan seleksi fitur. Penjelasan huruf a-k pada judul atas dan samping kiri seperti yang dijelaskan pada Tabel 4.4 Keterangan kode judul terhadap nama atribut.

Tabel 4.7 Pengujian ulang dengan hanya sebelas atribut

Dataset	**Klasifikasi**	***Training***	***Retraining***	**Selisih**	***Testing***	***Retesting***	**Selisih**
DA-2	*Random Forest*	0,872	0,864	0,008	0,941	0,941	0
DA-5	*Random Forest*	0,875	0,876	0,001	0,758	0,727	0,031
DA-2	*SVM*	0,880	0,880	0	1,000	1,000	0
DA-2	*Naive Bayes*	0,719	0,686	0,033	0,529	0,529	0
DA-5	*SVM*	0,880	0,880	0	0,939	0,939	0
DA-5	*kNN*	0,834	0,836	0,002	0,697	0,697	0

Tabel 4.7 Pengujian ulang dengan hanya sebelas atribut menunjukkan bahwa setelah penggunaan sebelas atribut, kemungkinan menghasilkan nilai uji ulang atau nilai latih ulang yang berbeda dari nilai awal adalah cukup kecil (selisih ≤ 0,033).

(Halaman ini sengaja dikosongkan)

BAB 5
PENUTUP

Pada bab ini dijelaskan tentang kesimpulan dan saran dari serangkaian pengujian pada bab sebelumnya.

5.1. Kesimpulan

Untuk mendeteksi pernyataan non-kebutuhan pada SKPL dapat dilakukan dengan cara melakukan klasifikasi terhadap pernyataan dalam SKPL. Dari proses klasifikasi dengan lima metode yang dilakukan, yaitu *SVM, Naive Bayes, Random Forest, kNN,* dan *Decision Tree*, dihasilkan metode *SVM* adalah metode terbaik untuk mendeteksi pernyataan non-kebutuhan. Hal tersebut ditunjukkan oleh nilai rata-rata akurasi sebesar 0,96. Selain itu dari plot *precision-recall* dan *ROC-recall*, titik yang dihasilkan metode *SVM* berada di atas garis diagonal dibandingkan metode lain dengan sumbu x dan y mendekati nilai 1.

Fitur-fitur yang mempengaruhi hasil deteksi derau yaitu, nilai normalisasi maksimal, nilai normalisasi mean, nilai normalisasi sandar deviasi, nilai normalisasi varian, nilai maksimal, nilai mean, Nilai standar deviasi, nilai varian, nilai NN yang buruk, nilai VB yang buruk, dan nilai kata lain yang buruk.

5.2. Saran

Saran yang dapat diberikan untuk pengembangan lebih lanjut penelitian ini adalah penggunaan metode lain untuk memperoleh nilai yang lebih baik. Selain itu diharapkan adanya pengembangan model klasifikasi pada metode *Naive Bayes* agar dapat mendeteksi pernyataan non-kebutuhan dengan lebih baik.

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Ali, J. *et al.* (2012) 'Random Forests and Decision Trees', *International Journal of Computer Science Issues*, 9(5), pp. 272–278.

Amancio, D. R. *et al.* (2014) 'A Systematic Comparison of Supervised Classifiers', 9(4). doi: 10.1371/journal.pone.0094137.

Basuki, S. (1991) *Pengantar ilmu perpustakaan*. Gramedia Pustaka Utama.

Benesty, J., Chen, J. and Huang, Y. (2008) 'On the importance of the pearson correlation coefficient in noise reduction', *IEEE Transactions on Audio, Speech and Language Processing*, 16(4), pp. 757–765. doi: 10.1109/TASL.2008.919072.

Cai, X. *et al.* (2011) 'Simultaneous Clustering and Noise Detection for Theme-based Summarization', *Proceedings of 5th International Joint Conference on Natural Language Processing*, pp. 491–499. Available at: http://aclweb.org/anthology/I11-1055.

Cohen, J. (1960) 'A Coefficient of Agreement for Nominal Scales', *Educational and Psychological Measurement*, 20(1), pp. 37–46. doi: 10.1177/001316446002000104.

Colditz, R. R. (2015) 'An evaluation of different training sample allocation schemes for discrete and continuous land cover classification using decision tree-based algorithms', *Remote Sensing*, 7(8), pp. 9655–9681. doi: 10.3390/rs70809655.

Garcia, E. (2015) 'Cosine Similarity Tutorial', *Information Retrieval iItelligence*, (April), pp. 4–10. Available at: https://www.ijcaonline.org/research/volume134/number7/irani-2016-ijca-907841.pdf%0Ahttp://www.ijcaonline.org/research/volume134/number7/irani-2016-ijca-907841.pdf.

Hamakonda, T. P. (1978) *Pengantar klasifikasi persepuluhan Dewey*. BPK Gunung Mulia.

Immitzer, M., Atzberger, C. and Koukal, T. (2012) 'Tree species classification with Random forest using very high spatial resolution 8-band worldView-2 satellite data', *Remote Sensing*, 4(9), pp. 2661–2693. doi: 10.3390/rs4092661.

Liu, Z. *et al.* (2010) 'Study on SVM Compared with the other Text Classification

Methods', in *2010 Second International Workshop on Education Technology and Computer Science*. IEEE, pp. 219–222. doi: 10.1109/ETCS.2010.248.

Manek, P. G. and Siahaan, D. (2019) 'Noise Detection in Software Requirements Specification Document Using Spectral Clustering', *JUTI: Jurnal Ilmiah Teknologi Informasi*, 17(1), p. 30. doi: 10.12962/j24068535.v17i1.a771.

Meyer, B. (1985) 'On Formalism in Specification', *IEEE Software*, 2(1), pp. 6–26. doi: 10.1109/MS.1985.229776.

Montes-y-Gómez, M., Gelbukh, A. and López-López, A. (2002) 'Detecting deviations in text collections: An approach using conceptual graphs', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 176–184. doi: 10.1007/3-540-46016-0_19.

Rossi, A. (1999) 'Incentives in managerial compensation: a survey of experimental research', *ROCK Working Papers*.

Siahaan, D. (2012) *Analisa Kebutuhan dalam Rekayasa Perangkat Lunak*.

Lampiran A. Hasil Data Latih

Tabel hasil pelatihan dataset DA-2 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.0315	71.1811	86.9291	82.0472	84.7244
Incorectly Classified Instances (%)	11.9685	28.8189	13.0709	17.9528	15.2756
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2456	0.0336	0.1282	0.0259
Mean Absolute Error	0.1197	0.2873	0.1857	0.1806	0.2
Root Mean Squarred Error	0.346	0.4655	0.3187	0.423	0.3645
Relative Absolute Error (%)	56.5254	135.6937	87.7036	85.3159	94.4493
Root Relative Squared Error (%)	106.5747	143.3869	98.189	130.3001	112.2761

Tabel hasil pelatihan dataset DA-1, DA-3 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.2726	70.523	87.0048	81.775	85.5784
Incorectly Classified Instances (%)	11.7274	29.477	12.9952	18.225	14.4216
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2077	0.0321	0.1146	0.1365
Mean Absolute Error	0.1173	0.2893	0.1846	0.1834	0.186
Root Mean Squarred Error	0.3425	0.4658	0.3217	0.4262	0.3558
Relative Absolute Error (%)	56.3597	139.0359	88.7057	88.122	89.3923
Root Relative Squared Error (%)	106.4279	144.7589	99.9654	132.4429	110.5719

Tabel hasil pelatihan dataset DA-1, DA-2, DA-4 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.0783	70.4626	86.121	82.7402	84.8754
Incorectly Classified Instances (%)	11.9217	29.5374	13.879	17.2598	15.1246
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2187	0.0426	0.1834	0.085
Mean Absolute Error	0.1192	0.2895	0.1875	0.1739	0.1968
Root Mean Squarred Error	0.3453	0.4662	0.3196	0.4146	0.3549
Relative Absolute Error (%)	56.4572	137.107	88.7895	82.3473	93.196
Root Relative Squared Error (%)	106.5444	143.8601	98.6189	127.9461	109.525

Tabel hasil pelatihan dataset DA-1 sampai DA-3, dan DA-5 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87.0242	69.0311	85.1211	80.6228	83.564
Incorectly Classified Instances (%)	12.9758	30.9689	14.8789	19.3772	16.436
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2284	0.0503	0.1118	0.0919
Mean Absolute Error	0.1298	0.3159	0.2006	0.1949	0.2186
Root Mean Squarred Error	0.3602	0.4844	0.3315	0.4394	0.3726
Relative Absolute Error (%)	57.1861	139.2248	88.3869	85.9148	96.3571
Root Relative Squared Error (%)	107.1879	144.1386	98.6523	130.7359	110.8801

Tabel hasil pelatihan dataset DA-1 sampai DA-4, dan DA-6 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.3013	71.3141	86.5385	82.5321	86.5385
Incorectly Classified Instances (%)	11.6987	28.6859	13.4615	17.4679	13.4615
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2244	0.0406	0.1495	0.073
Mean Absolute Error	0.117	0.2912	0.184	0.1758	0.1916
Root Mean Squarred Error	0.342	0.4618	0.3189	0.4172	0.3396
Relative Absolute Error (%)	56.3388	140.2194	88.6293	84.6782	92.2624
Root Relative Squared Error (%)	106.4121	143.6875	99.2219	129.8	105.6645

Tabel hasil pelatihan dataset DA-1 sampai DA-5, dan DA-7 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.3072	69.1099	86.911	83.9442	86.7365
Incorectly Classified Instances (%)	11.6928	30.8901	13.089	16.0558	13.2635
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2582	0.089	0.157	0.1625
Mean Absolute Error	0.1169	0.3101	0.1791	0.1619	0.1823
Root Mean Squarred Error	0.3419	0.5044	0.3122	0.3999	0.3356
Relative Absolute Error (%)	56.3084	149.3488	86.2275	77.9505	87.809
Root Relative Squared Error (%)	106.4067	156.9596	97.1553	124.4485	104.4436

Tabel hasil pelatihan dataset DA-1 sampai DA-6, dan DA-8 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87.9556	71.9493	87.1632	81.458	86.0539
Incorectly Classified Instances (%)	12.0444	28.0507	12.8368	18.542	13.9461
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2553	0.0731	0.1199	0.1356
Mean Absolute Error	0.1204	0.2945	0.1886	0.1865	0.19
Root Mean Squarred Error	0.3471	0.4694	0.3178	0.4299	0.3473
Relative Absolute Error (%)	56.5736	138.3172	88.5749	87.6121	89.2345
Root Relative Squared Error (%)	106.6195	144.2088	97.6198	132.0575	106.7028

Tabel hasil pelatihan dataset DA-1 sampai DA-7, dan DA-9 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87.6847	69.9507	86.5353	81.2808	86.0427
Incorectly Classified Instances (%)	12.3153	30.0493	13.4647	18.7192	13.9573
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2155	0.0662	0.1024	0.0869
Mean Absolute Error	0.1232	0.3029	0.1945	0.1883	0.197
Root Mean Squarred Error	0.3509	0.4688	0.3278	0.4319	0.3462
Relative Absolute Error (%)	56.7476	139.5708	89.6083	86.7805	90.7694
Root Relative Squared Error (%)	106.7835	142.6463	99.7421	131.4131	105.3382

Tabel hasil pelatihan dataset DA-1 sampai DA-8, dan DA-10 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.162	70.7165	86.2928	81.1526	85.514
Incorectly Classified Instances (%)	11.838	29.2835	13.7072	18.8474	14.486
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2388	0.0024	0.1022	0.0809
Mean Absolute Error	0.1184	0.2939	0.1865	0.1895	0.1908
Root Mean Squarred Error	0.3441	0.4693	0.3197	0.4334	0.3565
Relative Absolute Error (%)	56.4392	140.126	88.9141	90.3695	90.9548
Root Relative Squared Error (%)	106.495	145.2672	98.95	134.1431	110.3501

Tabel hasil pelatihan dataset DA-1 sampai DA-9, dan DA-11 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87.9675	74.1463	87.4797	83.4146	86.8293
Incorectly Classified Instances (%)	12.0325	25.8537	12.5203	16.5854	13.1707
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2741	0.1719	0.1576	0.2405
Mean Absolute Error	0.1203	0.2715	0.1761	0.1671	0.1754
Root Mean Squarred Error	0.3469	0.4461	0.3106	0.4065	0.336
Relative Absolute Error (%)	56.5599	127.6353	82.7991	78.5265	82.4319
Root Relative Squared Error (%)	106.613	137.1059	95.4583	124.9441	103.2741

Tabel hasil pelatihan dataset DA-1 sampai DA-10, dan DA-12 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87.9675	74.1463	87.4797	83.4146	86.8293
Incorectly Classified Instances (%)	12.0325	25.8537	12.5203	16.5854	13.1707
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2741	0.1719	0.1576	0.2405
Mean Absolute Error	0.1203	0.2715	0.1761	0.1671	0.1754
Root Mean Squarred Error	0.3469	0.4461	0.3106	0.4065	0.336
Relative Absolute Error (%)	56.5599	127.6353	82.7991	78.5265	82.4319
Root Relative Squared Error (%)	106.613	137.1059	95.4583	124.9441	103.2741

Tabel hasil pelatihan dataset DA-1 sampai DA-11, dan DA-13 sampai dengan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	91.5952	79.4168	90.3945	85.5918	90.223
Incorectly Classified Instances (%)	8.4048	20.5832	9.6055	14.4082	9.777
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.1892	0.037	-0.0314	0.0852
Mean Absolute Error	0.084	0.2318	0.1397	0.1454	0.1456
Root Mean Squarred Error	0.2899	0.3867	0.2751	0.3789	0.3002
Relative Absolute Error (%)	54.1243	149.3013	89.9347	93.6527	93.7851
Root Relative Squared Error (%)	104.4816	139.3562	99.1463	136.5403	108.1991

Tabel hasil pelatihan dataset DA-1 sampai DA-12, dan DA-14

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.3764	73.0627	86.5314	81.5498	86.5314
Incorectly Classified Instances (%)	11.6236	26.9373	13.4686	18.4502	13.4686
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.2175	0.0862	0.102	0.0497
Mean Absolute Error	0.1162	0.295	0.1843	0.1858	0.1926
Root Mean Squarred Error	0.3409	0.4589	0.32	0.4287	0.3387
Relative Absolute Error (%)	56.2443	142.733	89.1874	89.9	93.1798
Root Relative Squared Error (%)	106.3645	143.1719	99.8215	133.7343	105.6809

Tabel hasil pelatihan dataset DA-1 sampai DA-13

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88.3562	69.3493	87.1575	82.1918	84.589
Incorectly Classified Instances (%)	11.6438	30.6507	12.8425	17.8082	15.411
Time Taken	0	0	0	0	0
Kappa Statistic	0	0.1944	0.0745	0.1	0.0111
Mean Absolute Error	0.1164	0.3002	0.1833	0.1793	0.2001
Root Mean Squarred Error	0.3412	0.4726	0.3163	0.4212	0.3708
Relative Absolute Error (%)	56.2819	145.12	88.5777	86.668	96.7112
Root Relative Squared Error (%)	106.3791	147.3192	98.6025	131.3101	115.6049

Lampiran B. Hasil Uji Coba

Tabel hasil pengujian DA-1 sebagai data uji dan DA-2 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	100	46,1538	92,3077	84,6154	92,3077
Incorectly Classified Instances (%)	0	53,8462	7,6923	15,3846	7,6923
Time Taken	0	0,01	0	0	0
Kappa Statistic	1	0	0	0	0
Mean Absolute Error	0	0,6051	0,1469	0,1549	0,1195
Root Mean Squarred Error	0	0,6897	0,2235	0,3916	0,281
Relative Absolute Error (%)	0	500,5558	121,5455	128,1718	98,8386
Root Relative Squared Error (%)	0	570,5808	184,8842	323,9759	232,4466

Tabel hasil pengujian DA-2 sebagai data uji dan DA-1, DA-3 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88,2353	58,8235	82,3529	70,5882	94,1176
Incorectly Classified Instances (%)	11,7647	41,1765	17,6471	29,4118	5,8824
Time Taken	0	0	0	0	0
Kappa Statistic	0	0,048	-0,0851	-0,1644	0,6383
Mean Absolute Error	0,1176	0,4787	0,2606	0,2947	0,1307
Root Mean Squarred Error	0,343	0,6072	0,3556	0,5415	0,2574
Relative Absolute Error (%)	56,4926	229,8424	125,1312	141,5217	62,7572
Root Relative Squared Error (%)	106,4578	188,462	110,3752	168,0591	79,8849

Tabel hasil pengujian DA-3 sebagai data uji dan DA-1, DA-2, DA-4 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	89,5349	61,6279	89,5349	70,9302	88,3721
Incorectly Classified Instances (%)	10,4651	38,3721	10,4651	29,0698	11,6279
Time Taken	0	0	0	0	0
Kappa Statistic	0	0,1878	0	-0,0761	0,228
Mean Absolute Error	0,1047	0,3835	0,2083	0,2914	0,2212
Root Mean Squarred Error	0,3235	0,5148	0,3092	0,5382	0,3418
Relative Absolute Error (%)	52,3299	191,7885	104,1365	145,732	110,5886
Root Relative Squared Error (%)	105,5401	167,9489	100,887	175,589	111,508

Tabel hasil pengujian DA-4 sebagai data uji dan DA-1 sampai dengan DA-3 dan DA-5 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	98,5714	82,8571	97,1429	90	95,7143
Incorectly Classified Instances (%)	1,4286	17,1429	2,8571	10	4,2857
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,1195	-0,0145	-0,0251	-0,0194
Mean Absolute Error	0,0143	0,1672	0,0836	0,1014	0,0805
Root Mean Squarred Error	0,1195	0,3473	0,1835	0,3157	0,192
Relative Absolute Error (%)	10,0905	118,1091	59,0292	71,6075	56,8397
Root Relative Squared Error (%)	71,7989	208,6326	110,2458	189,637	115,3082

Tabel hasil pengujian DA-5 sebagai data uji dan DA-1 sampai dengan DA-4 dan DA-6 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87,5	83,3333	87,5	83,3333	87,5
Incorectly Classified Instances (%)	12,5	16,6667	12,5	16,6667	12,5
Time Taken	0	0	0	0	0
Kappa Statistic	0	0,5152	0	0,2381	0
Mean Absolute Error	0,125	0,193	0,1625	0,1677	0,1418
Root Mean Squarred Error	0,3536	0,3329	0,2994	0,4076	0,2798
Relative Absolute Error (%)	58,5047	90,3215	76,0561	78,5047	66,3906
Root Relative Squared Error (%)	106,882	100,6519	90,499	123,2203	84,5936

Tabel hasil pengujian DA-6 sebagai data uji dan DA-1 sampai dengan DA-5 dan DA-7 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	88	50,6667	86,6667	65,3333	82,6667
Incorectly Classified Instances (%)	12	49,3333	13,3333	34,6667	17,3333
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,0609	0,2188	-0,0417	-0,0797
Mean Absolute Error	0,12	0,4791	0,2549	0,3472	0,2301
Root Mean Squarred Error	0,3464	0,6421	0,3525	0,5878	0,3989
Relative Absolute Error (%)	57,176	228,2667	121,4672	165,4292	109,6402
Root Relative Squared Error (%)	106,5988	197,5964	108,4634	180,8686	122,7557

Tabel hasil pengujian DA-7 sebagai data uji dan DA-1 sampai dengan DA-6 dan DA-8 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	100	52,9412	94,1176	94,1176	94,1176
Incorectly Classified Instances (%)	0	47,0588	5,8824	5,8824	5,8824
Time Taken	0	0	0	0	0
Kappa Statistic	1	0	0	0	0
Mean Absolute Error	0	0,3679	0,1118	0,0602	0,0932
Root Mean Squarred Error	0	0,533	0,1801	0,2422	0,1759
Relative Absolute Error (%)	0	302,4205	91,8793	49,5034	76,6048
Root Relative Squared Error (%)	0	438,202	148,0814	199,0722	144,5632

Tabel hasil pengujian DA-8 sebagai data uji dan DA-1 sampai dengan DA-7 dan DA-9 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	97,4359	97,4359	97,4359	97,4359	97,4359
Incorectly Classified Instances (%)	2,5641	2,5641	2,5641	2,5641	2,5641
Time Taken	0	0	0	0	0
Kappa Statistic	0	0,6549	0	0	0
Mean Absolute Error	0,0256	0,0352	0,0521	0,0272	0,0632
Root Mean Squarred Error	0,1601	0,1543	0,1266	0,1599	0,1302
Relative Absolute Error (%)	17,8498	24,4898	36,2352	18,9308	43,9701
Root Relative Squared Error (%)	85,919	82,7914	67,9521	85,7827	69,8519

Tabel hasil pengujian DA-9 sebagai data uji dan DA-1 sampai dengan DA-8 dan DA-10 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	100	33,3333	100	100	100
Incorectly Classified Instances (%)	0	66,6667	0	0	0
Time Taken	0	0	0	0	0
Kappa Statistic	1	0	1	1	1
Mean Absolute Error	0	0,5669	0,1317	0,0016	0,0933
Root Mean Squarred Error	0	0,7008	0,1634	0,0016	0,1191
Relative Absolute Error (%)	0	474,1617	110,1212	1,2987	78,0472
Root Relative Squared Error (%)	0	586,088	136,6203	1,2987	99,5814

Tabel hasil pengujian DA-10 sebagai data uji dan DA-1 sampai dengan DA-9 dan DA-11 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	93,9394	12,1212	75,7576	69,697	60,6061
Incorectly Classified Instances (%)	6,0606	87,8788	24,2424	30,303	39,3939
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,0083	-0,1	-0,1074	-0,1143
Mean Absolute Error	0,0606	0,8446	0,3948	0,3037	0,448
Root Mean Squarred Error	0,2462	0,889	0,4428	0,5496	0,6113
Relative Absolute Error (%)	36,1983	504,4299	235,8319	181,3728	267,5594
Root Relative Squared Error (%)	99,9655	361,0012	179,8116	223,1681	248,2159

Tabel hasil pengujian DA-11 sebagai data uji dan DA-1 sampai dengan DA-10 dan DA-12 sampai dengan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	58,4615	66,1538	60	64,6154	58,4615
Incorectly Classified Instances (%)	41,5385	33,8462	40	35,3846	41,5385
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,2878	0,043	0,169	0,0124
Mean Absolute Error	0,4154	0,3709	0,4051	0,3543	0,4074
Root Mean Squarred Error	0,6445	0,4747	0,5756	0,5938	0,5764
Relative Absolute Error (%)	96,6351	86,2773	94,2371	82,435	94,7868
Root Relative Squared Error (%)	108,6798	80,0511	97,0592	100,1358	97,1992

Tabel hasil pengujian DA-12 sebagai data uji dan DA-1 sampai dengan DA-11, DA-13 dan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87,7358	86,7925	87,7358	86,7925	87,7358
Incorectly Classified Instances (%)	12,2642	13,2075	12,2642	13,2075	12,2642
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,4244	0,104	0,2343	0
Mean Absolute Error	0,1226	0,1568	0,1623	0,1334	0,2104
Root Mean Squarred Error	0,3502	0,289	0,316	0,3628	0,3281
Relative Absolute Error (%)	58,0052	74,1475	76,7454	63,107	99,4964
Root Relative Squared Error (%)	106,7483	88,107	96,3259	110,5756	100,0075

Tabel hasil pengujian DA-13 sebagai data uji dan DA-1 sampai dengan DA-12 dan DA-14 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	87,5	81,25	89,0625	76,5625	87,5
Incorectly Classified Instances (%)	12,5	18,75	10,9375	23,4375	12,5
Time Taken	0	0	0	0,01	0
Kappa Statistic	0	0,3514	0,2	0,1549	0
Mean Absolute Error	0,125	0,2069	0,1866	0,2353	0,1892
Root Mean Squarred Error	0,3536	0,3372	0,3274	0,4833	0,3314
Relative Absolute Error (%)	58,6	97,0175	87,4605	110,3	88,6909
Root Relative Squared Error (%)	106,8788	101,936	98,9697	146,1008	100,1922

Tabel hasil pengujian DA-14 sebagai data uji dan DA-1 sampai dengan DA-13 sebagai data latih.

Klasifikasi	**SVM**	**Naive Bayes**	**Random Forest**	**kNN**	**Decision Tree**
Algoritma	functions.SMO	bayes.Naive-Bayes	trees.Random-Forest	lazy.IBk	trees.J48
Correctly Classified Instances (%)	96,9697	96,9697	96,9697	96,9697	96,9697
Incorectly Classified Instances (%)	3,0303	3,0303	3,0303	3,0303	3,0303
Time Taken	0	0	0	0	0
Kappa Statistic	0	0	0	0	0
Mean Absolute Error	0,0303	0,0303	0,0436	0,0318	0,069
Root Mean Squarred Error	0,1741	0,1741	0,1753	0,1738	0,1718
Relative Absolute Error (%)	20,7534	20,7534	29,885	21,7962	47,2836
Root Relative Squared Error (%)	89,2879	89,2879	89,893	89,1469	88,1033

Lampiran C. Hasil Seleksi Fitur

Hasil seleksi fitur dari seluruh dataset

Dataset	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
1	0	0.07171	0.02208	0.02039	0.02099	0	0.00484	0.05668	0.07478	0.07106	0	0	0.0097	0	0	0	0.01137	0	0
2	0	0.07148	0.02077	0.0194	0.02006	0	0.00485	0.05619	0.07447	0.0707	0	0	0.00973	0	0	0	0	0	0
3	0	0.06628	0.02043	0.02055	0.01933	0	0.00498	0.05005	0.06924	0.06548	0	0	0.00998	0	0	0	0.01349	0	0
4	0	0.07146	0.0193	0.01747	0.01832	0	0.00481	0.05443	0.0753	0.07166	0	0	0.00964	0	0	0	0.01169	0	0
5	0	0.08665	0.02054	0.01882	0.01983	0	0.00498	0.06005	0.09216	0.0875	0	0	0.00999	0	0	0	0.01267	0	0
6	0	0.04404	0	0	0	0	0.00615	0.03592	0.05118	0.047	0	0.01643	0.01235	0	0	0	0.01471	0	0
7	0	0.06405	0	0	0.01887	0	0.00575	0.03938	0.07057	0.06739	0	0	0.01153	0	0	0	0.01484	0	0
8	0	0.07118	0	0.0187	0.01605	0	0.00533	0.04999	0.07563	0.07181	0	0	0.01069	0	0	0	0	0	0
9	0	0.06999	0.02055	0.02067	0.01796	0	0.00495	0.0514	0.07341	0.06978	0	0	0.00993	0	0	0	0.01264	0	0
10	0	0.07309	0.02219	0.02018	0.02111	0	0.00491	0.05376	0.07243	0.06888	0	0	0.00986	0	0	0	0	0	0
11	0	0.07369	0.02156	0.02136	0.01395	0	0.00548	0.05791	0.08105	0.07811	0	0	0.01099	0	0	0	0	0	0
12	0	0.06944	0.01953	0.01795	0.0184	0	0.00511	0.05456	0.07172	0.06852	0	0	0.01025	0	0	0	0	0	0
13	0	0.06369	0	0.01774	0.01615	0	0.00497	0.04963	0.06853	0.06518	0	0	0.00998	0	0	0	0	0	0
14	0	0.0864	0.0222	0.023	0.0196	0	0	0.0672	0.0825	0.0766	0	0	0	0	0	0	0.0147	0	0

Keterangan judul

Judul	Nama Atribut/Fitur	Keterangan
a	minValNorm	Nilai normalisasi minimal
b	maxValNorm	Nilai normalisasi maksimal
c	meanValNorm	Nilai normalisasi mean
d	STDValNorm	Nilai normalisasi standar deviasi
e	varSMNorm	Nilai normalisasi varian kesamaan
f	minVal	Nilai minimal
g	maxVal	Nilai maksimal
h	meanVal	Nilai mean
i	STDVal	Nilai standar deviasi
j	varSM	Nilai varian kesamaan
k	bad_MD	Nilai MD yang buruk
l	bad_NN	Nilai NN yang buruk
m	bad_VB	Nilai VB yang buruk
n	bad_RB	Nilai RB yang buruk
o	bad_JJ	Nilai JJ yang buruk
p	bad_DT	Nilai DT yang buruk
q	bad_punctuation	Nilai kata lain yang buruk
r	num_sentence	Jumlah kalimat
s	num_words	Jumlah kata