ANALISIS PERBANDINGAN METODE *MACHINE LEARNING: RANDOM FOREST* DAN *SUPPORT VECTOR MACHINE* UNTUK DETEKSI KANKER PARU-PARU

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Matematika (S1) dan
mencapai gelar Sarjana Sains

Oleh

Maria Artati Eka Setyorini
NIM 161810101032

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2020

ANALISIS PERBANDINGAN METODE *MACHINE LEARNING: RANDOM FOREST* DAN *SUPPORT VECTOR MACHINE* UNTUK DETEKSI KANKER PARU-PARU

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Matematika (S1) dan
mencapai gelar Sarjana Sains

Oleh

Maria Artati Eka Setyorini
NIM 161810101032

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2020

PERSEMBAHAN

Puji syukur kepada Tuhan Yang Maha Esa sehingga terselesaikannya skripsi ini dan saya persembahkan untuk:

1. Keluarga tercinta, Ayah Y. Leonardus Joni, Ibu Ch. Erwantari H., dan adik Y. Artanto D.N.;
2. Seluruh jajaran guru dan dosen dari TKK Siswa Rini Jember, SDK Maria Fatima Jember, SMPK Maria Fatima, SMAK Santo Paulus Jember dan Jurusan Matematika FMIPA Universitas Jember;
3. Almamater Jurusan Matematika FMIPA Universitas Jember, SMAK Santo Paulus Jember, SMPK Maria Fatima Jember, SDK Maria Fatima Jember, dan TKK Siswa Rini Jember;
4. Teman-teman angkatan MISDIRECTON'16.

MOTTO

Sains dibentuk oleh pengetahuan.

Kebijaksanaan dibentuk oleh kehidupan. –Immanuel Kant[*)]

Hiduplah seakan-akan kau akan mati besok.

Belajarlah seakan-akan kau akan hidup selamanya. –Mahatma Gandhi[**)]

Kesuksesan adalah buah dari usaha-usaha kecil yang diulang hari demi hari.[***)]

[*)] https://www.kutipkata.com/motto-hidup-singkat-bermakna-berbagai-tokoh-dunia/ [Diakses pada 09 Juni 2020]
[**)] https://www.kutipkata.com/motto-hidup-singkat-bermakna-berbagai-tokoh-dunia/ [Diakses pada 09 Juni 2020]
[***)] https://goodminds.id/motto-hidup/ [Diakses pada 09 Juni 2020]

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Maria Artati Eka Setyorini

NIM : 161810101032

menyatakan dengan sesungguhnya bahwa skripsi yang berjudul "Analisis Perbandingan Metode *Machine Learning: Random Forest* dan *Support Vector Machine* Untuk Deteksi Kanker Paru-Paru" adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya dan belum pernah diajukan pada institusi manapun, serta bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenar-benarnya tanpa ada tekanan dan paksaan dari pihak manapun dan bersedia mendapat sanksi akademik jika ternyata di kemudian hari pernyataan ini tidak benar.

Jember, Juli 2020

Yang menyatakan,

Maria Artati Eka Setyorini

NIM 161810101032

SKRIPSI

ANALISIS PERBANDINGAN METODE MACHINE LEARNING: RANDOM FOREST DAN SUPPORT VECTOR MACHINE UNTUK DETEKSI KANKER PARU-PARU

Oleh

Maria Artati Eka Setyorini
NIM 161810101032

Pembimbing

Dosen Pembimbing Utama : Dian Anggraeni, S.Si., M.Si.
Dosen Pembimbing Anggota : Dr. Mohamad Fatekurohman, S.Si., M.Si.

PENGESAHAN

Skripsi berjudul "Analisis Perbandingan Metode *Machine Learning: Random Forest* dan *Support Vector Machine* Untuk Deteksi Kanker Paru-Paru" telah diuji dan disahkan pada :

hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Tim Penguji :

Ketua,	Anggota I
Dian Anggraeni, S.Si, M.Si.	Dr. M. Fatekurohman, S.Si.,M.Si.
NIP 198202162006042002	NIP 196906061998031001
Anggota II	Anggota III
Prof. I Made Tirta, M.Sc., Ph.D.	Dr. Yuliani Setya Dewi, S.Si., M.Si
NIP 195912201985031002	NIP 197407162000032001

Mengesahkan
Dekan,

Drs. Achmad Sjaifullah, M.Sc., Ph.D.
NIP 195910091986021001

RINGKASAN

Analisis Perbandingan Metode *Machine Learning*: *Random Forest* dan *Support Vector Machine* Untuk Deteksi Kanker Paru-Paru; Maria Artati Eka Setyorini, 161810101032; 2020; 146 halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Kanker paru-paru merupakan penyakit dengan adanya pertumbuhan sel tidak terkontrol pada jaringan paru-paru. Kanker paru-paru dapat menyerang siapa saja dan sering tidak menimbulkan gejala awal, maka perlu adanya deteksi kanker. Penelitian ini tentang deteksi kanker menggunakan Algoritma *Data Mining* dengan metode *machine learning Random Forest* dan *Support Vector Machine* (SVM). *Random Forest* diawali dengan teknik klasifikasi dasar *decision tree.* Sesuai dengan namanya, konsep metode klasifikasi ini menciptakan sebuah hutan (*forest*) dengan sejumlah pohon (*tree*) secara acak (*random*). Sedangkan konsep metode klasifikasi SVM menjelaskan bagaimana upaya sederhana untuk menemukan fungsi pemisah terbaik (*hyperplane*). Dasarnya SVM bekerja dengan prinsip *linier classifier*, kemudian dikembangkan untuk dapat bekerja pada kasus *non linear* dengan menggunakan konsep kernel pada ruang kerja berdimensi tinggi.

Hasil klasifikasi *Random Forest* menghasilkan akurasi sebesar 90,32%. Sedangkan, hasil klasifikasi SVM menghasilkan akurasi sebesar 87,10%. Supaya dapat menampilkan visualisasi dari hasil klasifikasi masing-masing metode pada ruang 2D maka penelitian ini menggunakan PCA *(Principal Component Analysis)*. PCA juga sangat berpengaruh dalam meningkatkan akurasi dalam sebuah metode klasifikasi, dapat dilihat pada metode *Random Forest* menggunakan PCA menghasilkan akurasi sebesar 100% dan metode SVM menggunakan PCA menghasilkan akurasi sebesar 93,47%.

Tujuan dari penelitian ini adalah melakukan perbandingan akurasi di antara metode-metode klasifikasi yang sudah di analisis seperti metode *Random Forest*, SVM, PCA-*Random Forest*, dan PCA-SVM. Di antara empat metode tersebut dapat ditarik kesimpulan bahwa metode PCA-*Random Forest* menghasilkan tingkat akurasi tertinggi sebesar 100%. Artinya, metode tersebut sangat baik dalam

mengelompokkan kelas orang normal dan kelas orang terdiagnosis kanker paru-paru tanpa adanya misklasifikasi. Visualisasi hasil klasifikasi *Random Forest* dan SVM mampu memperlihatkan berapa jumlah orang yang normal, jumlah orang yang terdiagnosis kanker paru-paru dan jumlah data yang misklasifikasi.

PRAKATA

Puji syukur kepada Tuhan Yang Maha Esa atas segala rahmat-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Analisis Perbandingan Metode *Machine Learning Random Forest* dan *Support Vector Machine* Untuk Deteksi Kanker Paru-Paru". Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan strata satu (S1) pada Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Penyusunan skripsi mendapatkan dukungan serta bantuan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Dian Anggraeni, S.Si., M.Si. selaku Dosen Pembimbing Utama dan Dr. Mohamad Fatekurohman, S.Si., M.Si. selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, tenaga, pikiran, dan perhatian dalam penulisan skripsi ini;
2. Prof. I Made Tirta, M.Sc., Ph.D. dan Dr. Yuliani Setya Dewi, S.Si., M.Si. selaku Dosen Penguji yang telah memberikan kritik dan saran yang membangun demi kesempurnaan skripsi ini;
3. Ika Hesti Agustin, S.Si., M.Si. dan Bagus Juliyanto, S.Si., M.Si. selaku Dosen Pembimbing Akademik yang memberikan berbagai dukungan, motivasi dan pengarahan selama penulis menjadi mahasiswa;
4. Seluruh Dosen dan Staff Karyawan Jurusan Matematika Fakultas MIPA Universitas Jember;
5. Teman-teman HIMATIKA "Geokompstat" Fakultas MIPA Universitas Jember;
6. Semua pihak yang tidak dapat disebutkan satu per satu oleh penulis.

Guna menyempurnakan skripsi ini, penulis menerima kritik dan saran dari berbagai pihak. Penulis berharap, semoga skripsi ini dapat bermanfaat untuk penelitian-penelitian berikutnya.

Jember, Juli 2020

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR LAMPIRAN

Halaman

BAB I

PENDAHULUAN

1.1 Latar Belakang

Kanker dikenal sebagai salah satu penyakit yang paling ganas di dunia. Kanker adalah suatu istilah untuk penyakit di mana sel-sel membelah secara abnormal tanpa kontrol dan dapat menyerang jaringan di sekitarnya. Banyak manusia takut akan bahaya kanker karena bisa berujung pada kematian. Setiap tahun semakin meningkat jumlah kematian penderita kanker, dengan spesifikasi kanker paru-paru, kanker payudara, dan kanker serviks yang menduduki jumlah kematian terbesar. Berdasarkan data kanker yang pernah dirilis oleh *World Health Organization* (WHO), menyebutkan bahwa kematian terbanyak perempuan yang mengidap kanker payudara dan kanker serviks sedangkan kematian terbanyak pada laki-laki yang mengidap kanker paru-paru.

Kanker paru-paru merupakan penyakit dengan adanya pertumbuhan sel yang tidak terkontrol pada jaringan paru-paru (*National Cancer Institute,* 2009). Kanker paru-paru cenderung lebih banyak dialami oleh orang yang memiliki kebiasaan merokok. Kebiasaan merokok kebanyakan terjadi pada kaum pria, walaupun ada pula sedikit wanita yang merokok. Beberapa sumber menyebutkan bahwa kanker paru-paru juga bisa terjadi pada orang yang bukan perokok, seperti terpapar zat yang bersifat karsinogen, genetik dan lingkungan. Kebiasaan gaya hidup sehari-hari pun dapat menjadi dasar munculnya kanker paru-paru. Namun sayangnya, kanker paru-paru sering tidak menimbulkan gejala awal. Gejala awal barulah muncul saat tumor sudah semakin membesar atau kanker yang sudah menyebar ke jaringan dan organ sekitarnya. Sampai saat ini masih belum ada cara penanganan tepat untuk mengatasi kanker. Hanya cara sederhana yang dapat dilakukan oleh semua orang adalah berusaha untuk hidup sehat dan peka terhadap gejala awal yang muncul pada tubuh. Jikalau gejala awal berlangsung secara terus menerus, segera di bawa ke dokter untuk dilakukannya pemeriksaan secara dini.

Penting adanya pemeriksaan sedini mungkin supaya mendapatkan penanganan maupun antisipasi dengan cepat dan tepat. Melihat betapa pentingnya deteksi kanker membuat banyak peneliti yang ingin melakukan penelitian tentang diagnosis kanker.

Penelitian tentang deteksi kanker dapat menggunakan Algoritma *Data Mining*. Algoritma *Data Mining* berisikan proses untuk membangun model dari suatu data. Model tersebut dapat digunakan untuk menggambarkan kondisi dari suatu data dan membuat prediksi. Hal ini membuat penulis tertarik untuk melakukan penelitian tugas akhir mengenai Algoritma *Data Mining* dalam diagnosis kanker paru-paru dengan menggunakan metode *machine learning Support Vector Machine* (SVM) dan *Random Forest* yang dijalankan dengan menggunakan program R. Pada dasarnya, SVM merupakan sebuah algoritma klasifikasi untuk data *linear*. Namun jika terdapat data *non-linear* maka di proses dengan menggunakan kernel SVM pilihan terbaik. Metode SVM dikenal sebagai metode klasifikasi yang memiliki nilai akurasi tertinggi sehingga sudah banyak digunakan dalam beberapa penelitian dan tidak perlu lagi dibandingkan dengan metode klasifikasi lainnya. Namun kenyataannya, masih banyak penelitian yang membandingkan SVM dengan metode klasifikasi lainnya seperti pada jurnal-jurnal berikut.

Jurnal-jurnal berikut masing-masing memiliki kesimpulan berbeda yaitu Fachruddin (2015) menyimpulkan "Hasil Perbandingan Metode *Random Forest Classification* Dan *Support Vector Machine* Untuk Deteksi Epilepsi Menggunakan Data Rekaman *Electroen Cephalograph (EGG) yaitu* SVM memiliki rata-rata akurasi yang lebih baik dari pada *Random Forest*." Darmawan (2016) menyimpulkan "Hasil Deteksi Dini Penyakit Kanker Leher Rahim (Serviks) Di Kota Bogor Mengunakan *Regresi Logistik Biner* Dan *Support Vector Machine* (SVM) yaitu dengan memiliki nilai akurasi lebih tinggi SVM mampu mendeteksi lebih baik daripada metode Regresi Logistik Biner." Sedangkan, Aliady (2018) menyimpulkan "Hasil Implementasi *Support Vector Machine* (SVM) dan *Random Forest* Pada Diagnosis Kanker Payudara yaitu metode *Random Forest* memiliki tingkat akurasi yang lebih tinggi dari metode SVM."

Perbedaan kesimpulan dari kedua jurnal membuat penulis tergerak untuk membandingkan antara SVM dengan *Random Forest*. *Random Forest* adalah salah satu algoritma yang digunakan pada klasifikasi data dalam jumlah besar. *Random Forest* merupakan salah satu diantara banyak metode klasifikasi statistika yang menggunakan dasar Algoritma *Decision Tree* (Pohon Keputusan). *Decision Tree* digunakan dalam *Random Forest* hanya untuk melakukan proses seleksi data. Berdasarkan dari data kasus kematian terbesar akibat kanker dan perbedaan kesimpulan dari jurnal acuan yang dipakai oleh penulis, maka dibuatnya tugas akhir yang berjudul "Analisis Perbandingan Metode *Machine Learning: Random Forest Dan Support Vector Machine* Untuk Deteksi Kanker Paru-Paru"

1.2 Rumusan Masalah

Rumusan masalah dalam penelitian ini sebagai berikut

1) Bagaimana analisis metode *machine learning Random Forest* mampu mendeteksi kanker paru-paru?
2) Bagaimana analisis metode *machine learning* SVM mampu mendeteksi kanker paru-paru?
3) Bagaimana hasil perbandingan dari analisis metode *machine learning* SVM dan *Random Forest* dalam mendeteksi kanker paru-paru?

1.3 Tujuan Penelitian

Adapun tujuan penelitian ini sebagai berikut

1) Untuk menganalisis metode *Random Forest* dalam mendeteksi kanker paru-paru.
2) Untuk menganalisis metode SVM dalam mendeteksi kanker paru-paru.
3) Untuk membandingkan dua metode *machine learning* yaitu metode SVM dan *Random Forest* dalam mendeteksi kanker paru-paru sehingga dapat diketahui metode mana yang menghasilkan nilai akurasi lebih tinggi.

1.4 Manfaat Penelitian

Manfaat yang ingin didapatkan dari penelitian ini adalah sebagai berikut

1) Membantu para mahasiswa dalam melakukan pengembangan topik dari penelitian ini sebagai bahan tugas akhir.
2) Membantu para statistikawan dalam menggunakan metode-metode klasifikasi statistika agar mengetahui metode klasifikasi *machine learning* mana yang memiliki nilai akurasi tinggi.
3) Membantu programmer dalam membuat suatu sistem kecerdasan buatan yang secara otomatis dapat mendeteksi masalah-masalah biologis seperti kanker paru-paru dengan menggunakan data ekspresi gen, dan lain sebagainya.
4) Membantu meningkatkan kualitas pelayan medis dalam mendeteksi kanker, khususnya kanker paru-paru.
5) Membantu masyarakat untuk mengetahui gejala-gejala awal kanker, terutama kanker paru-paru, sehingga penderita dapat melakukan tindakan pencegahan.

BAB II

TINJAUAN PUSTAKA

Pada bab ini, disajikan teori-teori yang digunakan sebagai dasar melakukan penelitian, antara lain yaitu kanker paru-paru, *data mining*, *machine learning*, *Random Forest*, SVM (*Support Vector Machine)*, PCA (*Principal Component Analysis)*, *K-Fold Cross Validation*, dan *Confusion Matrix*. Teori-teori tersebut dijelaskan sebagai berikut :

2.1 Kanker

Kanker adalah pertumbuhan sel tidak beraturan yang muncul dari satu sel. Kanker merupakan pertumbuhan jaringan secara otonom, tidak mengikuti aturan dan regulasi sel yang tumbuh secara normal. Penyakit kanker merupakan penyakit dengan karakteristik adanya gangguan atau kegagalan dalam mekanisme pengaturan multiplikasi pada organisme multiseluler sehingga terjadi perubahan perilaku sel yang tidak terkontrol (Kementerian Kesehatan Republik Indonesia, 2015). Kanker sering dikenal oleh masyarakat sebagai tumor, padahal tidak semua tumor adalah kanker. Tumor adalah segala bentuk benjolan tidak normal atau abnormal. Tumor dibagi dalam 2 golongan yaitu tumor jinak dan tumor ganas. Kanker adalah istilah umum untuk semua jenis tumor ganas. Kanker dapat menimpa semua orang pada setiap bagian tubuh dan pada semua golongan umur namun lebih sering menimpa orang yang berusia 40 tahun. Umumnya, sebelum kanker meluas atau merusak jaringan disekitarnya, penderita tidak merasakan adanya keluhan ataupun gejala. Bila sudah ada keluhan atau gejala biasanya penyakitnya sudah stadium lanjut (Yayasan Sosialisasi Kanker Indonesia, 2016).

Di Indonesia, kasus baru akibat kanker yang menjadi peringkat paling atas diduduki oleh kanker payudara. Tidak hanya kanker payudara yang menjadi peringkat paling atas melainkan kanker paru-paru merupakan peringkat paling atas kematian terbanyak akibat kanker. Estimasi presentase kasus baru dan kematian akibat kanker khususnya di Indonesia dapat dilihat pada gambar 2.1

Indonesia
Source: Globocan 2018

Incidence, Mortality and Prevalence by cancer site

	New cases				Deaths				5-year prevalence (all ages)	
Cancer	Number	Rank	(%)	Cum.risk	Number	Rank	(%)	Cum.risk	Number	Prop.
Breast	58 256	1	16.7	4.61	22 692	2	11.0	1.96	160 653	121.23
Cervix uteri	32 469	2	9.3	2.58	18 279	3	8.8	1.64	84 201	63.54
Lung	30 023	3	8.6	1.47	26 095	1	12.6	1.30	28 937	10.85
Liver	18 468	4	5.3	0.88	18 148	4	8.8	0.87	14 383	5.39
Nasopharynx	17 992	5	5.2	0.73	11 204	6	5.4	0.52	48 401	18.14
Colon	15 245	6	4.4	0.72	9 207	7	4.4	0.40	32 595	12.22
Non-Hodgkin lymphoma	14 164	7	4.1	0.64	7 565	9	3.7	0.35	35 490	13.30
Rectum	14 112	8	4.0	0.66	6 827	10	3.3	0.31	32 069	12.02
Leukaemia	13 498	9	3.9	0.50	11 314	5	5.5	0.43	35 870	13.44
Ovary	13 310	10	3.8	1.06	7 842	8	3.8	0.69	32 818	24.76
Thyroid	11 470	11	3.3	0.47	2 119	19	1.0	0.07	34 249	12.84
Prostate	11 361	12	3.3	1.41	5 007	11	2.4	0.41	23 055	17.17
Corpus uteri	6 745	13	1.9	0.57	2 407	16	1.2	0.21	18 559	14.00
Bladder	6 716	14	1.9	0.36	3 375	14	1.6	0.17	17 151	6.43
Brain, nervous system	5 323	15	1.5	0.21	4 229	13	2.0	0.18	13 051	4.89
Lip, oral cavity	5 078	16	1.5	0.23	2 326	17	1.1	0.11	12 669	4.75
Pancreas	4 940	17	1.4	0.23	4 812	12	2.3	0.23	3 430	1.29
Larynx	3 188	18	0.91	0.16	1 564	20	0.75	0.07	7 845	2.94
Stomach	3 014	19	0.86	0.14	2 521	15	1.2	0.11	3 743	1.40
Multiple myeloma	2 717	20	0.78	0.14	2 250	18	1.1	0.12	5 884	2.21
Salivary glands	2 330	21	0.67	0.11	890	24	0.43	0.05	5 080	1.90
Kidney	2 112	22	0.61	0.09	1 225	21	0.59	0.06	4 910	1.84
Melanoma of skin	1 392	23	0.40	0.07	797	25	0.38	0.04	3 703	1.39
Testis	1 382	24	0.40	0.09	283	31	0.14	0.03	4 818	3.59
Oropharynx	1 303	25	0.37	0.06	626	26	0.30	0.03	3 582	1.34
Gallbladder	1 217	26	0.35	0.06	1 056	23	0.51	0.05	1 370	0.51
Oesophagus	1 154	27	0.33	0.05	1 058	22	0.51	0.05	1 079	0.40
Vulva	1 153	28	0.33	0.10	420	28	0.20	0.04	3 403	2.57
Hodgkin lymphoma	1 047	29	0.30	0.05	574	27	0.28	0.03	3 242	1.22
Penis	899	30	0.26	0.09	334	30	0.16	0.03	2 361	1.76
Anus	660	31	0.19	0.03	352	29	0.17	0.02	1 511	0.57
Vagina	412	32	0.12	0.04	208	32	0.10	0.02	1 074	0.81
Hypopharynx	229	33	0.07	0.01	134	34	0.06	0.01	324	0.12
Mesothelioma	206	34	0.06	0.01	171	33	0.08	0.01	216	0.08
Kaposi sarcoma	91	35	0.03	0.01	63	35	0.03	0.00	220	0.08
All cancer sites	348 809	-	-	14.35	207 210	-	-	9.07	775 120	290.53

Gambar 2.1. Estimasi Persentase Kasus Baru dan Kematian Akibat Kanker di Indonesia (Globocan, 2018).

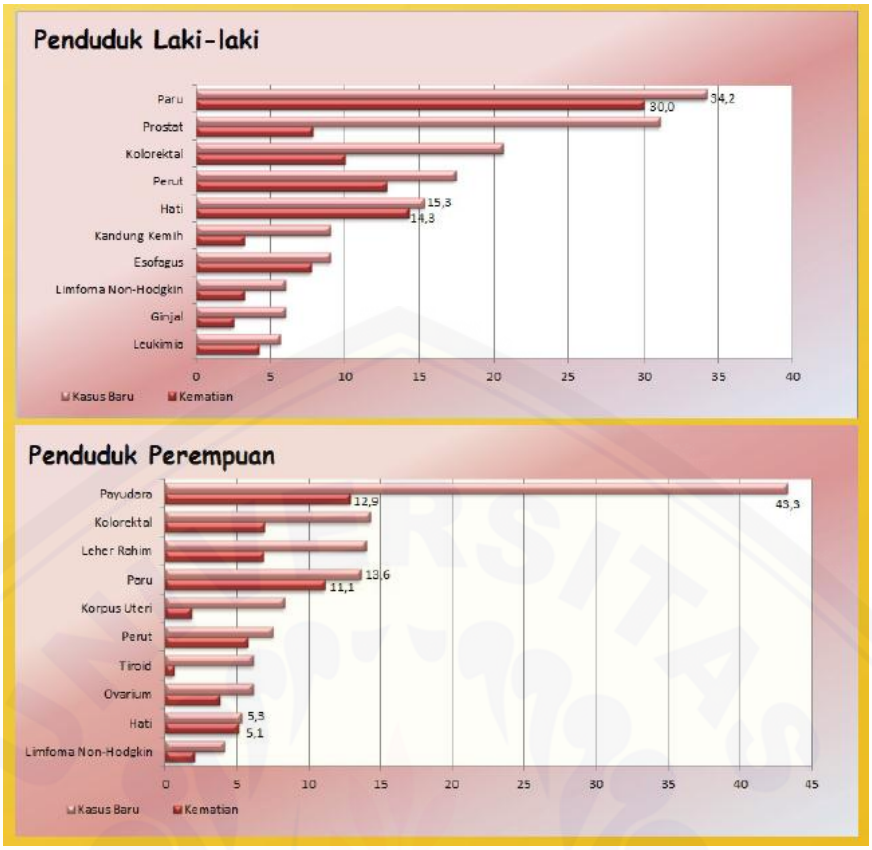
2.2 Kanker Paru-Paru

Kanker adalah kelompok penyakit yang menyebabkan sel-sel di dalam tubuh berubah dan tumbuh tak terkendali. Sebagian besar jenis sel kanker kadang membentuk benjolan atau massa yang disebut tumor, dan dinamai sesuai dengan bagian tubuh dimana tumor berasal (*American Cancer Society*, 2015). Kanker paru adalah pertumbuhan sel kanker yang tidak terkendali dalam jaringan paru yang dapat disebabkan oleh sejumlah karsinogen, terutama asap rokok. Penyebab kanker paru sangat berkaitan dengan kebiasan merokok. Asap rokok yang telah diidentifikasi dapat menyebabkan kanker dengan 63 jenis bersifat karsinogen dan beracun. Kasus kanker paru paling banyak disebabkan oleh perokok aktif sebesar 80%, dimana perokok pasif 20% beresiko terkena kanker paru. Selain faktor utama

penyebab kanker paru, terdapat faktor lain seperti polusi udara, gaya hidup, genetik dan lingkungan.

Menurut *World Health Organization* (WHO), kanker paru-paru merupakan penyebab kematian utama dalam kelompok kanker. Keluhan dan gejala penyakit ini tidak spesifik, seperti batuk darah, batuk kronik, berat badan menurun dan gejala lain yang juga dapat dijumpai pada jenis penyakit paru lain. Penemuan dini penyakit ini berdasarkan keluhan saja jarang terjadi, biasanya keluhan yang ringan terjadi pada mereka yang telah memasuki *stage* II dan III. Di Indonesia kasus kanker paru terdiagnosis ketika penyakit telah berada pada staging lanjut. Kasus kanker paru yang ditemukan sejak stadium awal kemungkinan hanya sekitar 15% saja. Dengan meningkatnya kesadaran masyarakat tentang penyakit ini, disertai dengan meningkatnya pengetahuan dokter dan peralatan diagnostik maka pendeteksian dini seharusnya dapat dilakukan secara maksimal. Deteksi dini dan penanganan yang tepat diharapkan mampu mengurangi angka kematian yang diakibatkan oleh kanker paru-paru serta mampu meningkatkan angka harapan hidup (Perhimpunan Dokter Paru Indonesia, 2003).

Kanker paru-paru tidak hanya diidap oleh kaum laki-laki saja melainkan kaum perempuan juga dapat mengidap kanker paru-paru. Bukan perempuan perokok saja yang bisa terkena kanker paru-paru tetapi perempuan yang lebih terpapar zat-zat karsinogen, gaya hidup tidak sehat, genetik, bahkan faktor lingkungan bisa saja terkena kanker paru-paru. Dapat dilihat pada gambar 2.2 bahwa kasus baru akibat kanker paru-paru lebih banyak di derita oleh kaum perempuan, sedangkan kematian akibat kanker paru-paru lebih banyak di derita oleh kaum laki-laki.



Gambar 2.2. Perbandingan kasus baru dan kematian akibat kanker paru-paru yang di derita penduduk laki-laki dan perempuan (Globocan, 2012).

2.3 *Data Mining*

Data Mining adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan *machine learning*. *Data mining* digunakan untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai database besar (Turban, 2005). Proses yang umumnya dilakukan oleh *data mining* antara lain: deskripsi, prediksi, estimasi, klasifikasi, *clustering* dan asosiasi. Secara rinci proses *data mining* dijelaskan sebagai berikut:

a. Deskripsi

Deskripsi bertujuan untuk mengidentifikasi pola yang muncul secara berulang pada suatu data dan mengubah pola tersebut menjadi aturan dan kriteria yang dapat mudah dimengerti oleh para ahli pada domain aplikasinya. Aturan yang dihasilkan harus mudah dimengerti agar dapat dengan efektif meningkatkan tingkat

pengetahuan (*knowledge*) pada sistem. Tugas deskriptif merupakan tugas *data mining* yang sering dibutuhkan pada teknik *postprocessing* untuk melakukan validasi dan menjelaskan hasil dari proses *data mining*. *Postprocessing* merupakan proses yang digunakan untuk memastikan hanya hasil yang *valid* dan berguna yang dapat digunakan oleh pihak yang berkepentingan.

b. Prediksi

Prediksi memiliki kemiripan dengan klasifikasi, akan tetapi data diklasifikasikan berdasarkan perilaku atau nilai yang diperkirakan pada masa yang akan datang. Contoh dari tugas prediksi misalnya untuk memprediksikan adanya pengurangan jumlah pelanggan dalam waktu dekat dan prediksi harga saham dalam tiga bulan yang akan datang.

c. Estimasi

Estimasi hampir sama dengan prediksi, kecuali variabel target estimasi lebih ke arah numerik dari pada ke arah kategori. Model dibangun menggunakan *record* lengkap yang menyediakan nilai dari variabel target sebagai nilai prediksi. Selanjutnya, pada peninjauan berikutnya estimasi nilai dari variabel target dibuat berdasarkan nilai variabel prediksi. Sebagai contoh, akan dilakukan estimasi tekanan darah sistolik pada pasien rumah sakit berdasarkan umur pasien, jenis kelamin, berat badan, dan level sodium darah. Hubungan antara tekanan darah sistolik dan nilai variabel prediksi dalam proses pembelajaran akan menghasilkan model estimasi.

d. Klasifikasi

Klasifikasi merupakan proses menemukan sebuah model atau fungsi yang mendeskripsikan dan membedakan data ke dalam kelas-kelas. Klasifikasi melibatkan proses pemeriksaan karakteristik dari objek dan memasukkan objek ke dalam salah satu kelas yang sudah didefinisikan sebelumnya.

e. *Clustering*

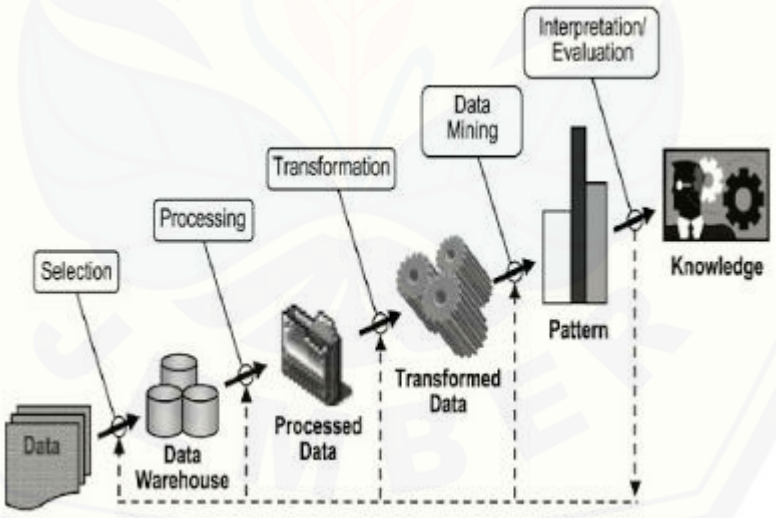
Clustering merupakan pengelompokan data tanpa berdasarkan kelas data tertentu ke dalam kelas objek yang sama. Sebuah kluster adalah kumpulan *record* yang memiliki kemiripan suatu dengan yang lainnya dan memiliki ketidakmiripan

dengan *record* dalam kluster lain. Tujuannya adalah untuk menghasilkan pengelompokan objek yang mirip satu sama lain dalam kelompok-kelompok. Semakin besar kemiripan objek dalam suatu kluster dan semakin besar perbedaan tiap kluster maka kualitas analisis kluster semakin baik.

f. Asosiasi

Tugas asosiasi dalam data mining adalah menemukan atribut yang muncul dalam suatu waktu. Dalam dunia bisnis lebih umum disebut analisis keranjang belanja (*market basket analisys*). Tugas asosiasi berusaha untuk mengungkap aturan untuk mengukur hubungan antara dua atau lebih atribut (Larose, 2005).

Tahapan yang dilakukan pada proses *data mining* diawali dari seleksi data dari data sumber ke data target, tahap *preprocessing* untuk memperbaiki kualitas data, transformasi, data mining serta tahap interpretasi dan evaluasi yang menghasilkan output berupa pengetahuan baru yang diharapkan memberikan kontribusi yang lebih baik. Melalui gambar 2.3 tahapan proses *data mining* digambarkan supaya lebih mudah dipahami



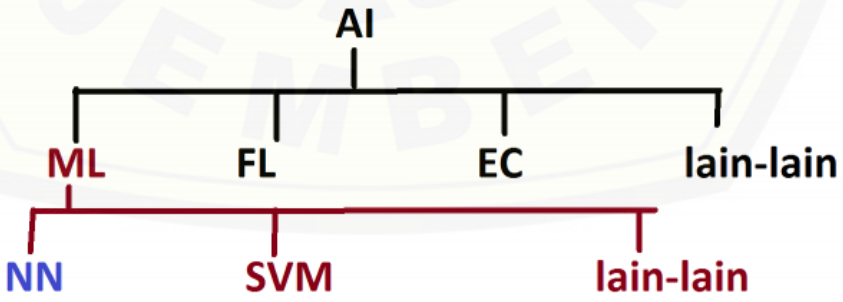
Gambar 2. 3 Skema *Data Mining* (Fayyad, 1996).

2.4 *Machine Learning*

Kecerdasan Buatan atau AI (*Artificial Intelligence*) adalah teknik yang digunakan untuk meniru kecerdasan yang dimiliki oleh makhluk hidup maupun

benda mati untuk menyelesaikan sebuah persoalan. Untuk melakukan hal itu, setidaknya ada tiga metode yang dikembangkan yaitu *Fuzzy Logic* (FL), *Evolutionary Computing* (EC), dan *Machine Learning* (ML). *Machine Learning* atau pembelajaran mesin merupakan teknik yang paling populer karena banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah. Sesuai namanya *Machine Learning* artinya mencoba menirukan bagaimana proses manusia atau makhluk cerdas belajar dan mengeneralisasi (Abu, 2017).

Machine Learning merupakan pendekatan dalam AI yang banyak digunakan untuk menggantikan atau menirukan perilaku manusia untuk menyelesaikan masalah atau melakukan otomatisasi. Setidaknya ada dua aplikasi utama dalam *Machine Learning* yaitu klasifikasi dan prediksi. Ciri khas dari *Machine Learning* adalah adanya proses pelatihan, pembelajaran, atau *training*. Oleh karena itu, *Machine Learning* membutuhkan data untuk dipelajari yang disebut sebagai data *training*. Klasifikasi adalah metode dalam *Machine Learning* yang digunakan oleh mesin untuk memilah atau mengklasifikasikan obyek berdasarkan ciri tertentu sebagaimana manusia mencoba membedakan benda satu dengan yang lain. Sedangkan, prediksi atau regresi digunakan oleh mesin untuk menerka keluaran dari suatu data masukan berdasarkan data yang sudah dipelajari dalam *training*. Pada gambar 2.4 dapat dilihat bahwa metode *Machine Learning* yang paling populer digunakan yaitu *Support Vector Machine* (SVM) dan *Neural Network (NN)*.



Gambar 2.4 Skema AI (*Artificial Intelligence*) (Tanaka, 2014).

Metode *machine learning* umumnya dikelompokkan menjadi 2 pendekatan:

a. Supervised Learning

Dalam *supervised learning*, komputer membentuk algoritma sesuai ilmu/data yang diinput/dilabel manusia untuk memprediksi luaran (*output*) dari suatu dataset pasien yang sudah diketahui sebelumnya. Keterbatasan *supervised learning* yaitu membutuhkan data dalam jumlah sangat besar sehingga memakan waktu lama karena harus dilabel manual oleh manusia, serta membutuhkan validasi berulang dengan dataset lainnya untuk melatih ketepatan model algoritma tersebut. Beberapa algoritma yang termasuk dalam *supervised learning* adalah (Krittanawong, 2017):

- Regresi Linier Berganda
- Analisis Deret Waktu (*Time Series Analysis*)
- *Decision Tree* dan *Random Forest*
- *Naive Bayes Classifier*
- *Nearest Neighbor Classifier*
- *Artificial Neural Network (ANN)*
- *Support Vector Machine (SVM)*

b. Unsupervised Learning

Pada *unsupervised learning*, komputer membentuk algoritma dari hasil pembelajaran sendiri berdasarkan data-data pasien yang sudah ada. Salah satu tujuan dari algoritma ini adalah mengelompokkan objek yang hampir sama dalam suatu area tertentu. Keuntungan *unsupervised learning* yaitu dapat menemukan pola tersembunyi dari data-data yang ada untuk mengidentifikasi, seperti mengidentifikasi mekanisme penyakit baru yang belum diketahui sebelumnya dan *unsupervised learning* dapat melakukan prediksi maupun klasifikasi tanpa perlu dilatih terlebih dahulu. Kekurangan *unsupervised learning* adalah sulitnya menemukan pola awal untuk membentuk algoritma, sehingga membutuhkan *manual coding* dalam beberapa hal di awalnya, serta validasi berulang.

Beberapa algoritma yang dapat digunakan dalam *unsupervised learning* adalah (Johnson, 2018) :

- *K-Means*
- *Hierarchical Clustering*
- DBSCAN
- *Fuzzy C-Means*
- *Self-Orginizing Map*

2.5 *Random Forest*

Random forest pertama kali dikenalkan oleh Breiman pada tahun 2001. Dalam penelitiannya, menunjukkan kelebihan *random forest* antara lain dapat menghasilkan *error* yang lebih rendah, memberikan hasil yang bagus dalam klasifikasi, dapat mengatasi data training dalam jumlah sangat besar secara efisien, dan metode yang efektif untuk mengestimasi missing data (Breiman, 2001). Akan tetapi permasalahan umum yang sering terjadi pada saat mengimplementasikan *random forest* adalah waktu pemrosesan yang lama karena menggunakan data yang banyak dan membangun model *tree* yang banyak pula untuk membentuk *random trees*.

Random Forest merupakan sebuah metode *ensemble*. Metode *ensemble* merupakan cara untuk meningkatkan akurasi metode klasifikasi dengan cara mengkombinasikan metode klasifikasi (Han, 2012). *Random Forest* diawali dengan teknik dasar data *mining* yaitu *decision tree*. Pada *decision tree, input* dimasukkan pada bagian atas *(root)* kemudian turun kebagian bawah *(leaf)* untuk menentukan data termasuk kelas apa (Breiman, 2001). Dengan kata lain *Random Forest* terdiri dari sekumpulan *decision tree* (pohon keputusan), dimana kumpulan *decision tree* tersebut digunakan untuk mengklasifikasi data ke suatu kelas.

Random Forest merupakan metode klasifikasi yang *supervised*. Sesuai dengan namanya, metode ini menciptakan sebuah hutan (*forest*) dengan sejumlah pohon (*tree*). Secara umum, semakin banyak pohon (*tree*) pada sebuah hutan *(forest)* maka semakin kuat juga hutan tersebut terlihat. Pada kasus yang sama,

semakin banyak *tree*, maka semakin besar pula akurasi yang didapatkan (Polamuri, 2017).

Decision Tree akan menggunakan *information gain* dan *gini index* untuk perhitungan dalam menentukan *root node* dan *rule*. Untuk mengetahui perhitungan *information gain* dan *gini index* dapat melihat persamaan 2.1, 2.2, dan 2.3. Sama halnya dengan *Random Forest* yang akan menggunakan *information gain* dan *gini index* untuk perhitungan dalam membangun *tree* (Han, 2012), hanya saja *Random Forest* akan membangun lebih dari satu *tree*. Masing-masing *tree* dibangun menggunakan set data dengan atribut yang diambil secara acak dari data *training*. Dengan kata lain, setiap *tree* akan bergantung pada nilai dari sampel vektor yang independen dengan distribusi yang sama pada setiap *tree* (Han, 2012). Selama proses klasifikasi setiap *tree* akan memberikan *voting* kelas yang paling populer (Han, 2012).

Metode ini juga merupakan metode pohon gabungan yang berasal dari metode *Classification And Regression Tree* (CART) dan didasarkan pada teknik pohon keputusan (*decision tree*), sehingga mampu mengatasi masalah *non-linier*. Berikut ini adalah algoritma dari metode *Random Forest*. Algoritma dibagi menjadi dua bagian, bagian pertama adalah pembuatan "*n*" pohon (*tree*) untuk membentuk hutan *(forest)* yang acak *(random)*. Bagian kedua adalah algoritma untuk melakukan prediksi dari *Random Forest* yang sudah dibuat (Han, 2012).

Input yang terdiri dari:

D = dataset yang terdiri dari d baris.

K = angka dari jumlah *tree.*

Langkah-langkah metode *Random Forest* (Bagian 1):

1. Membuat data sampel data D_i dengan mengambil acak dari dataset *D* dengan pengembalian (*replacement*).
2. Menggunakan sampel data D_i untuk membangun *tree* ke *i (i=1,2,...,k).*

3. Mengulangi langkah satu dan dua sebanyak *k*.

Metode *Random Forest* di awali dengan pemilihan *"k"* sampel dataset Di yang di ambil secara acak dengan pengembalian (*replacement*). Langkah selanjutnya adalah menggunakan dataset D_i untuk membangun *decision tree* ke *i*, metode CART dapat digunakan. Metode CART menggunakan *information gain* dalam menentukan setiap *node* pada *tree*. Sebelum menghitung *information gain* terlebih dahulu harus menghitung nilai *entropy*. Perhitungan *entropy* dapat dihitung menggunakan rumus yang dapat dilihat pada persamaan 2.1.

$$Entropy\ (S) = \sum_{i=1}^{n} - pi * log_2\ pi \quad (2.1)$$

Keterangan :

S : Himpunan kasus

n : Jumlah partisi S

pi : Proporsi Si terhadap S

Setelah menemukan nilai *entropy* maka memasukkan nilai *entropy* kedalam rumus *information gain* seperti pada persamaan 2.2 dibawah ini.

$$Gain\ (S, A) = Entropy\ (S) - \sum_{i=1}^{n} \frac{|Si|}{|S|} * Entropy\ (Si) \quad (2.2)$$

Keterangan :

S : Himpunan kasus

A : Fitur

n : Jumlah partisi atribut A

|*Si*| : Proporsi Si terhadap S

|*S*| : Jumlah kasus dalam S

Selain perhitungan *information gain*, perhitungan *gini index* juga sangat diperlukan dalam membangun sebuah *tree*. Perhitungan *gini index* dapat dilihat pada persamaan 2.3 di bawah ini.

$$Gini\ (A) = 1 - \sum_{i=1}^{n}(pi)^2 \quad (2.3)$$

Keterangan :

i : Kelas atribut

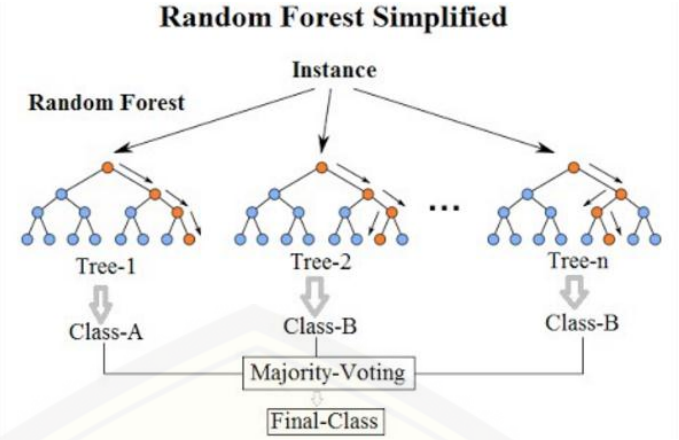
n : Jumlah kelas variable Y

pi : Proporsi jumlah kelas dalam atribut *i* terhadap jumlah kelas *n* dalam atribut

Langkah- langkah untuk proses prediksi data *test* (Bagian 2):

1. Mengambil data *test* dan menggunakan *rule* dari setiap *tree* untuk memprediksi keluaran klasifikasi dari data tersebut, lalu menyimpan hasil yang didapat
2. Menghitung suara (*vote*) untuk setiap target yang diprediksi dari setiap *tree*
3. Mempertimbangkan target prediksi yang terpilih dengan memilih target kelas yang paling banyak diprediksi sebagai hasil prediksi akhir dari metode *random forest*

Untuk memprediksi kelas target dari data *test* menggunakan *random forest*, masukkan data *test* melalui aturan-aturan (*rule*) yang sudah dibuat menggunakan *tree*. Hasil prediksi setiap *tree* bisa saja ada yang berbeda dan ada yang sama, maka prediksi akhir akan dipilih berdasarkan prediksi kelas yang terbanyak diprediksi. Misalkan dari 100 *tree*, 80 *tree* memprediksi target adalah kelas A dan sisanya kelas B, maka prediksi akhir yang dipilih adalah kelas A konsep pemilihan dengan suara terbanyak yaitu hasil prediksi dari semua *tree* dinamakan *majority voting*. Penjelasan tentang *random forest* di atas, supaya mudah dipahami dapat dijelaskan dengan melalui gambar 2.5.



Gambar 2.5 Tahapan *Random Forest* secara sederhana

Random forest adalah kombinasi dari masing – masing *tree* yang baik kemudian dikombinasikan ke dalam satu model. *Random Forest* bergantung pada sebuah nilai *vector random* dengan distribusi yang sama pada semua pohon yang masing masing *decision tree* memiliki kedalaman yang maksimal. *Random forest* adalah *classifier* yang terdiri dari *classifier* yang berbentuk pohon *{h(x, θ k), k = 1, . . .}* dengan *θ k* adalah *random vector* yang didistribusikan secara independen dan masing masing *tree* pada sebuah unit memilih kelas yang paling populer pada *input x.* Berikut ini karakteristik akurasi pada *random forest* yaitu:

1. Memusatkan *random forest*

   Terdapat *classifier h1(x), h2(x), . . . , hk (x)* dan dengan *training set* dari distribusi *random vector* X, Y. Berikut fungsi yang terbentuk

$$mg(\mathbf{X}, Y) = av_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(h_k(\mathbf{X}) = j). \quad (2.4)$$

   sedangkan, fungsi error yang digunakan

$$PE^* = P_{\mathbf{X},Y}(mg(\mathbf{X}, Y) < 0) \quad (2.5)$$

   Dari kedua fungsi di atas didapatkan hasil dari penggabungan fungsi

$$P_{\mathbf{X},Y}(P_\Theta(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j) < 0). \quad (2.6)$$

Pada hasil tersebut menjelaskan mengapa *random forest* tidak *overfit* saat *tree* di tambahkan, tetapi menghasilkan nilai yang terbatas pada error.

2. Kekuatan dan Korelasi

Fungsi yang dihasilkan adalah

$$PE^* \leq \sum_i \text{var}(P_\Theta(h(\mathbf{X}, \Theta) = Y) - P_\Theta(h(\mathbf{X}, \Theta) = j))s_j^2 \qquad (2.7)$$

Fungsi tersebut kekuatan tidak bergantung pada *forest*.

2.6 SVM *(Support Vector Machine)*

Pattern Recognition merupakan salah satu bidang dalam komputer sains untuk memetakan suatu data ke dalam konsep tertentu yang telah didefinisikan sebelumnya. Konsep tertentu ini disebut *class* atau *category* (Byun, 2003). Aplikasi *pattern recognition* sangat luas, salah satu di antaranya mengklasifikasikan penyakit secara otomatis berdasarkan hasil diagnosa kondisi medis pasien. Salah satu metode yang akhir-akhir ini banyak mendapat perhatian sebagai *state of the art* dalam *pattern recognition* adalah SVM (Tsuda, 2000).

SVM merupakan suatu teknik untuk melakukan prediksi, baik prediksi dalam kasus klasifikasi maupun regresi (Fachrurrazi, 2011). Berdasarkan dari karakteristiknya, metode SVM dibagi menjadi dua yaitu SVM *Linear* dan SVM *Non-Linear*. SVM *Linear* merupakan data yang dipisahkan secara linier yaitu memisahkan kedua kelas pada *hyperplane* dengan *soft margin*. SVM *Non-Linear* yaitu menerapkan fungsi dari kernel *trick* terhadap ruang yang berdimensi tinggi (Rachman, 2012). Karakteristik SVM secara umum dirangkum sebagai berikut:

a. Secara prinsip SVM adalah *linear classifier*.
b. *Pattern recognition* dilakukan dengan mentransformasikan data pada ruang input (*input space*) ke ruang yang berdimensi lebih tinggi (*feature space*), dan optimisasi dilakukan pada ruang vektor yang baru tersebut. Hal ini membedakan SVM dari solusi *pattern recognition* pada umumnya, yang

melakukan optimisasi parameter pada hasil transformasi yang berdimensi lebih rendah daripada dimensi *input space*.

c. Menerapkan strategi *Structural Risk Minimization* (SRM).

d. Prinsip kerja SVM pada dasarnya hanya mampu menangani klasifikasi dua kelas, namun telah dikembangkan untuk klasifikasi lebih dari dua kelas *(multiclass)* dengan adanya *pattern recognition* (Nugroho, 2003).

Berbagai studi telah menunjukkan kelebihan metode SVM dibandingkan metode konvensional lain, SVM juga memiliki berbagai kelemahan. Kelebihan SVM antara lain sebagai berikut:

a. Generalisasi

Generalisasi didefinisikan sebagai kemampuan suatu metode seperti SVM dan NN (Neural Network) untuk mengklasifikasikan suatu *pattern* yang tidak termasuk data yang dipakai dalam fase pembelajaran metode itu. *Generalization error* dipengaruhi oleh dua faktor yaitu error terhadap *training set* dan satu faktor lagi yang dipengaruhi oleh dimensi VC (Vapnik-Chervokinensis). Adapun SVM selain meminimalkan error pada *training-set* juga meminimalkan faktor kedua. Strategi ini disebut *Structural Risk Minimization* (SRM) dan dalam SVM diwujudkan dengan memilih *hyperplane* dengan margin terbesar. Berbagai studi menunjukkan bahwa pendekatan SRM pada SVM memberikan *generalization error* lebih kecil daripada yang diperoleh dari strategi pada metode lain (Vapnik, 1999).

b. *Curse of dimensionality*

Curse of dimensionality didefinisikan sebagai masalah yang dihadapi suatu metode *pattern recognition* dalam mengestimasikan parameter dikarenakan jumlah sampel data yang relatif sedikit dibandingkan dimensional ruang vektor data tersebut. Semakin tinggi dimensi dari ruang vektor informasi yang diolah, maka membawa konsekuensi dibutuhkannya jumlah data dalam proses pembelajaran. Tingkat generalisasi yang diperoleh oleh SVM tidak dipengaruhi oleh dimensi dari *input vector* (Vapnik, 1999). Hal ini merupakan alasan mengapa SVM merupakan salah satu metode yang tepat

dipakai untuk memecahkan masalah berdimensi tinggi dalam keterbatasan sampel data yang ada.

c. Landasan teori

Sebagai metode yang berbasis statistik, SVM memiliki landasan teori yang dapat dianalisa dengan jelas, dan tidak bersifat *black box.*

d. *Feasibility*

SVM relatif mudah untuk dapat diimplementasikan, karena proses penentuan *support vector* dapat dirumuskan dalam *QP problem*. Jika memiliki *library* untuk menyelesaikan *QP problem*, maka dengan sendirinya SVM dapat mudah diimplementasikan.

Selain memiliki kelebihan, SVM juga memiliki kelemahan atau keterbatasan, antara lain:

a. Sulit dipakai dalam *problem* berskala besar. Skala besar dalam hal ini dimaksudkan dengan jumlah sampel yang diolah.
b. SVM secara teoritik dikembangkan untuk masalah klasifikasi dengan dua kelas. Dewasa ini SVM telah dimodifikasi agar dapat menyelesaikan *multiclass-problem.* Demikian, dapat dikatakan penelitian dan pengembangan SVM pada *multiclass-problem* masih merupakan tema penelitian yang terbuka.

Konsep dasar SVM yaitu bertujuan untuk menemukan *hyperplane* terbaik sebagai pemisah dua buah kelas. Seperti gambar 2.6 yang menunjukkan pemisahan antara dua kelas yaitu kelas -1 dan kelas +1 dengan metode SVM. Gambar 2.6 memperlihatkan yang tergabung pada class –1 disimbolkan dengan warna merah (kotak), sedangkan pada class +1 disimbolkan dengan warna kuning (lingkaran). Berbagai alternatif *hyperplane* seperti *discrimination boundaries* yang ditunjukkan seperti pada gambar 2.6(a). *Hyperplane* terbaik antara kedua kelas dapat ditemukan dengan mengukur margin *hyperplane* tersebut dan mencari titik maksimalnya. Margin adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing kelas. *Pattern* yang paling dekat ini disebut sebagai *support vector*.

Data yang tersedia dinotasikan sebagai $\overrightarrow{x_i} \in \mathbb{R}^m$, sedangkan label masing-masing dinotasikan $y_i \in \{-1,+1\}$ untuk $i = 1,2,\ldots,n$ dimana i adalah banyaknya data.

Diasumsikan kedua kelas -1 dan +1 dapat terpisah secara sempurna oleh *hyperplane* berdimensi m, yang didefinisikan

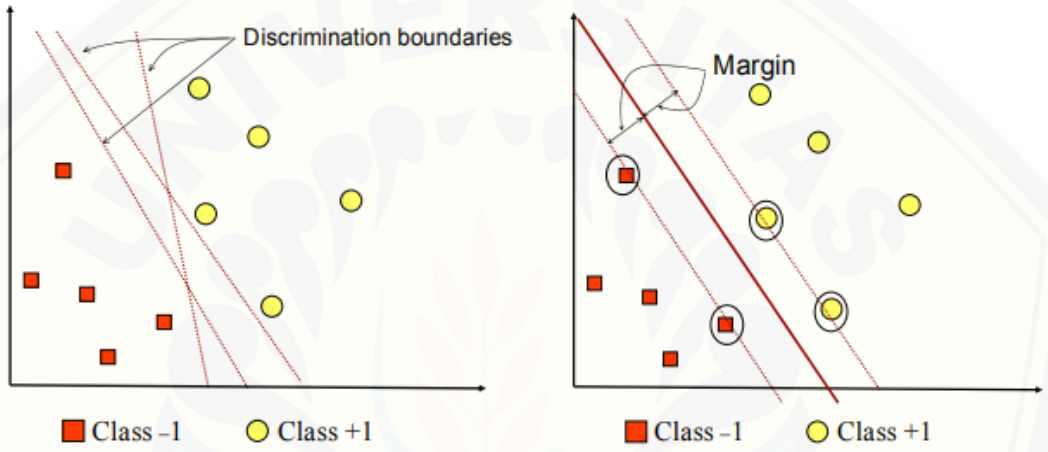
$$\vec{w}.\vec{x} + b = 0 \quad (2.8)$$

Pattern $\vec{x}_1$ yang termasuk kelas -1 (sampel negatif) dapat dirumuskan sebagai *pattern* yang memenuhi pertidaksamaan

$$\vec{w}.\vec{x}_1 + b \leq -1 \quad (2.9)$$

sedangkan *pattern* yang termasuk kelas +1 (sampel positif) (Nugroho, 2007)

$$\vec{w}.\vec{x}_1 + b \geq +1 \quad (2.10)$$



(a) (b)

Gambar 2.6 Klasifikasi 2 kelas dengan metode SVM

Margin terbesar dapat ditemukan pada gambar 2.6(b) dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya yaitu $\frac{1}{||\vec{w}||}$ ($\|w\|$ adalah norm dari vektor w). Hal ini dirumuskan sebagai Permasalahan Pemrograman Kuadratik (*QP problem*) yaitu dengan cara mencari titik minimal dengan memperhatikan batasan. Persamaan *QP problem* dapat dilihat pada persamaan 2.11

$$\min_{\vec{w}} \tau(w) = \frac{1}{2}||\vec{w}||^2 \quad (2.11)$$

$$y_i(w.x_i + b) - 1 \geq 0, \forall_i \quad (2.12)$$

Masalah ini dapat dipecahkan dengan berbagai teknik komputasi, di antaranya *Lagrange Multiplier.*

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i y_i (w.x_i + b - 1) \quad (2.13)$$

α_i adalah *Lagrange Multipliers* yang bernilai nol atau positif $(\alpha_i \geq 0)$. Nilai optimal dari persamaan 2.13 dapat dihitung dengan meminimalkan L terhadap $\vec{w}$ dan b serta memaksimalkan L terhadap α_i. Titik optimal pada gradient L = 0, maka persamaan 2.13 dapat dimodifikasi sebagai maksimalisasi *problem* yang hanya mengandung α_i saja sebagaimana persamaan 2.14 di bawah ini

$$max \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1} \alpha_i \alpha_j y_i y_j x_i x_j \quad (2.14)$$

$$\alpha_i \geq 0,\ i = 1, 2, \ldots, n \sum_{i=1}^{n} \alpha_i y_i = 0 \quad (2.15)$$

Dari hasil dari perhitungan pada persamaan 2.14 diperoleh α_i yang kebanyakan bernilai positif. Data yang berkorelasi dengan αi yang positif disebut dengan *support vector* (Prasetyo, 2012).

Berdasarkan persamaan dapat diasumsikan bahwa kedua kelas dapat terpisah secara sempurna oleh *hyperplane*. Akan tetapi, umumnya dua buah kelas tidak dapat terpisah secara sempurna. Hal ini menyebabkan kendala pada persamaan 2.13 tidak dapat terpenuhi, sehingga optimisasi tidak dapat dilakukan. Untuk mengatasi masalah ini, SVM dirumuskan ulang dengan memperkenalkan teknik *soft margin*. Dalam *soft margin*, persamaan 2.13 dimodifikasi dengan memasukkan *slack variable* ξ_i $(\xi_i \geq 0)$ adalah

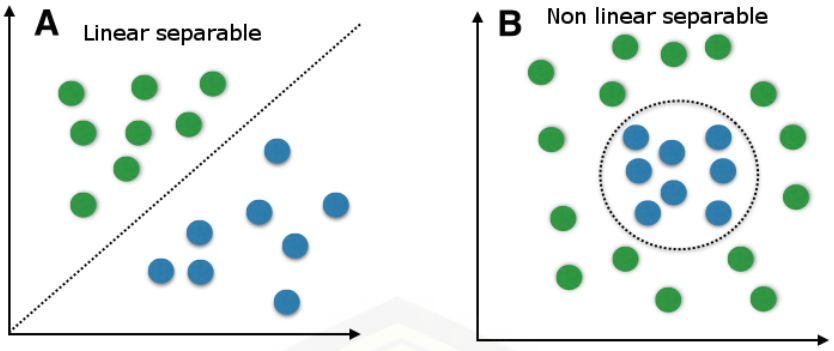
$$y_i(w.x_i + b) \geq 1 - \xi_i, \forall_i \quad (2.16)$$

Dengan demikian persamaan 2.16 menjadi

$$\min_{\vec{w}} \tau(w) = \frac{1}{2}||\vec{w}||^2 + C\sum_{i=1}^{n} \xi_i \quad (2.17)$$

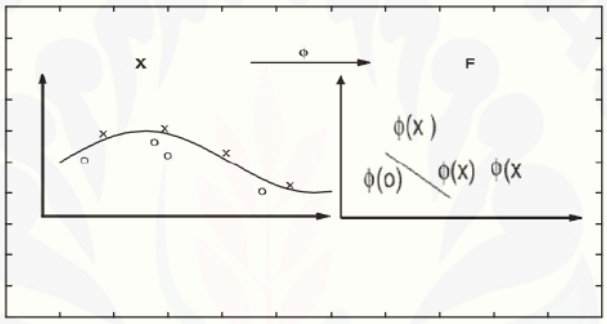
Paramater C dipilih untuk mengontrol antara margin dan error klasifikasi ξ. Apabila nilai C yang besar maka akan memberikan penalti yang lebih besar terhadap error klasifikasi tersebut.

Tingkat akurasi pada model yang akan dihasilkan oleh proses peralihan dengan SVM sangat bergantung terhadap fungsi kernel dan parameter yang digunakan (Siagian, 2011). Pada umumnya dalam permasalahan nyata, jarang ditemukan data yang bersifat *linear separable* kebanyakan bersifat *non-linear separable.* Contoh *linear separable* dan *non-linear separable* dapat dilihat seperti pada gambar 2.7.



Gambar 2.7 *Linear separable* dan *Non-linear separable*

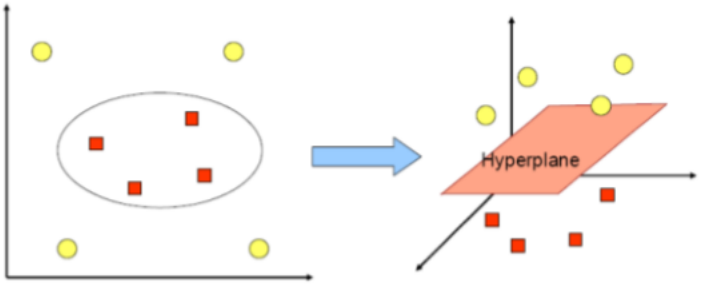
Fungsi kernel digunakan dalam SVM untuk mengatasi data yang bersifat *non-linear separable*. Ketika memasukkan fungsi kernel, maka problem data *non-linier* menjadi *linier* dalam *space* baru seperti yang tampak pada ilustrasi gambar 2.8.



Gambar 2.8 Transformasi *Non-Linear* menjadi *Linear*

Dalam SVM *non linear*, data $\vec{x}$ dipetakan oleh fungsi $\phi(\vec{x})$ ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, *hyperplane* yang memisahkan kedua kelas tersebut dapat dikonstruksikan. Ilustrasi dari konsep ini dapat dilihat pada gambar 2.9. Pada gambar 2.9(a) diperlihatkan data pada kelas kuning dan data pada kelas merah yang berada pada *input space* berdimensi dua tidak dapat dipisahkan secara linear. Selanjutnya, gambar 2.9(b) menunjukkan bahwa fungsi Φ memetakan setiap data pada *input space* tersebut ke ruang vektor baru yang berdimensi lebih tinggi (dimensi 3), dimana kedua kelas dapat dipisahkan secara linear oleh sebuah *hyperplane*. Notasi matematika dari fungsi Φ dapat di lihat pada persamaan 2.18 (Darsyah, 2013)

$$\Phi = \mathbb{R}^m \Longrightarrow \mathbb{R}^n \qquad (2.18)$$



(a) Input Space X (b) High-dimensional Feature Space Φ(X)

Gambar 2.9 Fungsi Φ mengawankan data dari ruang *input* ke ruang vektor yang berdimensi lebih tinggi.

Beberapa fungsi kernel yang umum dipakai yaitu fungsi kernel *linear*, *polynomial*, *RBF (Radial Basic Function)* atau biasa yang dikenal dengan kernel *Gaussian*, dan *sigmoid* (Nugroho, 2003). Rumus fungsi kernel tersebut terdapat pada tabel 2.1

Tabel 2.1 Kernel yang umum digunakan di SVM.

Jenis Kernel	**Definisi**
Linear	$K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i, \vec{x}_j)$
Polynomial	$K(\vec{x}_i, \vec{x}_j) = (\vec{x}_i, \vec{x}_j + 1)^p$
RBF (*Radial basic Function*) atau *Gaussian*	$K(\vec{x}_i, \vec{x}_j) = \exp\left(-\frac{\left\|\vec{x}_i - \vec{x}_j\right\|}{2\sigma^2}\right)$
Sigmoid	$K(x_i, x_j) = tanh(\alpha x_i . x_j + \beta)$

Selanjutnya, proses pembelajaran pada SVM dalam menemukan titik-titik *support vector* hanya bergantung pada *dot product* dari data yang sudah ditransformasikan pada ruang baru yang berdimensi lebih tinggi, yaitu $\Phi(\vec{x_i}).\ \Phi(\vec{x_j})$.

Umumnya transformasi Φ ini tidak diketahui dan sangat sulit untuk dipahami secara mudah, maka perhitungan *dot product* digantikan dengan fungsi kernel $K(\vec{x_i}.\ \vec{x_j})$ yang mendefinisikan secara implisit transformasi Φ . Hal tersebut disebut sebagai *Kernel Trick*, yang dirumuskan pada persamaan 2.19.

$$K(\vec{x_i}.\ \vec{x_j}) = \Phi(\vec{x_i}).\ \Phi(\vec{x_j}) \qquad (2.19)$$

Kernel trick memberikan berbagai kemudahan dalam proses pembelajaran SVM untuk menentukan *support vector* hanya cukup dengan mengetahui fungsi kernel yang dipakai dan tidak perlu mengetahui wujud dari fungsi non linear Φ. Selanjutnya, hasil klasifikasi dari data $\vec{x}$ diperoleh dari persamaan berikut :

$$f(\Phi(\vec{x}) = \overrightarrow{w}.\Phi(\vec{x}) + b \quad (2.20)$$

$$= \sum_{i=1,\vec{x}\in SV}^{n} \alpha_i \, y_i \, \Phi(\vec{x}) \, . \, \Phi(\overrightarrow{x_i}) + b \quad (2.21)$$

$$= \sum_{i=1,\vec{x}\in SV}^{n} \alpha_i \, y_i \; K(\vec{x}, \overrightarrow{x_i}) \; + b \quad (2.22)$$

Pada persamaan di atas, dimaksudkan dengan subset dari training set yang terpilih sebagai *support vector*, dimana data $\overrightarrow{x_i}$ yang berkorespondensi pada $\alpha_i \geq 0$.

2.7 PCA (*Principal Component Analysis)*

PCA digunakan untuk mengurangi besarnya dimensi dari data yang diobservasi menjadi dimensi yang lebih kecil tanpa kehilangan informasi yang signifikan dalam menggambarkan keseluruhan data. PCA merupakan kombinasi linear dari peubah yang diamati, informasi yang terkandung pada PCA merupakan gabungan dari semua peubah dengan bobot tertentu. Kombinasi *linear* yang dipilih merupakan kombinasi *linear* dengan ragam paling besar yang memuat informasi paling banyak.

Analisis Komponen Utama (*Principal Component Analysis*) adalah analisis *multivariate* yang mentransformasi variabel-variabel asal yang saling berkorelasi menjadi variabel-variabel baru yang tidak saling berkorelasi dengan mereduksi sejumlah variabel tersebut sehingga mempunyai dimensi yang lebih kecil namun dapat menerangkan sebagian besar keragaman variabel aslinya. Banyaknya komponen utama yang terbentuk sama dengan banyaknya variabel asli. Pereduksian (penyederhanaan) dimensi dilakukan dengan kriteria persentase keragaman data yang diterangkan oleh beberapa komponen utama pertama. Apabila beberapa komponen utama pertama telah menerangkan lebih dari 75% keragaman data asli, maka analisis cukup dilakukan sampai dengan komponen utama tersebut. Bila komponen utama diturunkan dari populasi multivariat normal dengan random vektor **X** = (*X*1, *X*2,… , *Xp*) dan vektor rata-rata μ = (μ1, μ2, … , μp) dan matriks

kovarians **Σ** dengan akar ciri (*eigenvalue*) yaitu $\lambda 1 \geq \lambda 2 \geq \cdots \geq \lambda p \geq 0$ didapat kombinasi linier komponen utama yaitu sebagai berikut.

$$Y_1 = e'_1 X = e'_{11} X_1 + e'_{21} X_2 + \cdots + e'_{p1} X_p$$
$$Y_2 = e'_2 X = e'_{12} X_1 + e'_{22} X_2 + \cdots + e'_{p2} X_p$$
$$\vdots$$
$$Y_p = e'_p X = e'_{1p} X_1 + e'_{2p} X_p + \cdots + e'_{pp} X_p$$

(2.23)

Maka Var(*Yi*) = *ei*'Σ*ei* dan Cov(*Yi*,*Yk*) = *ei*'Σ*ei* dimana *i*,*k* = 1, 2, ... , *p*. Syarat untuk membentuk komponen utama yang merupakan kombinasi linear dari variabel **X** agar mempunyai varian maksimum adalah dengan memilih vektor ciri (*eigen vector*) yaitu *e* = (*e*1, *e*2, …, *ep*) sedemikian hingga Var(*Yi*) = *ei*'Σ*ei* maksimum dan *ei*'*ei* = 1.

- Komponen utama pertama adalah kombinasi linear *e*1'*X* yang memaksimumkan Var(*e*1'*X*) dengan syarat *e*1'*e*1 = 1.
- Komponen utama kedua adalah kombinasi linear *e*2'*X* yang memaksimumkan Var(*e*2'*X*) dengan syarat *e*2'*e*2 = 1.
- Komponen utama ke-*i* adalah kombinasi linear *ei*'*X* yang memaksimumkan Var(*ei*'*X*) dengan syarat *ei*'*ek* = 1 dan Cov(*ei*'*ek*)=0 untuk *k* < 1.

Antar komponen utama tersebut tidak berkorelasi dan mempunyai variasi yang sama dengan akar ciri dari **Σ**. Akar ciri dari matriks ragam peragam **Σ** merupakan varian dari komponen utama **Y**, sehingga matriks ragam peragam dari **Y** adalah:

$$\mathbf{\Sigma} = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_p \end{bmatrix}$$

(2.24)

Total keragaman variabel asal akan sama dengan total keragaman yang diterangkan oleh komponen utama yaitu:

$$\sum_{j=1}^{p} \text{var}(X_i) = \text{tr}(\Sigma) = \lambda_1 + \lambda_2 + \cdots + \lambda_p = \sum_{j=1}^{p} \text{var}(Y_i) \tag{2.25}$$

Penyusutan dimensi dari variabel asal dilakukan dengan mengambil sejumlah kecil komponen yang mampu menerangkan bagian terbesar keragaman data. Apabila komponen utama yang diambil sebanyak *q* komponen, dimana *q* < *p*, maka proporsi dari keragaman total yang bisa diterangkan oleh komponen utama ke-*i* adalah:

$$\frac{\lambda_i}{\lambda_1 + \lambda_2 + \cdots + \lambda_p} \qquad i = 1, 2, \ldots, p \tag{2.26}$$

Penurunan komponen utama dari matriks korelasi dilakukan apabila data sudah terlebih dahulu ditransformasikan ke dalam bentuk baku **Z**. Transformasi ini dilakukan terhadap data yang satuan pengamatannya tidak sama. Bila variabel yang diamati ukurannya pada skala dengan perbedaan yang sangat lebar atau satuan ukurannya tidak sama, maka variabel tersebut perlu dibakukan (*standardized*). Variabel baku (**Z**) didapat dari transformasi terhadap variabel asal dalam matriks berikut:

$$\mathbf{Z} = \left(\mathbf{V}^{1/2}\right)^{-1}(\mathbf{X} - \boldsymbol{\mu}) \tag{2.27}$$

$\mathbf{V}^{1/2}$ adalah matriks simpangan baku dengan unsur diagonal utama adalah $(\alpha ii)^{1/2}$ sedangkan unsur lainnya adalah nol. Nilai harapan E(**Z**) = 0 dan keragamannya adalah

$$\text{Cov}(\mathbf{Z}) = \left(\mathbf{V}^{1/2}\right)^{-1} \Sigma \left(\mathbf{V}^{1/2}\right)^{-1} = \rho \tag{2.28}$$

Dengan demikian komponen utama dari Z dapat ditentukan dari vektor ciri yang didapat melalui matriks korelasi variabel asal ρ. Untuk mencari akar ciri dan menentukan vektor pembobotnya sama seperti pada matriks **Σ**. Sementara *trace* matriks korelasi ρ akan sama dengan jumlah *p* variabel yang dipakai. Pemilihan komponen utama yang digunakan didasarkan pada nilai akar

cirinya, yaitu komponen utama akan digunakan jika akar cirinya lebih besar dari satu (Mattjik, 2011).

2.8 *K-Fold Cross Validation*

Akurasi model yang akan dihasikan dari proses pelatihan SVM sangat bergantung pada fungsi kernel serta parameter yang digunakan. Oleh karena itu performansinya dapat dioptimasi dengan mencari (mengestimasi) parameter terbaik. Ada beberapa cara yang dapat dilakukan antara lain *cross validation* (mudah digunakan) dan *leave-one-out* (akurat tetapi membutuhkan biaya komputasi yang tinggi). *Cross-validation* adalah metode statistik yang mengevaluasi dan membandingkan algoritma pembelajaran. Secara umum, *cross validation* digunakan untuk cek kestabilan hasil klasifikasi (Payam, 2008).

Bentuk dari *cross validation* adalah *k-fold cross validation* yang berfungsi untuk menentukan nilai *folds* atau partisi data. *K-folds cross validation* juga dapat digunakan untuk menentukan nilai parameter C dan parameter kernel yang tidak *overfitting* pada data pelatihan. Dengan metode ini, data yang diambil secara acak kemudian dibagi menjadi *k* buah partisi dengan ukuran yang sama. Selanjutnya, dilakukan iterasi sebanyak *k.* Pada setiap iterasi digunakan sebuah partisi sebagai data pengujian, sedangkan *k-1* partisi sisanya digunakan sebagai data pelatihan. Metode *k-fold cross validation* yang sering dipakai adalah *5-fold cross validation* dan *10-fold cross validation*. Misalkan untuk *4-fold cross validation*, data dibagi menjadi 4 bagian seperti ilustrasi pada gambar 2.10. Setiap bagian akan digunakan untuk *training* dan *testing* secara bergantian. Tiga dari empat data digunakan untuk *training* dan satu dari empat data untuk *testing* yang dilakukan secara berulang sebanyak empat kali sampai semua bagian digunakan untuk *testing*.

Jika data bagian kedua, ketiga, dan keempat untuk *training*, maka data bagian pertama untuk *testing.* Jika data bagian pertama, ketiga, dan keempat digunakan untuk *training*, maka data kedua digunakan untuk *testing*. Jika data bagian pertama, kedua, dan keempat digunakan untuk training, maka data ketiga digunakan untuk testing. Begitu seterusnya sampai *k* yang diinginkan (Ian, 2011).

Gambar 2.10 *K-fold cross validation*

2.9 *Confusion Matrix*

Confusion Matrix adalah suatu metode yang biasanya digunakan untuk melakukan perhitungan akurasi pada konsep data mining. Pada pengukuran kinerja menggunakan *confusion matrix*, terdapat 4 (empat) istilah sebagai representasi hasil proses klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Nilai *True Negative* (TN) merupakan jumlah data negatif yang terdeteksi dengan benar, sedangkan *False Positive* (FP) merupakan data negatif namun terdeteksi sebagai data positif. Sementara itu, *True Positive* (TP) merupakan data positif yang terdeteksi benar. *False Negative* (FN) merupakan kebalikan dari *True Positive*, sehingga data posifit, namun terdeteksi sebagai data negatif (Visa, 2011). Pada jenis klasifikasi *binary* yang hanya memiliki 2 keluaran kelas, *confusion matrix* dapat disajikan seperti pada tabel 2.2.

Tabel 2.2 *Confusion Matrix*

Reference	***Prediction***	
	Positive	***Negative***
Positive	TP *(True Positive)*	FN *(False Negative)*
Negative	FP *(False Positive)*	TN *(True Negative)*

Berdasarkan nilai *True Negative* (TN), *False Positive* (FP), *False Negative* (FN), dan *True Positive* (TP) dapat diperoleh nilai akurasi (*accuration*), presisi (*precission*), sensivitas (*recall*), dan *specificity*. Nilai akurasi menggambarkan seberapa akurat sistem dapat mengklasifikasikan data secara benar. Dengan kata lain, nilai akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan keseluruhan data. Nilai akurasi dapat diperoleh dengan persamaan 2.27. Nilai presisi menggambarkan jumlah data kategori positif yang diklasifikasikan secara benar dibagi dengan total data yang diklasifikasi positif. Presisi dapat diperoleh dengan persamaan 2.28. Sementara itu, *recall* menunjukkan berapa persen data kategori positif yang terklasifikasikan dengan benar oleh sistem. *Recall* dikenal sebagai sensitivitas. Nilai *recall* diperoleh dengan persamaan 2.29. *Specificity* merupakan kebenaran memprediksi negatif dibandingkan dengan keseluruhan data negatif. *Specificity* dapat dilihat pada persamaan 2.30.

$$\text{Akurasi } (Accuration) = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \times 100\% \quad (2.29)$$

$$Presisi\ (Precission) = \frac{TP}{TP+FP} \times 100\% \quad (2.30)$$

$$Recall\ (Sensitivitas) = \frac{TP}{TP+FN} \times 100\% \quad (2.31)$$

$$Specificity = \frac{TN}{TN+FP} \times 100\% \quad (2.32)$$

Keterangan :

- TP adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem.
- TN adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem.
- FN adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah oleh sistem.
- FP adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah oleh sistem.

Penampakkan ketika *Confusion Matrix* dijalankan dengan menggunakan program R dapat dilihat pada gambar 2.11.

```
> confusionMatrix(predictions$class, y_test)
Confusion Matrix and Statistics

            Reference
Prediction   democrat republican
  democrat         60          8
  republican        6         34

               Accuracy : 0.8704
                 95% CI : (0.7921, 0.9273)
    No Information Rate : 0.6111
    P-Value [Acc > NIR] : 2.568e-09

                  Kappa : 0.7249
 Mcnemar's Test P-Value : 0.7893

            Sensitivity : 0.9091
            Specificity : 0.8095
         Pos Pred Value : 0.8824
         Neg Pred Value : 0.8500
             Prevalence : 0.6111
         Detection Rate : 0.5556
   Detection Prevalence : 0.6296
      Balanced Accuracy : 0.8593

       'Positive' Class : democrat
```

Gambar 2.11 *Output Confusion Matrix* pada program R

BAB III
METODOLOGI PENELITIAN

Pada bab ini dibahas tentang metodologi penelitian yaitu langkah-langkah dalam melakukan penelitian yang dijelaskan sebagai berikut.

3.1 Data Penelitian

Penelitian ini akan menggunakan data sekunder pasien kanker paru-paru yang diperoleh dari *dataset by Staceyin Robert* dengan alamat website https://data.world/sta427ceyin/survey-lung-cancer. Data penelitian yang didapatkan sebanyak 309 data pengamatan dengan 16 variabel. 16 variabel yaitu terdiri dari *Gender, Age, Smoking, Yellow Fingers, Axiety, Peer Pressure, Chronic Disease, Fatigue, Allergy, Wheezing, Alcohol Consuming, Coughing, Shortness Of Breath, Swallowing Difficulty, Chest Pain,* dan *Lung Cancer.*

Dalam sebuah penelitian terdapat 2 jenis variabel yang digunakan yaitu Variabel *Dependen* dan Variabel *Independen*. Variabel *Independen* (x) pada penelitian ini ada 15 variabel dengan total pengamatan (n) sebanyak 309 obyek, yaitu : *Gender* (x_1), *Age* (x_2), *Smoking* (x_3), *Yellow Fingers* (x_4), *Axiety* (x_5), *Peer Pressure* (x_6), *Chronic Disease* (x_7), *Fatigue* (x_8), *Allergy* (x_9), *Wheezing* (x_{10}), *Alcohol Consuming* (x_{11}), *Coughing* (x_{12}), *Shortness Of Breath* (x_{13}), *Swallowing Difficulty* (x_{14}), *Chest Pain* (x_{15}). Sedangkan, Variabel *Dependen* (y) pada penelitian ini terdapat 1 variabel yaitu *Lung Cancer* yang terdiri dari 2 kelas dimana kelas (NO) untuk normal dan kelas (YES) untuk terdiagnosis kanker paru-paru.

3.2 Langkah-Langkah Penelitian

Langkah-langkah penelitian dalam penelitian ini sebagai berikut.

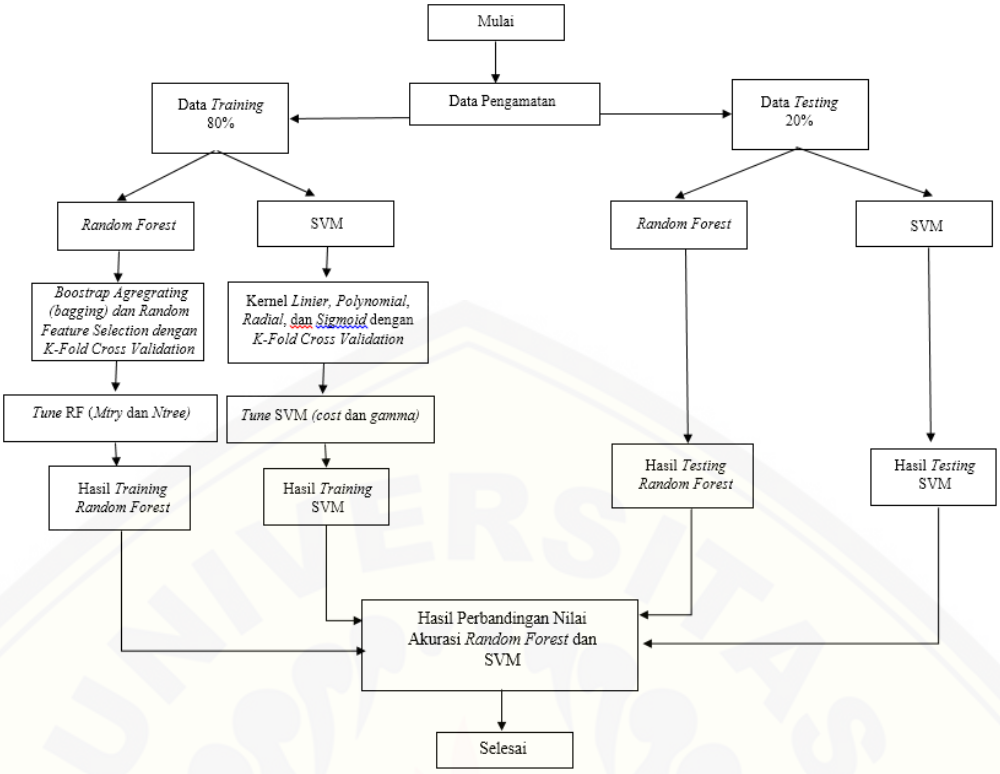
Menggunakan Data Pengamatan :

1. Mengambil data sekunder pasien kanker paru-paru dari website resmi https://data.world/sta427ceyin/survey-lung-cancer dengan format xlsx.
2. Mengolah data yang didapat menggunakan Ms. Excel.
3. Menginput data yang sudah di olah ke dalam program R (R Studio).

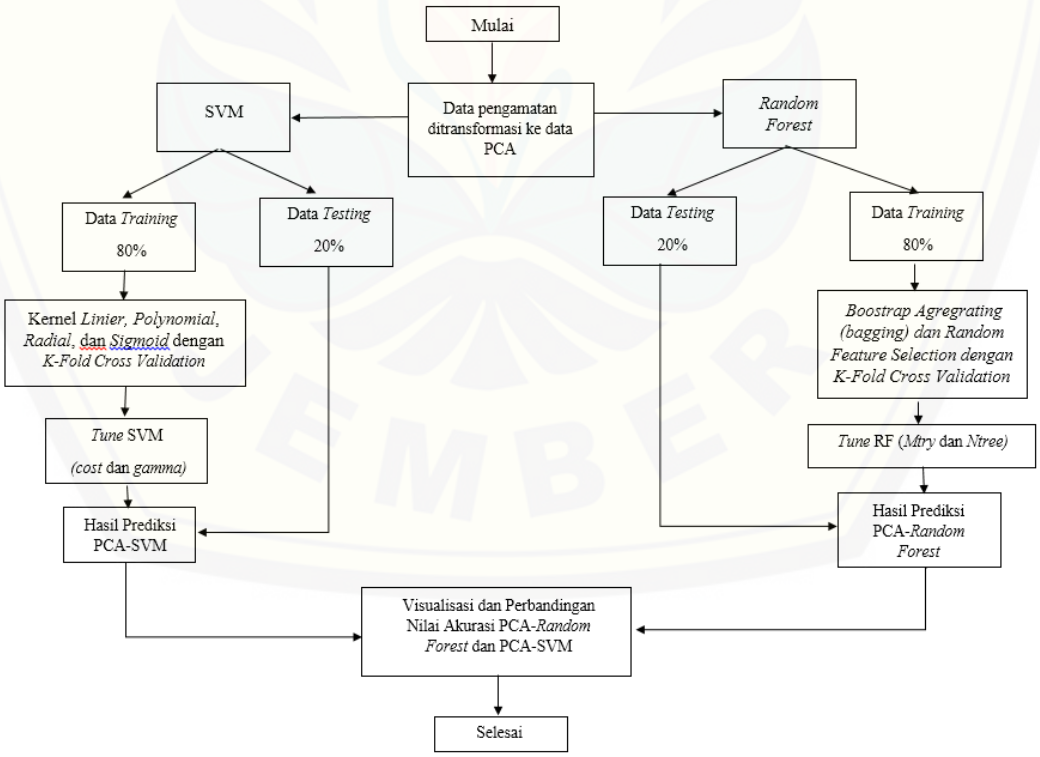
4. Menginstall *package* terlebih dahulu. *Package* yang digunakan yaitu *package* e1071, *package* randomForest, *package* caret, *package* readxl, dan *package* psych.
5. Membagi data menjadi dua data yaitu data *training* dan data *testing*.
6. Menentukan parameter terbaik pada *Random Forest (mtry, ntree).*
7. Melakukan permodelan *Random Forest* menggunakan program R.
8. Mendapatkan hasil prediksi klasifikasi *Random Forest.*
9. Menentukan fungsi kernel terbaik yang akan digunakan pada SVM yaitu kernel *Linier, Polynomial, Gaussian,* dan *Sigmoid.*
10. Melakukan permodelan SVM menggunakan program R.
11. Mendapatkan hasil prediksi klasifikasi SVM.

Menggunakan Data PCA :

12. Mengubah data pasien kanker paru-paru menjadi data PCA.
13. Melakukan permodelan PCA-*Random Forest* menggunakan program R.
14. Mendapatkan hasil prediksi klasifikasi PCA-*Random Forest*.
15. Visualisasi dengan plot hasil klasifikasi PCA-*Random Forest*
16. Melakukan permodelan PCA-SVM menggunakan program R.
17. Mendapatkan hasil prediksi klasifikasi PCA-SVM.
18. Visualisasi dengan plot hasil klasifikasi PCA-SVM dengan kernel *linear, polynomial, radial,* dan *sigmoid*.
19. Melakukan perbandingan nilai akurasi *Random Forest*, SVM, *PCA-Random Forest,* dan PCA-SVM.
20. Penarikan kesimpulan.



Gambar 3.1 Skema Penelitian Metode *Random Forest* dan SVM



Gambar 3.2 Skema Penelitian Metode PCA-*Random Forest* dan PCA-SVM

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan analisis yang telah dilakukan diperoleh beberapa kesimpulan sebagai berikut.

1.) Akurasi hasil klasifikasi menggunakan *Random Forest* sebesar 90,32%. Hasil klasifikasi PCA-*Random Forest* menunjukkan nilai akurasi yang meningkat menjadi 100%. Visualisasi hasil klasifikasi PCA-*Random Forest* mampu menggambarkan plot data dari 2 kelas kanker paru-paru yaitu NO dan YES secara tepat.

2.) Hasil klasifikasi SVM dengan kernel *linear* menunjukkan nilai akurasi sebesar 87,10%. Selain untuk menampilkan visualisasi, PCA juga dapat membantu meningkatkan akurasi klasifikasi PCA-SVM menjadi 93,47%. Visualisasi hasil klasifikasi PCA-SVM terdapat kesalahan klasifikasi sebanyak tiga data sehingga metode PCA-SVM dikatakan kurang tepat dalam melakukan klasifikasi.

3.) Perbandingan metode *machine learning* antara *Random Forest*, SVM, PCA-*Random Forest*, dan PCA-SVM dapat ditarik kesimpulan bahwa pengujian metode klasifikasi terbaik di tunjukkan oleh metode PCA-*Random Forest*.

4.) Dari hasil analisis klasifikasi *Random Forest* dan SVM terhadap deteksi kanker paru-paru banyak di derita oleh pria pada umur lebih dari 55 tahun ke atas dengan gejala yang paling dominan di alami yaitu *Alcohol Consuming* (mengonsumsi alkohol), *Chest Pain* (nyeri dada), dan *Coughing* (batuk).

5.2 Saran

Berdasarkan hasil dan kesimpulan yang diperoleh, masih terdapat kekurangan yang dapat diperbaiki pada penelitian selanjutnya. Penulis menyarankan untuk pengembangan selanjutnya adalah melakukan penelitian dengan menggabungkan metode klasifikasi SVM (*Support Vector Machine*) dan *Random Forest* karena kedua metode tersebut menghasilkan nilai klasifikasi yang

baik sebesar 90% keatas. Penggabungan kedua metode tersebut diyakini akan menghasilkan algoritma RFSVM yang mempunyai hasil lebih baik.

DAFTAR PUSTAKA

Abu, A. H. 2017. *Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning*. Yayasan Cahaya Islam, Jurnal Teknologi Indonesia. Diakses pada tanggal 13 Januari 2020 dari (https://www.researchgate.net/publication/320395378_Mengenal_Artificial_Intelligence_Machine_Learning_Neural_Network_dan_Deep_Learning)

Aliady, Hafizhan. 2018. *Implementasi Support Vector Machine (SVM) Dan Random Forest Pada Diagnosis Kanker Payudara*. Universitas Islam Indonesia.

American Cancer Society. 2015. *Breast Cancer Fact & Figures 2015 - 2016.* Atlanta: *American Cancer Society*, Inc.

Breiman, L. and Adele C. 2005. *Random Forests*. Retrieved From (http://www.Stat.Berkeley.Edu/~Breiman/Randomforests/Cc_Home.Htm , diakses 1 Oktober 2019).

Breiman, L. 1996. *Bagging Predictors* : *Machine Learning 24*, 123-140. Retrieved From(http://www.Stat.Berkeley.Edu/~Breiman/Randomforests/Cc_Home .Htm, diakses 1 Oktober 2019).

Breiman, L. 2001. *Random Forest* : *Machine Learning 45*, 5-32. Retrieved From (http://www.Stat.Berkeley.Edu/~Breiman/Randomforests/Cc_Home.Htm , diakses 1 Oktober 2019).

Breiman, L. 2001. *Machine Learning*.Berkeley:University of California.

Breiman, L., dkk. 1984. *Classification and Regression Trees*. Chapman &Hall/CRC, New York.

Byun, H., and Lee S.W. 2003. A Survey on Pattern Recognition Applications of Support Vector Machines. *International Journal of Pattern Recognition and Artificial Intelligence*, Vol.17, No.3, pp.459-486.

Darmawan, Agil. 2016. *Deteksi Dini Penyakit Kanker Leher Rahim (Serviks) Di Kota Bogor Mengunakan Regresi Logistik Biner Dan Support Vector Machine (SVM).* Institut Teknologi Sepuluh Nopember.

Darsyah, M. Y. 2013. *Menakar Tingkat Akurasi Support Vector Machine*. *Statistika, Vol. 1, No. 1*, 15-20. Retrieved From (http://www.jurnal.unimus.ac.id, diakses pada 9 Oktober 2019).

Fachruddin, Muhammad Idrus. 2015. *Perbandingan Metode Random Forest Classification Dan Support Vector Machine Untuk Deteksi Epilepsi Menggunakan Data Rekaman Electroen Cephalograph (EGG).* Institut Teknologi Sepuluh Nopember.

Fayyad, U. 1996. *Advances in Knowledge Discovery and Data Mining*. MIT Press.

Firmani, A. N. 2016. *Penyelesaian Regresi Semiparametrik dengan Menggunakan Regresi Random Forest*. Skripsi, Program Studi Statistika FMIPA UGM, Yogyakarta.

Han, J. 2012. *Data Mining Concepts and Techniques Third Edition*. USA : Elsevier.

Ho, T. K. 1995. *Random Decision Forests (PDF).* Proceedings of the 3rd International Conference on Document Analysis and Recognition, Montreal, QC, 14–16 August 1995. pp. 278–282.

Ian, H. W., Eibe Frank and Mark A. Hall. 2011. *Data Mining Practical Machine Learning Tools and Techniques.* 3rd Edition, Elsevier.

Johnson, K.W., dkk. 2018. *Artificial Intelligence in Cardiology*. J Am Col Cardiol.; 71(23): 2668-79. Retrieved From https://www.sciencedirect.com/science/article/pii/S0735109718344085?via%3Dihub

Karatzouglou, A., David M., and Kurt H. 2006. Support Vector Machine in R. *Journal of Statistic Software*. Vol. 15, Issue 9.

Kementerian Kesehatan Republik Indonesia. (2015a, September 23). *Infodatin Kanker*. Retrieved From Kementerian Kesehatan Republik Indonesia : http://www.Depkes.go.id, diakses 3 Oktober 2019).

Kementerian Kesehatan Republik Indonesia. (2015b, September 23). *Infodatin Kanker Paru-Paru*. Retrieved From Kementerian Kesehatan Republik Indonesia : http://www.Depkes.go.id, diakses 3 Oktober 2019).

Krittanawong, C., dkk. 2017. *Artificial intelligence in precision cardiovascular medicine.* J Am Coll Cardiol. 69(21): 2657-64. Retrieved From https://www.sciencedirect.com/science/article/pii/S0735109717368456?via%3Dihub

Larose, D. T. 2005. *Discovering Knowledge in Data : An Introduction to Data Mining*. John Willey & Sons, Inc.

Mattjik, A. A., dkk. 2011. *Sidik Peubah Ganda Dengan Menggunakan SAS.* Departemen Statistika Institut Pertanian Bogor : IPB Press.

Mubarok, M. I. 2018. *Pohon Regresi dengan Pendekatan Generalized Unbiased Interaction Detection Estimation (Guide) untuk Data Multirespon*. Skripsi, Program Studi Statistika FMIPA UGM, Yogyakarta.

National Cancer Institute. 2011. *Depression.* Diunduh: 25 Januari 2020. http://www.cancer.gov/cancertopics/pdq/supportivecare/depression/Patient /page2/AllPages.

Nugroho, A., Witarto A. B., dan Handoko D. 2003. *Support Vector Machine, Teori dan Aplikasinya dalam Bioinformatika.* Diakses pada tanggal 21 Mei 2019 dari [http://www.ilmukomputer.com].

Nugroho. 2017. *Sistem Klasifikasi Variabel Tingkat Penerimaan Konsumen Terhadap Mobil Menggunakan Metode Random Forest*. Diakses pada tanggal 12 Januari 2020 dari https://www.researchgate.net/publication/320413581_Sistem_Klasifikasi_Variabel_Tingkat_Penerimaan_Konsumen_Terhadap_Mobil_Menggunakan_Metode_Random_Forest.

Payam, R., Lie T., and Huan L. 2008. *Cross Validation*. Arizona State University. USA.

Perhimpunan Dokter Paru Indonesia. 2003. *Pedoman Diagnosis & Penatalaksanaan Di Indonesia.* Retrieved From https://agus34drajat.files.wordpress.com/2010/10/kankerparu.pdf

Polamuri, S. *How Random Forest Algorithm Works In Machine Learning*. Diakses pada tanggal 15 Januari 2020 dari https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learning

Prasetyo, E. 2012. *Konsep dan Aplikasi Menggunakan MATLAB.* Yogyakarta: ANDI.

Rachman, F. dan Purnami S.W. 2012. *Perbandingan Klasifikasi Tingkat Keganasan Breast Cancer dengan Menggunakan Regresi Logistik Ordinal dan Support Vector Machine*. Speirs V. Choosing The right cell line for breast cancer research. BioMed Central.

Tanaka, M. and Okutomi M. 2014. *A novel inference of a restricted boltzmann machine. Pattern Recognition (ICPR).* 22nd International Conference on pp 1526–1531.

Tsuda, K. 2000. Overview of Support Vector Machine. *Journal of IEICE*, Vol.83, No.6, pp.460-466, Japan.

Turban, E. 2005. *Decision Support Systems and Intelligent Systems Edisi Bahasa Indonesia Jilid 1*. Andi: Yogyakarta.

Vapnik, V.N. 1999. *The Nature of Statistical Learning Theory*, 2nd edition, Springer-Verlag, New York Berlin Heidelberg.

Visa, S., Brian Ramsay, A. Ralescu, and E.V.D. Knaap. 2011. Confusion Matrix-Based Feature Selection. *Midwest Artificial Intellegence and Cognitive Science Conference*, vol. 710.

Yayasan Sosialisasi Kanker Indonesia. 2016. *Penjelasan Umum Tentang Kanker*. (Yayasan Sosialisasi Kanker Indonesia : http://yski.org/info-cancer.html, diakses 2 Oktober 2019)

LAMPIRAN

Lampiran A. Data Penelitian

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COUGHING	SHORTNESS OF BREATH	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
1	M	69	1	2	2	1	1	2	1	2	2	2	2	2	2	YES
2	M	74	2	1	1	1	2	2	2	1	1	1	2	2	2	YES
3	F	59	1	1	1	2	1	2	1	2	1	2	2	1	2	NO
4	M	63	2	2	2	1	1	1	1	1	2	1	1	2	2	NO
5	F	63	1	2	1	1	1	1	1	2	1	2	2	1	1	NO
6	F	75	1	2	1	1	2	2	2	2	1	2	2	1	1	YES
7	M	52	2	1	1	1	1	2	1	2	2	2	2	1	2	YES
8	F	51	2	2	2	2	1	2	2	1	1	1	2	2	1	YES
9	F	68	2	1	2	1	1	2	1	1	1	1	1	1	1	NO
10	M	53	2	2	2	2	2	1	2	1	2	1	1	2	2	YES
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
300	M	65	2	2	2	2	1	2	2	1	1	1	2	2	1	YES
301	F	65	2	2	2	2	2	2	2	2	1	2	2	2	2	YES
302	M	64	1	2	2	2	1	1	2	1	2	1	1	2	2	YES
303	F	65	2	2	2	2	1	2	1	2	1	2	2	2	1	YES
304	M	51	1	2	1	1	2	2	2	2	2	2	2	1	2	YES
305	F	56	1	1	1	2	2	2	1	1	2	2	2	2	1	YES
306	M	70	2	1	1	1	1	2	2	2	2	2	2	1	2	YES
307	M	58	2	1	1	1	1	1	2	2	2	2	1	1	2	YES
308	M	67	2	1	2	1	1	2	2	1	2	2	2	1	2	YES
309	M	62	1	1	1	2	1	2	2	2	2	1	1	2	1	YES

Lampiran B. Pembagian Data *Training* dan Data *Testing*

```
### Dataset
> data1 <- read.csv(file.choose(), header = TRUE, sep=";")

### Training Set : Validation Set = 80 : 20 (random)
> set.seed(22)
> train <- sample(nrow(data1), 0.80*nrow(data1), replace = FALSE)
> TrainSet <- data1[train,]
> dim(TrainSet)
[1] 247  16
> ValidSet <- data1[-train,]
> dim(ValidSet)
[1] 62 16
```

Lampiran C. *Tuning Random Forest*

```
# Tuning Mtry (5fold)
> model <- train(LUNG_CANCER ~ ., # Survived is a function of the
variables we decided to include
+                data = TrainSet, # Use the train data frame as t
he training data
+                method = 'rf',# Use the 'random forest' algorith
m
+                trControl = trainControl(method = 'cv', # Use cr
oss-validation
```

```
+                                                 number = 5)) # Use 5 fo
lds for cross-validation
> model
Random Forest

247 samples
 15 predictor
  2 classes: 'NO', 'YES'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 198, 198, 197, 198, 197
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9069388  0.4577008
   8    0.9108571  0.5353224
  15    0.8906122  0.4166533

Accuracy was used to select the optimal model using the largest v
alue.
The final value used for the model was mtry = 8.

# Tuning Mtry (10fold)
> model <- train(LUNG_CANCER ~ ., # Survived is a function of the
variables we decided to include
+                data = TrainSet, # Use the train data frame as t
he training data
+                method = 'rf',# Use the 'random forest' algorith
m
+                trControl = trainControl(method = 'cv', # Use cr
oss-validation
+                                         number = 10)) # Use 5 f
olds for cross-validation
> model
Random Forest
247 samples
15 predictor
  2 classes: 'NO', 'YES'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 222, 223, 222, 222, 222, 222, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9111667  0.4640519
   8    0.9150000  0.5552002
  15    0.9068333  0.5160941

Accuracy was used to select the optimal model using the largest v
alue.
The final value used for the model was mtry = 8.
> plot(model)

# Algorithm Tune (tuneRF)
> set.seed(22)
```

```
> bestmtry <- tuneRF(x, y, stepFactor=1.5, improve=1e-5, ntree=50
0, mtry=8)
mtry = 8  OOB error = 8.41%
Searching left ...
mtry = 6        OOB error = 9.06%
-0.07692308 1e-05
Searching right ...
mtry = 12       OOB error = 8.74%
-0.03846154 1e-05
> print(bestmtry)
       mtry   OOBError
6.OOB     6 0.09061489
8.OOB     8 0.08414239
12.OOB   12 0.08737864


> # Tuning Ntree (Manual Search)
> control <- trainControl(method="repeatedcv", number=10, repeats
=3, search="grid")
> tunegrid <- expand.grid(.mtry=c(sqrt(ncol(x))))
> modellist <- list()
> fit <- train(LUNG_CANCER~., data=TrainSet, method="rf",  tuneGr
id=tunegrid, trControl=control, ntree=ntree)
> key <- toString(ntree)
> modellist[[key]] <- fit
> for (ntree in c(25,50,100,500,1000)) {
+   fit <- train(LUNG_CANCER~., data=TrainSet, method="rf",  tune
Grid=tunegrid, trControl=control, ntree=ntree)
+   key <- toString(ntree)
+   modellist[[key]] <- fit
+ }
> # compare results
> results <- resamples(modellist)
> summary(results)

Call:
summary.resamples(object = results)

Models: 25, 50, 100, 500, 1000
Number of resamples: 30


Accuracy
          Min.   1st Qu.    Median      Mean   3rd Qu. Max. NA's
25   0.8333333 0.8800000 0.9183333 0.9140085 0.9583333 1.00    0
50   0.7916667 0.8750000 0.9183333 0.9053248 0.9583333 0.96    0
100  0.8400000 0.8800000 0.9166667 0.9149231 0.9230769 1.00    0
500  0.8400000 0.8811538 0.9200000 0.9219316 0.9583333 1.00    0
1000 0.8750000 0.9166667 0.9200000 0.9217137 0.9495192 1.00    0

Kappa
            Min.   1st Qu.    Median      Mean   3rd Qu.       Max
. NA's
25   -0.06666667 0.3362832 0.5611794 0.5368707 0.7777778 1.000000
0    0
50   -0.06666667 0.2736318 0.4856982 0.4808612 0.7777778 0.834437
1    0
100  -0.06382979 0.3688791 0.4840426 0.5221800 0.6855519 1.000000
0    0
500  -0.06382979 0.3862520 0.6190476 0.5785844 0.7777778 1.000000
0    0
1000  0.00000000 0.4680851 0.5472973 0.5614559 0.7598039 1.000000
0    0
```

```
> dotplot(results)
> rf<- randomForest(LUNG_CANCER ~ ., data = TrainSet, ntree = 500
, mtry = 8)
> rf

Call:
 randomForest(formula = LUNG_CANCER ~ ., data = TrainSet, ntree =
500,      mtry = 8)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 8

        OOB estimate of  error rate: 7.69%
Confusion matrix:

       YES     NO      class.error
  YES  210     6       0.02777778
   NO  13      18      0.41935484
```

Lampiran D. Pengujian Data *Training Random Forest*

```
### Predicting on train set
> predTrain <- predict(rf, TrainSet, ntree = 500, mtry = 8, type
= "class")
> # Checking classification accuracy
> mean(predTrain == TrainSet$LUNG_CANCER)
[1] 0.996
> table(predTrain, TrainSet$LUNG_CANCER)

predValid      YES     NO
       YES     216     1
       NO      0       30

> confusionMatrix(predTrain, TrainSet$LUNG_CANCER)



Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     216     1
       NO      0       30

               Accuracy : 0.996
                 95% CI : (0.978, 1)
    No Information Rate : 0.874
    P-Value [Acc > NIR] : 1.5e-13

                  Kappa : 0.981

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.968
            Specificity : 1.000
         Pos Pred Value : 1.000
```

```
         Neg Pred Value : 0.995
             Prevalence : 0.126
         Detection Rate : 0.121
   Detection Prevalence : 0.121
      Balanced Accuracy : 0.984

       'Positive' Class : NO
```

Lampiran E. Pengujian Data *Testing Random Forest*

```
### Predicting on Validation set
> predValid <- predict(rf, ValidSet, ntree = 500, mtry = 5, type =
"class")

###Checking classification accuracy
> mean(predValid == ValidSet$LUNG_CANCER)
[1] 0.9032258
> table(predValid,ValidSet$LUNG_CANCER)
predValid      YES     NO
       YES     51      3
       NO      3       5

> confusionMatrix(predValid, ValidSet$LUNG_CANCER)
Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     51      3
       NO      3       5

               Accuracy : 0.9032
                 95% CI : (0.8012, 0.9637)
    No Information Rate : 0.871
    P-Value [Acc > NIR] : 0.2964

                  Kappa : 0.5694

 Mcnemar's Test P-Value : 1.0000

            Sensitivity : 0.62500
            Specificity : 0.94444
         Pos Pred Value : 0.62500
         Neg Pred Value : 0.94444
             Prevalence : 0.12903
         Detection Rate : 0.08065
   Detection Prevalence : 0.12903
      Balanced Accuracy : 0.78472

       'Positive' Class : NO
```

Lampiran F. Menentukan *Importance Variable Random Forest*

```
###To check important variables
> importance(rf)
                       MeanDecreaseGini
GENDER                         1.796382
AGE                            8.066360
SMOKING                        1.659181
YELLOW_FINGERS                 3.924769
ANXIETY                        2.905595
PEER_PRESSURE                  3.741379
CHRONIC.DISEASE                3.487210
FATIGUE                        2.844219
ALLERGY                        5.133087
WHEEZING                       2.761725
ALCOHOL.CONSUMING              4.799033
COUGHING                       3.545676
SHORTNESS.OF.BREATH            1.639898
SWALLOWING.DIFFICULTY          2.074746
CHEST.PAIN                     2.222466
> varImpPlot(rf)
> varImpPlot(rf,pch=19, main = "Importance Variable of Random Forest")
```

Lampiran G1. Pengujian Data *Training* Kernel *Linear* SVM

```
> library(e1071)
> library(caret)
> library(readxl)
> library(kernlab)

###Input Data
> data=read.csv("E:\\skripsi\\data\\lung cancer.csv", sep=";")
###Training sample with n observation, 80% data training sisanya
data testing
> set.seed(22)
> n=round(nrow(data)*0.80)
> n
[1] 247
> sample=sample(seq_len(nrow(data)), size = n)
> train=data[sample, ]
> dim(train)
[1] 247  16
> test=data[-sample, ]
> dim(test)
[1] 62 16
```

```
###Fitting kernel SVM with 5-fold
> tc <- tune.control(cross = 5)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="linear", trainControl=tc)
> summary(data.svm)

Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "linear", trainControl = tc)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  44

 ( 23 21 )


Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
train_pred     YES     NO
       YES     212     4
       NO      8       23
Success ratio on training set :  95.1417004048583 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     212     4
       NO      8       23

               Accuracy : 0.9514
                 95% CI : (0.9167, 0.9746)
```

```
    No Information Rate : 0.8907
    P-Value [Acc > NIR] : 0.000603

                  Kappa : 0.7657

 Mcnemar's Test P-Value : 0.386476

            Sensitivity : 0.85185
            Specificity : 0.96364
         Pos Pred Value : 0.74194
         Neg Pred Value : 0.98148
             Prevalence : 0.10931
         Detection Rate : 0.09312
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.90774

       'Positive' Class : NO


#Fitting kernel SVM with 10-fold
> tc <- tune.control(cross = 10)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classificati
on", kernel="linear", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "linear", trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  44

 ( 23 21 )

Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
```

```
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
 train_pred    YES     NO
       YES     212     4
       NO      8       23

Success ratio on training set :  95.1417004048583 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     212     4
       NO      8       23

               Accuracy : 0.9514
                 95% CI : (0.9167, 0.9746)
    No Information Rate : 0.8907
    P-Value [Acc > NIR] : 0.000603

                  Kappa : 0.7657

 Mcnemar's Test P-Value : 0.386476

            Sensitivity : 0.85185
            Specificity : 0.96364
         Pos Pred Value : 0.74194
         Neg Pred Value : 0.98148
             Prevalence : 0.10931
         Detection Rate : 0.09312
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.90774

       'Positive' Class : NO
```

Lampiran G2. Pengujian Data *Training* Kernel *Polynomial* SVM

```
###Fitting kernel SVM with 5-fold
> tc <- tune.control(cross = 5)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="polynomial", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "polynomial",
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  1
     degree:  3
     coef.0:  0
```

```
Number of Support Vectors:  91

 ( 62 29 )

Number of Classes:  2

Levels:
 NO YES


###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
    train_pred YES     NO
       YES     214     2
       NO      9       22
Success ratio on training set :  95.5465587044534 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     214     2
       NO      9       22

               Accuracy : 0.9555
                 95% CI : (0.9217, 0.9776)
    No Information Rate : 0.9028
    P-Value [Acc > NIR] : 0.001709

                  Kappa : 0.7754

 Mcnemar's Test P-Value : 0.070440

            Sensitivity : 0.91667
            Specificity : 0.95964
         Pos Pred Value : 0.70968
         Neg Pred Value : 0.99074
             Prevalence : 0.09717
         Detection Rate : 0.08907
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.93815
```

```
       'Positive' Class : NO

###Fitting kernel SVM with 10-fold
> tc <- tune.control(cross = 10)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classificati
on", kernel="polynomial", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "polynomial",
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  1
     degree:  3
     coef.0:  0

Number of Support Vectors:  91

 ( 62 29 )

Number of Classes:  2

Levels:
 NO YES


###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))


Training set confusion matrix :
    train_pred YES     NO
       YES     214     2
       NO      9       22
Success ratio on training set :  95.5465587044534 %
> confusionMatrix(train$LUNG_CANCER, train_pred)


Confusion Matrix and Statistics
```

```
               Prediction
Reference      YES     NO
       YES     214     2
       NO      9       22

               Accuracy : 0.9555
                 95% CI : (0.9217, 0.9776)
    No Information Rate : 0.9028
    P-Value [Acc > NIR] : 0.001709

                  Kappa : 0.7754

 Mcnemar's Test P-Value : 0.070440

            Sensitivity : 0.91667
            Specificity : 0.95964
         Pos Pred Value : 0.70968
         Neg Pred Value : 0.99074
             Prevalence : 0.09717
         Detection Rate : 0.08907
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.93815

       'Positive' Class : NO
```

Lampiran G3. Pengujian Data *Training* Kernel *Radial* SVM

```
###Fitting kernel SVM with 5-fold
> tc <- tune.control(cross = 5)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="radial", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "radial",
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  77

 ( 47 30 )

Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
```

```
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
 train_pred    YES     NO
       YES     213     3
       NO      8       23
Success ratio on training set :  95.5465587044534 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
                Prediction
Reference       YES     NO
       YES      213     3
       NO       8       23

               Accuracy : 0.9555
                 95% CI : (0.9217, 0.9776)
    No Information Rate : 0.8947
    P-Value [Acc > NIR] : 0.0004666

                  Kappa : 0.7821

 Mcnemar's Test P-Value : 0.2278000

            Sensitivity : 0.88462
            Specificity : 0.96380
         Pos Pred Value : 0.74194
         Neg Pred Value : 0.98611
             Prevalence : 0.10526
         Detection Rate : 0.09312
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.92421

       'Positive' Class : NO


###Fitting kernel SVM with 10-fold
> tc <- tune.control(cross = 10)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="radial", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "radial",
```

```
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  77

 ( 47 30 )

Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :

    train_pred YES     NO
       YES     213     3
       NO      8       23
Success ratio on training set :  95.5465587044534 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     213     3
       NO      8       23


               Accuracy : 0.9555
                 95% CI : (0.9217, 0.9776)
    No Information Rate : 0.8947
    P-Value [Acc > NIR] : 0.0004666

                  Kappa : 0.7821

 Mcnemar's Test P-Value : 0.2278000

            Sensitivity : 0.88462
```

```
           Specificity : 0.96380
        Pos Pred Value : 0.74194
        Neg Pred Value : 0.98611
            Prevalence : 0.10526
        Detection Rate : 0.09312
  Detection Prevalence : 0.12551
     Balanced Accuracy : 0.92421

      'Positive' Class : NO
```

Lampiran G4. Pengujian Data *Training* Kernel *Sigmoid* SVM

```
###Fitting kernel SVM with 5-fold
> tc <- tune.control(cross = 5)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="sigmoid", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "sigmoid",
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
     coef.0:  0

Number of Support Vectors:  61

 ( 32 29 )

Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
train_pred     YES     NO
```

```
       YES      211     5
       NO       12      19
Success ratio on training set :  93.1174089068826 %
> confusionMatrix(train$LUNG_CANCER, train_pred)



Confusion Matrix and Statistics
                Prediction
Reference       YES     NO
       YES      211     5
       NO       12      19

               Accuracy : 0.9312
                 95% CI : (0.8921, 0.9594)
    No Information Rate : 0.9028
    P-Value [Acc > NIR] : 0.07643

                  Kappa : 0.6529

 Mcnemar's Test P-Value : 0.14561

            Sensitivity : 0.79167
            Specificity : 0.94619
         Pos Pred Value : 0.61290
         Neg Pred Value : 0.97685
             Prevalence : 0.09717
         Detection Rate : 0.07692
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.86893

       'Positive' Class : NO

###Fitting kernel SVM witn 10-fold
> tc <- tune.control(cross = 10)
> data.svm<-svm(LUNG_CANCER ~., data = train, type="C-classifica
tion", kernel="sigmoid", trainControl=tc)
> summary(data.svm)
Call:
svm(formula = LUNG_CANCER ~ ., data = train, type = "C-classifica
tion", kernel = "sigmoid",
    trainControl = tc)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
     coef.0:  0

Number of Support Vectors:  61

 ( 32 29 )
```

```
Number of Classes:  2

Levels:
 NO YES

###Prediction training data result
> train_pred = predict(data.svm, newdata=train[-16])
> cm_train = table(train$LUNG_CANCER, train_pred)
> success_ratio <- function(cm) {
+   ratio = (sum(cm[1,1]+cm[2,2])/sum(cm))
+   return(ratio)}
> cm_train_str = capture.output(show(cm_train))
> writeLines(c(
+   "Training set confusion matrix : ",
+   cm_train_str,
+   paste("Success ratio on training set : ",
+   toString(success_ratio(cm=cm_train)*100), "%")))
Training set confusion matrix :
train_pred     YES     NO
       YES     211     5
       NO      12      19
Success ratio on training set :  93.1174089068826 %
> confusionMatrix(train$LUNG_CANCER, train_pred)
Confusion Matrix and Statistics
                Prediction
Reference       YES     NO
       YES      211     5
       NO       12      19


               Accuracy : 0.9312
                 95% CI : (0.8921, 0.9594)
    No Information Rate : 0.9028
    P-Value [Acc > NIR] : 0.07643

                  Kappa : 0.6529

 Mcnemar's Test P-Value : 0.14561

            Sensitivity : 0.79167
            Specificity : 0.94619
         Pos Pred Value : 0.61290
         Neg Pred Value : 0.97685
             Prevalence : 0.09717
         Detection Rate : 0.07692
   Detection Prevalence : 0.12551
      Balanced Accuracy : 0.86893

       'Positive' Class : NO
```

Lampiran H. Proses *Tuning* SVM

```
###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 5)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="linear",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),
tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 5-fold cross validation

- best parameters:
 cost gamma
  0.1     1

- best performance: 0.06865306

- Detailed performance results:
    cost gamma      error dispersion
1  1e-03     1 0.12538776 0.04963708
2  1e-02     1 0.12538776 0.04963708
3  1e-01     1 0.06865306 0.04815496
4  1e+00     1 0.06877143 0.03884634
5  1e+01     1 0.08073469 0.02763540
6  1e-03     2 0.12538776 0.04963708
7  1e-02     2 0.12538776 0.04963708
8  1e-01     2 0.06888980 0.04815496
9  1e+00     2 0.06877143 0.03884634
10 1e+01     2 0.08073469 0.02763540
11 1e-03     3 0.12538776 0.04963708
12 1e-02     3 0.12538776 0.04963708
13 1e-01     3 0.06888980 0.04815496
14 1e+00     3 0.06877143 0.03884634
15 1e+01     3 0.08073469 0.02763540
16 1e-03     4 0.12538776 0.04963708
17 1e-02     4 0.12538776 0.04963708
18 1e-01     4 0.06898980 0.04815496
19 1e+00     4 0.06877143 0.03884634
20 1e+01     4 0.08073469 0.02763540

###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 5)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="polynomial
", ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),
tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 5-fold cross validation

- best parameters:
  cost gamma
 0.001     1
```

```
- best performance: 0.08497959

- Detailed performance results:
    cost gamma      error dispersion
1  1e-03     1 0.08497959 0.03909897
2  1e-02     1 0.11746939 0.03942359
3  1e-01     1 0.12546939 0.01659373
4  1e+00     1 0.13363265 0.01141982
5  1e+01     1 0.13363265 0.01141982
6  1e-03     2 0.11338776 0.03420837
7  1e-02     2 0.12955102 0.01810608
8  1e-01     2 0.13363265 0.01141982
9  1e+00     2 0.13363265 0.01141982
10 1e+01     2 0.13363265 0.01141982
11 1e-03     3 0.12563265 0.02719483
12 1e-02     3 0.13363265 0.01141982
13 1e-01     3 0.13363265 0.01141982
14 1e+00     3 0.13363265 0.01141982
15 1e+01     3 0.13363265 0.01141982
16 1e-03     4 0.12955102 0.01810608
17 1e-02     4 0.13363265 0.01141982
18 1e-01     4 0.13363265 0.01141982
19 1e+00     4 0.13363265 0.01141982
20 1e+01     4 0.13363265 0.01141982


###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 5)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="radial",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),tun
econtrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 5-fold cross validation

- best parameters:
 cost gamma
   10     1

- best performance: 0.1172245

- Detailed performance results:
    cost gamma     error dispersion
1  1e-03     1 0.1253061 0.02891052
2  1e-02     1 0.1253061 0.02891052
3  1e-01     1 0.1253061 0.02891052
4  1e+00     1 0.1253061 0.02891052
5  1e+01     1 0.1172245 0.02530843
6  1e-03     2 0.1253061 0.02891052
7  1e-02     2 0.1253061 0.02891052
8  1e-01     2 0.1253061 0.02891052
9  1e+00     2 0.1212245 0.03079402
10 1e+01     2 0.1172245 0.02530843
11 1e-03     3 0.1253061 0.02891052
```

```
12 1e-02     3 0.1253061 0.02891052
13 1e-01     3 0.1253061 0.02891052
14 1e+00     3 0.1212245 0.03079402
15 1e+01     3 0.1172245 0.02530843
16 1e-03     4 0.1253061 0.02891052
17 1e-02     4 0.1253061 0.02891052
18 1e-01     4 0.1253061 0.02891052
19 1e+00     4 0.1212245 0.03079402
20 1e+01     4 0.1253061 0.02891052
```

```
###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 5)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="sigmoid",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),tun
econtrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 5-fold cross validation

- best parameters:
  cost gamma
 0.001     1

- best performance: 0.1256327

- Detailed performance results:
    cost gamma     error dispersion
1  1e-03     1 0.1256327 0.04404042
2  1e-02     1 0.1256327 0.04404042
3  1e-01     1 0.1337959 0.05122766
4  1e+00     1 0.1783673 0.06342705
5  1e+01     1 0.1987755 0.08967803
6  1e-03     2 0.1256327 0.04404042
7  1e-02     2 0.1256327 0.04404042
8  1e-01     2 0.1501224 0.05158415
9  1e+00     2 0.1865306 0.07591727
10 1e+01     2 0.1864490 0.08947410
11 1e-03     3 0.1256327 0.04404042
12 1e-02     3 0.1256327 0.04404042
13 1e-01     3 0.1297143 0.04688942
14 1e+00     3 0.1906122 0.06758770
15 1e+01     3 0.1823673 0.07212338
16 1e-03     4 0.1256327 0.04404042
17 1e-02     4 0.1256327 0.04404042
18 1e-01     4 0.1337959 0.05122766
19 1e+00     4 0.1986122 0.07821179
20 1e+01     4 0.1742857 0.06239694
```

```
###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 10)
```

```
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="linear", r
anges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),tune
control = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
    1     1

- best performance: 0.06483333

- Detailed performance results:
    cost gamma      error dispersion
1  1e-03     1 0.12566667 0.08967431
2  1e-02     1 0.12566667 0.08967431
3  1e-01     1 0.06866667 0.04691600
4  1e+00     1 0.06483333 0.04354259
5  1e+01     1 0.07300000 0.05028806
6  1e-03     2 0.12566667 0.08967431
7  1e-02     2 0.12566667 0.08967431
8  1e-01     2 0.06866667 0.04691600
9  1e+00     2 0.06483333 0.04354259
10 1e+01     2 0.07300000 0.05028806
11 1e-03     3 0.12566667 0.08967431
12 1e-02     3 0.12566667 0.08967431
13 1e-01     3 0.06866667 0.04691600
14 1e+00     3 0.06483333 0.04354259
15 1e+01     3 0.07300000 0.05028806
16 1e-03     4 0.12566667 0.08967431
17 1e-02     4 0.12566667 0.08967431
18 1e-01     4 0.06866667 0.04691600
19 1e+00     4 0.06483333 0.04354259
20 1e+01     4 0.07300000 0.05028806

###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 10)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="polynomial
", ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),
tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
  cost gamma
 0.001     1

- best performance: 0.07633333
```

```
- Detailed performance results:
    cost gamma      error dispersion
1  1e-03     1 0.07633333 0.07149549
2  1e-02     1 0.10483333 0.05974457
3  1e-01     1 0.12150000 0.06869350
4  1e+00     1 0.12550000 0.06163438
5  1e+01     1 0.12550000 0.06163438
6  1e-03     2 0.10083333 0.05691259
7  1e-02     2 0.11750000 0.06994376
8  1e-01     2 0.12550000 0.06163438
9  1e+00     2 0.12550000 0.06163438
10 1e+01     2 0.12550000 0.06163438
11 1e-03     3 0.12100000 0.07243643
12 1e-02     3 0.12550000 0.06163438
13 1e-01     3 0.12550000 0.06163438
14 1e+00     3 0.12550000 0.06163438
15 1e+01     3 0.12550000 0.06163438
16 1e-03     4 0.11750000 0.06994376
17 1e-02     4 0.12550000 0.06163438
18 1e-01     4 0.12550000 0.06163438
19 1e+00     4 0.12550000 0.06163438
20 1e+01     4 0.12550000 0.06163438


###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 10)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="radial", r
anges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),tune
control = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
   10     2

- best performance: 0.1171667

- Detailed performance results:
    cost gamma     error dispersion
1  1e-03     1 0.1251667 0.06603614
2  1e-02     1 0.1251667 0.06603614
3  1e-01     1 0.1251667 0.06603614
4  1e+00     1 0.1251667 0.07142919
5  1e+01     1 0.1213333 0.07566087
6  1e-03     2 0.1251667 0.06603614
7  1e-02     2 0.1251667 0.06603614
8  1e-01     2 0.1251667 0.06603614
9  1e+00     2 0.1211667 0.07285640
10 1e+01     2 0.1171667 0.07401639
11 1e-03     3 0.1251667 0.06603614
12 1e-02     3 0.1251667 0.06603614
13 1e-01     3 0.1251667 0.06603614
```

```
14 1e+00     3 0.1211667 0.07285640
15 1e+01     3 0.1171667 0.07401639
16 1e-03     4 0.1251667 0.06603614
17 1e-02     4 0.1251667 0.06603614
18 1e-01     4 0.1251667 0.06603614
19 1e+00     4 0.1211667 0.07285640
20 1e+01     4 0.1251667 0.07142919
```

```
###tuning (mencari best parameter dan error terkecil)
> tc <- tune.control(cross = 10)
> tuning <- tune(svm, LUNG_CANCER~.,data=train, kernel="sigmoid",
ranges=list(cost=c(0.001, 0.01, 0.1, 1, 10), gamma=c(1,2,3,4)),tun
econtrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
  cost gamma
 0.001     1

- best performance: 0.1251667

- Detailed performance results:
    cost gamma     error dispersion
1  1e-03     1 0.1251667 0.08294521
2  1e-02     1 0.1251667 0.08294521
3  1e-01     1 0.1291667 0.08363003
4  1e+00     1 0.1738333 0.07809479
5  1e+01     1 0.1860000 0.06535846
6  1e-03     2 0.1251667 0.08294521
7  1e-02     2 0.1251667 0.08294521
8  1e-01     2 0.1373333 0.07773888
9  1e+00     2 0.1738333 0.06200383
10 1e+01     2 0.1700000 0.05907204
11 1e-03     3 0.1251667 0.08294521
12 1e-02     3 0.1251667 0.08294521
13 1e-01     3 0.1373333 0.07773888
14 1e+00     3 0.1813333 0.09951053
15 1e+01     3 0.1978333 0.10158330
16 1e-03     4 0.1251667 0.08294521
17 1e-02     4 0.1251667 0.08294521
18 1e-01     4 0.1415000 0.07812008
19 1e+00     4 0.1740000 0.09617435
20 1e+01     4 0.1861667 0.08263056
```

Lampiran I. Pengujian Data *Testing* SVM

```
> #Pengujian 10 fold testing
> folds = createFolds(train$LUNG_CANCER, k = 10)
> cv = lapply(folds, function(x) {
```

```
+   training_fold = train[-x, ]
+   test_fold = train[x, ]
+   data.svm = svm(formula = LUNG_CANCER ~ .,
+                  data = training_fold,
+                  type = 'C-classification',
+                  kernel = 'linear',cost=1, gamma=1)
+   y_pred = predict(data.svm, newdata = test_fold[-16])
+   cm_test= table(test_fold$LUNG_CANCER, y_pred)
+   accuracy = (cm_test[1,1] + cm_test[2,2]) / (cm_test[1,1] + cm_
test[2,2] + cm_test[1,2] + cm_test[2,1])
+   return(accuracy)})
> accuracy = mean(as.numeric(cv))
> accuracy
[1] 0.9317949
```

cv	list [10]	List of length 10
Fold01	double [1]	0.84
Fold02	double [1]	0.88
Fold03	double [1]	0.9583333
Fold04	double [1]	1
Fold05	double [1]	0.9166667
Fold06	double [1]	0.92
Fold07	double [1]	0.96
Fold08	double [1]	0.8846154
Fold09	double [1]	1
Fold10	double [1]	0.9583333

```
> prediksi = predict(tuning$best.model,newdata = test[-16])
> table(test$LUNG_CANCER,prediksi)

prediksi        YES     NO
       YES      54      0
       NO       8       0

> prediksi
  3   6  11  19  20  30  32  36  42  43  45  58  59  60  62  69
76  78  87  89  96 102
YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES Y
ES YES YES YES YES YES
105 106 116 118 121 122 128 131 148 150 157 158 166 169 188 194 1
97 198 201 206 213 214
YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES Y
ES YES YES YES YES YES
```

```
218 222 225 227 229 231 239 244 248 262 266 276 287 288 289 291 3
00 301
YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES YES Y
ES YES
Levels: NO YES
> confusionMatrix(prediksi, test$LUNG_CANCER)
Confusion Matrix and Statistics

               Prediction
Reference      YES     NO
       YES     54      0
       NO      8       0

               Accuracy : 0.871
                 95% CI : (0.7615, 0.9426)
    No Information Rate : 0.871
    P-Value [Acc > NIR] : 0.59277

                  Kappa : 0

 Mcnemar's Test P-Value : 0.01333

            Sensitivity : 0.000
            Specificity : 1.000
         Pos Pred Value :   NaN
         Neg Pred Value : 0.871
             Prevalence : 0.129
         Detection Rate : 0.000
   Detection Prevalence : 0.000
      Balanced Accuracy : 0.500

       'Positive' Class : NO
```

Lampiran J. PCA Data

```
> library(e1071)
> library(caret)
> library(ggplot2)
> library(kernlab)
> #Memanggil data
> data=read.csv("E:\\skripsi\\data\\lung cancer.csv", sep=";")
> head(data)
> dat=data.frame(t(data))
> cancer<-data[1:309,1:15]
> lung<-data[1:309,16]
> cancer <- data.frame(lapply(cancer, as.numeric), stringsAsFacto
rs=FALSE)
> dordor=prcomp(cancer,scale. = TRUE)
> dordor
Standard deviations (1, .., p=15):
 [1] 1.6912209 1.3932831 1.2416131 1.1590014 1.0599099 0.9839575
0.9442036 0.9046532
 [9] 0.8562766 0.7953282 0.7579268 0.7246214 0.6562393 0.5754706
0.5338962

Rotation (n x k) = (15 x 15):
```

	PC1	PC2	PC3	PC4	PC5
GENDER	-0.32117615	0.30796295	-0.07892998	0.2617	4010 -0.170485514
AGE	-0.03518621	0.03748262	0.26187721	-0.1237	7342 -0.461205736
SMOKING	0.04525341	0.04437384	-0.20753428	0.5791	8597 0.201747443
YELLOW_FINGERS	0.37975679	0.11660228	0.31474395	0.1039	1255 -0.054111488
ANXIETY	0.40363758	0.22539826	0.13662371	0.2582	5160 -0.068687957
PEER_PRESSURE	0.30539340	0.07985054	0.29882717	-0.1147	1295 0.165534288
CHRONIC.DISEASE	0.08917655	0.04603290	-0.08369445	-0.5226	5007 0.468213885
FATIGUE	-0.10371553	-0.43013384	0.29918982	0.1520	7720 0.271342907
ALLERGY	-0.25224440	0.21718872	0.11006684	-0.1500	8914 0.365246278
WHEEZING	-0.24632359	0.09510497	0.45346950	-0.0827	8613 0.039636986
ALCOHOL.CONSUMING	-0.33376881	0.42437150	0.01653525	-0.0806	7680 -0.007876472
COUGHING	-0.28002159	-0.08139354	0.48438140	0.0205	1142 -0.201964572
SHORTNESS.OF.BREATH	-0.14061785	-0.44548319	0.18896056	0.2620	4949 0.233671626
SWALLOWING.DIFFICULTY	0.29021677	0.33203231	0.29551774	0.1058	9646 0.208028135
CHEST.PAIN	-0.23734263	0.30371272	0.05374244	0.2785	3327 0.340466194

	PC6	PC7	PC8	PC9	PC10
GENDER	0.08830342	-0.05062642	0.30297216	-0.2123	31843 -0.19378537
AGE	-0.57605445	0.51577533	0.22054448	0.1381	40989 0.04510178
SMOKING	-0.23719170	0.21018804	-0.38606435	0.4248	04599 0.22900888
YELLOW_FINGERS	-0.11831270	-0.31903216	-0.11742967	-0.3444	14478 0.13884606
ANXIETY	-0.24308988	-0.17193340	0.03074648	0.0038	63714 -0.32738423
PEER_PRESSURE	0.34114773	0.50570703	-0.11673231	-0.1765	01571 0.22290084
CHRONIC.DISEASE	-0.38632471	-0.16202411	0.22758797	0.1313	26870 0.18794761
FATIGUE	0.14337306	0.31238957	0.19606752	-0.0571	71337 -0.33822936
ALLERGY	-0.27254379	0.13816399	-0.52234881	-0.2550	06282 -0.41905095
WHEEZING	0.24786984	-0.21554098	-0.04250811	0.5876	22475 0.06442882
ALCOHOL.CONSUMING	0.01614958	0.02172900	0.01133855	0.0749	68407 -0.15270863

```
COUGHING                -0.09485516 -0.21866058 -0.30102597 -0.1430
31892  0.29717226
SHORTNESS.OF.BREATH     -0.30081506 -0.22699296  0.21382492 -0.0431
82978 -0.07328906
SWALLOWING.DIFFICULTY    0.08335968 -0.01445847  0.26448945  0.2600
90632 -0.22014326
CHEST.PAIN              -0.04911701  0.10366215  0.33293523 -0.2813
36572  0.48670396
                                PC11        PC12        PC13
PC14          PC15
GENDER                  -0.08089307  0.51663581 -0.47113187  0.0609
8255  0.1194611750
AGE                      0.12739994 -0.01894730 -0.05332774  0.0806
8678 -0.0847518831
SMOKING                 -0.09632094  0.25897220 -0.05375352 -0.0588
1028 -0.1080424113
YELLOW_FINGERS           0.24774117  0.31968261  0.07329203  0.1554
3827 -0.5184832959
ANXIETY                  0.01263227  0.04274956  0.33774038 -0.0512
7331  0.6170212771
PEER_PRESSURE           -0.32047225  0.16121843 -0.07206197  0.3389
4130  0.2348341028
CHRONIC.DISEASE         -0.12409479  0.39610642 -0.06094889 -0.1571
0593  0.0998680656
FATIGUE                  0.15228236  0.31952824  0.25638539 -0.3729
3291 -0.1166410147
ALLERGY                  0.21094833 -0.13947966 -0.20613354  0.0846
4244  0.0263965030
WHEEZING                 0.38232802  0.12502552 -0.01135021  0.2708
7676  0.1474985339
ALCOHOL.CONSUMING       -0.43669117  0.08350305  0.60689175  0.1379
9300 -0.2946688074
COUGHING                -0.36369759 -0.01381139 -0.07686432 -0.4674
1213  0.1615000934
SHORTNESS.OF.BREATH     -0.32824781 -0.16786861 -0.06945119  0.5357
8858 -0.0009302669
SWALLOWING.DIFFICULTY -0.22710005 -0.36288402 -0.35546576 -0.2679
5525 -0.2984465766
CHEST.PAIN               0.30356317 -0.27300771  0.17376728 -0.0628
9303  0.1190427758

> prediksi=predict(dordor,newdata=cancer)
> head(prediksi)
            PC1        PC2        PC3        PC4        PC5        PC6
[1,] -0.1640482  0.2091336 -2.0141314 -0.4959408 -0.9768023  0.8331597
[2,] -0.4608205 -0.6030873  1.1885498 -0.8852348  2.5092921 -0.2292342
[3,] -1.4312585 -1.5081433 -0.9940443  0.5554339 -0.6932702 -1.4060235
[4,]  1.8866105  1.5110171  1.3598099 -1.8560093 -1.1516581  0.2684996
[5,] -0.4893499 -1.3666298 -0.1268886  1.0043726 -2.3230860  1.4398996
[6,] -1.2083366 -1.3017439 -0.7301279  1.8282317  0.3376601  1.4911086
            PC7         PC8         PC9        PC10        PC11        PC12
[1,] -1.8158109  0.94275728  0.91713223 -0.10352720  0.6429572 -0.2187327
[2,] -0.4640190 -0.04422908  0.49568012  0.25561260 -1.4304833 -0.7955466
[3,] -0.7812819  1.01434095 -0.88137611  0.19149929 -0.6489430  0.2792562
[4,] -0.9945928  0.71826315  0.28049606  0.01158793  0.3275796 -0.2736686
[5,] -0.4006732  0.17898050 -0.74678204  0.59150627 -1.0505852  0.9155850
[6,]  0.6717752 -0.39958120 -0.05226049  1.18801804 -0.2558346 -0.0540877
           PC13
```

```
[1,]  0.2595164
[2,]  0.6065733
[3,] -1.1447042
[4,]  0.8184049
[5,]  0.4105099
[6,]  0.5355413
> new=data.frame(prediksi,lung)
> date=data.frame(new$PC1,new$PC2,lung)
> head(new)
         PC1        PC2        PC3        PC4        PC5        PC6        PC7
1 -0.1640482  0.2091336 -2.0141314 -0.4959408 -0.9768023  0.8331597 -1.8158109
2 -0.4608205 -0.6030873  1.1885498 -0.8852348  2.5092921 -0.2292342 -0.4640190
3 -1.4312585 -1.5081433 -0.9940443  0.5554339 -0.6932702 -1.4060235 -0.7812819
4  1.8866105  1.5110171  1.3598099 -1.8560093 -1.1516581  0.2684996 -0.9945928
5 -0.4893499 -1.3666298 -0.1268886  1.0043726 -2.3230860  1.4398996 -0.4006732
6 -1.2083366 -1.3017439 -0.7301279  1.8282317  0.3376601  1.4911086  0.6717752
          PC8         PC9        PC10       PC11        PC12        PC13 lung
1  0.94275728  0.91713223 -0.10352720  0.6429572 -0.2187327  0.2595164  YES
2 -0.04422908  0.49568012  0.25561260 -1.4304833 -0.7955466  0.6065733  YES
3  1.01434095 -0.88137611  0.19149929 -0.6489430  0.2792562 -1.1447042   NO
4  0.71826315  0.28049606  0.01158793  0.3275796 -0.2736686  0.8184049   NO
5  0.17898050 -0.74678204  0.59150627 -1.0505852  0.9155850  0.4105099   NO
6 -0.39958120 -0.05226049  1.18801804 -0.2558346 -0.0540877  0.5355413  YES
```

Lampiran K. Proses *Tuning Random Forest* Data PCA

```
### Tuning Mtry
> model <- train(lung ~ ., # Survived is a function of the variab
les we decided to include
+                data = training, # Use the train data frame as t
he training data
+                method = 'rf',# Use the 'random forest' algorith
m
+                trControl = trainControl(method = 'cv', # Use cr
oss-validation
+                                         number = 10)) # Use 5 f
olds for cross-validation
note: only 1 unique complexity parameters in default grid. Trunca
ting the grid to 1 .

> model
Random Forest

263 samples
  2 predictor
  2 classes: 'NO', 'YES'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 236, 236, 237, 237, 237, 237, ...
Resampling results:

  Accuracy   Kappa
  0.9086895  0.4645481

Tuning parameter 'mtry' was held constant at a value of 2

### Tuning Ntree
```

```
> control <- trainControl(method="repeatedcv", number=10, repeats
=3, search="grid")
> tunegrid <- expand.grid(.mtry=c(sqrt(ncol(x))))
> modellist <- list()
> for (ntree in c(25,50,100,500,1000)) {
+   fit <- train(lung~., data=training, method="rf",  tuneGrid=tu
negrid, trControl=control, ntree=ntree)
+   key <- toString(ntree)
+   modellist[[key]] <- fit
+ }
> # compare results
> results <- resamples(modellist)
> summary(results)

Call:
summary.resamples(object = results)

Models: 25, 50, 100, 500, 1000
Number of resamples: 30



Accuracy
         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. N
A's
25   0.8076923 0.8600427 0.9230769 0.9098765 0.9615385 1.000000
0
50   0.8518519 0.8846154 0.9230769 0.9137702 0.9259259 0.962963
0
100  0.8148148 0.8846154 0.9230769 0.9073599 0.9252137 1.000000
0
500  0.7692308 0.8846154 0.9230769 0.9111586 0.9259259 0.962963
0
1000 0.7692308 0.8856838 0.9230769 0.9150522 0.9259259 1.000000
0

Kappa
            Min.   1st Qu.    Median      Mean   3rd Qu.      Max
. NA's
25   -0.10169492 0.2463768 0.4693878 0.4629148 0.7796610 1.000000
0    0
50   -0.05882353 0.3389831 0.4699880 0.4552312 0.6250000 0.835443
0    0
100  -0.05882353 0.3389831 0.4693878 0.4439121 0.5946640 1.000000
0    0
500  -0.13043478 0.3715842 0.4884587 0.4713391 0.6245471 0.780487
8    0
1000 -0.13043478 0.3734445 0.4705882 0.4740505 0.6250000 1.000000
0    0
```

Lampiran L. Klasifikasi dan Visualisasi Data Training *Random Forest* Data PCA

###Klasifikasi *Random Forest* Proses *Training*

```
> rf1<- randomForest(lung ~ ., data = training, mtry=2, ntree=100
0)
```

```
> rf1

Call:
 randomForest(formula = lung ~ ., data = training, mtry = 2, ntre
e = 1000)
               Type of random forest: classification
                     Number of trees: 1000
No. of variables tried at each split: 2

        OOB estimate of  error rate: 7.6%
Confusion matrix:

        YES     NO      class.error
   YES  229     4       0.01716738
   NO   16      14      0.53333333

### perform pca and extract scores
library(ellipse)
pcaOutput <- prcomp(as.matrix(cancer), scale = TRUE, center = TRU
E)
pcaOutput2 <- as.data.frame(pcaOutput$x)


### define groups for plotting
pcaOutput2$groups <- rf1$lung

centroids <- aggregate(cbind(PC1,PC2)~lung, pcaOutput2, mean)

conf.rgn  <- do.call(rbind, lapply(unique(pcaOutput2$groups), fun
ction(t)
  data.frame(groups = as.character(t),
  ellipse(cov(pcaOutput2[pcaOutput2$groups == t, 1:2]),
  centre = as.matrix(centroids[centroids$groups == t, 2:3]),
  level = 0.95),
  stringsAsFactors = FALSE)))

ggplot(data = pcaOutput2, aes(x = PC1, y = PC2, group = lung, col
or =lung)) + geom_point(size = 2, alpha = 0.6) + labs(color = "Lu
ng Cancer",fill = "red")
```

Lampiran M. Klasifikasi dan Visualisasi Data *Testing Random Forest* Data PCA

###Klasifikasi *Random Forest* Proses *Testing*

```
> rf1<- randomForest(lung ~ ., data = validation, mtry=2, ntree=1
000)

### Predicting on Validation set
> predValid <- predict(rf1, validation, type = "class")
### Checking classification accuracy
> mean(predValid == validation$lung)
[1] 1
> table(predValid,validation$lung)

preValid        YES     NO
       YES      37      0
       NO       0       9
```

```
> confusionMatrix(predValid, validation$lung)
Confusion Matrix and Statistics

               Prediction
Reference      YES     NO
       YES     37      0
       NO      0       9

               Accuracy : 1
                 95% CI : (0.9229, 1)
    No Information Rate : 0.8043
    P-Value [Acc > NIR] : 4.471e-05

                  Kappa : 1

 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0000
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 1.0000
             Prevalence : 0.1957
         Detection Rate : 0.1957
   Detection Prevalence : 0.1957
      Balanced Accuracy : 1.0000

       'Positive' Class : NO
```

Lampiran N1. Klasifikasi SVM untuk Visualisasi dengan Kernel *Linear*

```
###partisi data
set.seed(22)
split = createDataPartition(data$LUNG_CANCER, p=0.80, list=FALSE)
train = data[split,]
test  = data[-split,]
summary(train)
summary(test)
date[,"train"]<-ifelse(runif(nrow(date))<0.80, 1, 0)
###menampilkan 6 data setelah PCA teratas
head(date)
###menampilkan 6 data setelah PCA terbawah
tail(date)
###data training yang digunakan untuk proses SVM
training<-date[date$train == 1,]
training
summary(training)
###data testing yang digunakan untuk proses SVM
validation<-date[date$train == 0, ]
validation
summary(validation)
###data training untuk svm
trainColNum<-grep("train",names(date))
training<-training[,-trainColNum]
training
###data testing untuk svm
validation<-validation[,-trainColNum]
```

```
validation
###klasifikasi SVM Kernel Linear
> svmfit1 <- svm(lung~ ., data = training, kernel = "linear",
+                cost = 1)
> summary(svmfit1)

Call:
svm(formula = lung ~ ., data = training, kernel = "linear", cost
= 1)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  65

 ( 35 30 )

Number of Classes:  2

Levels:
 NO YES

###confusion matrix data training
> pred_train<-predict(svmfit1,training)
> table(pred_train,training$lung)

pred_train     YES     NO
       YES     233     0
       NO      30      0
> mean(pred_train == training$lung)
[1] 0.8859316
> confusionMatrix(training$lung, pred_train)

Confusion Matrix and Statistics
               Prediction
Reference      YES     NO
       YES     233     0
       NO      30      0


               Accuracy : 0.8859
                 95% CI : (0.8412, 0.9217)
    No Information Rate : 1
    P-Value [Acc > NIR] : 1

                  Kappa : 0

 Mcnemar's Test P-Value : 1.192e-07

            Sensitivity :     NA
            Specificity : 0.8859
         Pos Pred Value :     NA
         Neg Pred Value :     NA
             Prevalence : 0.0000
         Detection Rate : 0.0000
```

```
   Detection Prevalence : 0.1141
      Balanced Accuracy :     NA

       'Positive' Class : NO

###plot proses training
> plot(svmfit1, training)


###confusion matrix data testing
> set.seed(22)
> output.tune <- tune(svm, lung ~ ., data = validation, kernel =
+                       "linear", ranges = list(cost = c(0.001, 0
.01, 0.1, 1, 10), gamma = c(1,2,3,4)), tunecontrol = tc)
> summary(output.tune)

Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
   10     1

- best performance: 0.12

- Detailed performance results:
    cost gamma error dispersion
1  1e-03     1 0.195  0.1921371
2  1e-02     1 0.195  0.1921371
3  1e-01     1 0.195  0.1921371
4  1e+00     1 0.165  0.1856670
5  1e+01     1 0.120  0.1932184
6  1e-03     2 0.195  0.1921371
7  1e-02     2 0.195  0.1921371
8  1e-01     2 0.195  0.1921371
9  1e+00     2 0.190  0.2131770
10 1e+01     2 0.185  0.2000694
11 1e-03     3 0.195  0.1921371
12 1e-02     3 0.195  0.1921371
13 1e-01     3 0.195  0.1921371
14 1e+00     3 0.190  0.2131770
15 1e+01     3 0.250  0.2054805
16 1e-03     4 0.195  0.1921371
17 1e-02     4 0.195  0.1921371
18 1e-01     4 0.195  0.1921371
19 1e+00     4 0.215  0.2028272
20 1e+01     4 0.250  0.2054805

> prediksi = predict(output.tune$best.model,newdata = validation)
> table(validation$lung,prediksi)

prediksi       YES     NO
       YES     37      0
       NO      3       6
> prediksi
```

```
  7  20  21  26  31  33  41  46  59  66  70  88 101 107 114 116 1
18 136 151 154 156 165
YES  NO YES YES  NO YES YES YES YES YES YES YES YES YES YES YES Y
ES YES YES YES YES YES
166 170 172 179 184 187 210 211 218 235 242 243 245 251 263 266 2
69 273 281 288 293 296
YES YES YES YES  NO YES YES YES YES YES YES YES YES YES  NO YES Y
ES  NO  NO YES YES YES
301 306
YES YES
Levels: NO YES
> confusionMatrix(prediksi, validation$lung)
Confusion Matrix and Statistics
                Prediction
Reference       YES     NO
       YES      37      0
       NO       3       6


               Accuracy : 0.9348
                 95% CI : (0.821, 0.9863)
    No Information Rate : 0.8043
    P-Value [Acc > NIR] : 0.01305

                  Kappa : 0.7629

 Mcnemar's Test P-Value : 0.24821

            Sensitivity : 0.6667
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.9250
             Prevalence : 0.1957
         Detection Rate : 0.1304
   Detection Prevalence : 0.1304
      Balanced Accuracy : 0.8333

       'Positive' Class : NO

### plot proses testing
> plot(svmfit1, validation)
```

Lampiran N2. Klasifikasi SVM untuk Visualisasi dengan Kernel *Polynomial*

```
### Klasifikasi SVM Kernel Polynomial
> svmfit1 <- svm(lung~ ., data = training, kernel = "polynomial",
+                cost = 1)
> summary(svmfit1)
Call:
svm(formula = lung ~ ., data = training, kernel = "polynomial",
    cost = 1)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  1
```

```
     degree:  3
     coef.0:  0

Number of Support Vectors:  60

 ( 30 30 )

Number of Classes:  2

Levels:
 NO YES

> #confusion matrix data training
> pred_train<-predict(svmfit1,training)
> table(pred_train,training$lung)

pred_train     YES     NO
       YES     231     2
       NO      23      7
> mean(pred_train == training$lung)
[1] 0.904943
> confusionMatrix(training$lung, pred_train)
Confusion Matrix and Statistics
                Prediction
Reference       YES     NO
       YES      231     2
       NO       23      7

               Accuracy : 0.9049
                 95% CI : (0.8629, 0.9375)
    No Information Rate : 0.9658
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3234

 Mcnemar's Test P-Value : 6.334e-05

            Sensitivity : 0.77778
            Specificity : 0.90945
         Pos Pred Value : 0.23333
         Neg Pred Value : 0.99142
             Prevalence : 0.03422
         Detection Rate : 0.02662
   Detection Prevalence : 0.11407
      Balanced Accuracy : 0.84361

       'Positive' Class : NO

### plot proses training
> plot(svmfit1, training)

###confusion matrix data testing
> set.seed(22)
> output.tune <- tune(svm, lung ~ ., data = validation, kernel =
+                       "polynomial", ranges = list(cost = c(0.001
, 0.01, 0.1, 1, 10), gamma = c(1,2,3,4)), tunecontrol = tc)
> summary(output.tune)
```

```
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
  0.1     1

- best performance: 0.155

- Detailed performance results:
    cost gamma error dispersion
1  1e-03     1 0.200  0.1394433
2  1e-02     1 0.180  0.1531158
3  1e-01     1 0.155  0.1091635
4  1e+00     1 0.180  0.1531158
5  1e+01     1 0.205  0.2178812
6  1e-03     2 0.180  0.1531158
7  1e-02     2 0.155  0.1091635
8  1e-01     2 0.180  0.1531158
9  1e+00     2 0.205  0.2178812
10 1e+01     2 0.180  0.1531158
11 1e-03     3 0.155  0.1091635
12 1e-02     3 0.155  0.1091635
13 1e-01     3 0.180  0.1531158
14 1e+00     3 0.180  0.1531158
15 1e+01     3 0.180  0.1531158
16 1e-03     4 0.155  0.1091635
17 1e-02     4 0.200  0.1683251
18 1e-01     4 0.205  0.2178812
19 1e+00     4 0.180  0.1531158
20 1e+01     4 0.180  0.1531158

> pred_test<-predict(svmfit1,validation)
> table(pred_test,validation$lung)

          Prediction
Reference      YES     NO
       YES     36      1
       NO      6       3
> mean(pred_test == validation$lung)
[1] 0.8478261
> confusionMatrix(validation$lung, pred_test)
Confusion Matrix and Statistics

          Prediction
Reference      YES     NO
       YES     36      1
       NO      6       3

               Accuracy : 0.8478
                 95% CI : (0.7113, 0.9366)
    No Information Rate : 0.913
    P-Value [Acc > NIR] : 0.9569

                  Kappa : 0.3878
```

```
 Mcnemar's Test P-Value : 0.1306

            Sensitivity : 0.75000
            Specificity : 0.85714
         Pos Pred Value : 0.33333
         Neg Pred Value : 0.97297
             Prevalence : 0.08696
         Detection Rate : 0.06522
   Detection Prevalence : 0.19565
      Balanced Accuracy : 0.80357

       'Positive' Class : NO

###plot proses testing
> plot(svmfit1, validation)
```

Lampiran N3. Klasifikasi SVM untuk Visualisasi dengan Kernel *Radial*

```
###Klasifikasi SVM Kernel Radial
> svmfit1 <- svm(lung~ ., data = training, kernel = "radial",
+                cost = 1)
> summary(svmfit1)
Call:
svm(formula = lung ~ ., data = training, kernel = "radial", cost
= 1)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  72

 ( 42 30 )

Number of Classes:  2

Levels:
 NO YES

###confusion matrix data training
> pred_train<-predict(svmfit1,training)
> table(pred_train,training$lung)

pred_train      YES     NO
       YES      231     2
       NO       23      7
> mean(pred_train == training$lung)
[1] 0.904943
> confusionMatrix(training$lung, pred_train)
Confusion Matrix and Statistics

          Prediction
Reference       YES     NO
       YES      231     2
       NO       23      7
```

```
               Accuracy : 0.9049
                 95% CI : (0.8629, 0.9375)
    No Information Rate : 0.9658
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3234

 Mcnemar's Test P-Value : 6.334e-05

            Sensitivity : 0.77778
            Specificity : 0.90945
         Pos Pred Value : 0.23333
         Neg Pred Value : 0.99142
             Prevalence : 0.03422
         Detection Rate : 0.02662
   Detection Prevalence : 0.11407
      Balanced Accuracy : 0.84361

       'Positive' Class : NO
###plot proses training
> plot(svmfit1, training)

###confusion matrix data testing
> set.seed(22)
> output.tune <- tune(svm, lung ~ ., data = validation, kernel =
+                       "radial", ranges = list(cost = c(0.001, 0
.01, 0.1, 1, 10), gamma = c(1,2,3,4)), tunecontrol = tc)
> summary(output.tune)

Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
 cost gamma
   10     1

- best performance: 0.125

- Detailed performance results:
    cost gamma error dispersion
1  1e-03     1 0.180  0.2394438
2  1e-02     1 0.180  0.2394438
3  1e-01     1 0.180  0.2394438
4  1e+00     1 0.145  0.1674150
5  1e+01     1 0.125  0.1438556
6  1e-03     2 0.180  0.2394438
7  1e-02     2 0.180  0.2394438
8  1e-01     2 0.180  0.2394438
9  1e+00     2 0.145  0.1674150
10 1e+01     2 0.230  0.2311805
11 1e-03     3 0.180  0.2394438
12 1e-02     3 0.180  0.2394438
13 1e-01     3 0.180  0.2394438
14 1e+00     3 0.145  0.1674150
```

```
15 1e+01     3 0.255  0.2166026
16 1e-03     4 0.180  0.2394438
17 1e-02     4 0.180  0.2394438
18 1e-01     4 0.180  0.2394438
19 1e+00     4 0.165  0.2082333
20 1e+01     4 0.250  0.2549510

> pred_test<-predict(svmfit1,validation)
> table(pred_test,validation$lung)

pred_test       YES     NO
       YES      36      1
       NO       5       4
> mean(pred_test == validation$lung)
[1] 0.8695652
> confusionMatrix(validation$lung, pred_test)
Confusion Matrix and Statistics

          Prediction
Reference       YES     NO
       YES      36      1
       NO       5       4


               Accuracy : 0.8696
                 95% CI : (0.7374, 0.9506)
    No Information Rate : 0.8913
    P-Value [Acc > NIR] : 0.7710

                  Kappa : 0.5018

 Mcnemar's Test P-Value : 0.2207

            Sensitivity : 0.80000
            Specificity : 0.87805
         Pos Pred Value : 0.44444
         Neg Pred Value : 0.97297
             Prevalence : 0.10870
         Detection Rate : 0.08696
   Detection Prevalence : 0.19565
      Balanced Accuracy : 0.83902

       'Positive' Class : NO

###plot proses testing
> plot(svmfit1, validation)
```

Lampiran N4. Klasifikasi SVM untuk Visualisasi dengan Kernel *Sigmoid*

```
###Klasifikasi SVM Kernel Sigmoid
> svmfit1 <- svm(lung~ ., data = training, kernel = "sigmoid",
+                cost = 1)
> summary(svmfit1)
Call:
svm(formula = lung ~ ., data = training, kernel = "sigmoid", cost
= 1)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
```

```
     coef.0:  0
Number of Support Vectors:  62

 ( 32 30 )

Number of Classes:  2

Levels:
 NO YES

###confusion matrix data training
> pred_train<-predict(svmfit1,training)
> table(pred_train,training$lung)

pred_train     YES     NO
       YES     37      0
       NO      9       0
> mean(pred_train == training$lung)
[1] 0.8250951
> confusionMatrix(training$lung, pred_train)
Confusion Matrix and Statistics

          Prediction
Reference       YES     NO
       YES      217     16
       NO       30      0


               Accuracy : 0.8251
                 95% CI : (0.7737, 0.869)
    No Information Rate : 0.9392
    P-Value [Acc > NIR] : 1.00000

                  Kappa : -0.0862

 Mcnemar's Test P-Value : 0.05527

            Sensitivity : 0.00000
            Specificity : 0.87854
         Pos Pred Value : 0.00000
         Neg Pred Value : 0.93133
             Prevalence : 0.06084
         Detection Rate : 0.00000
   Detection Prevalence : 0.11407
      Balanced Accuracy : 0.43927

       'Positive' Class : NO

###plot proses training
> plot(svmfit1, training)

###confusion matrix data testing
> set.seed(22)
> output.tune <- tune(svm, lung ~ ., data = validation, kernel =
+                       "sigmoid", ranges = list(cost = c(0.001,
0.01, 0.1, 1, 10), gamma = c(1,2,3,4)), tunecontrol = tc)
> summary(output.tune)
```

```
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:
  cost gamma
 0.001     1

- best performance: 0.195

- Detailed performance results:
    cost gamma error dispersion
1  1e-03     1 0.195  0.1921371
2  1e-02     1 0.195  0.1921371
3  1e-01     1 0.195  0.1921371
4  1e+00     1 0.260  0.1696401
5  1e+01     1 0.445  0.1921371
6  1e-03     2 0.195  0.1921371
7  1e-02     2 0.195  0.1921371
8  1e-01     2 0.195  0.1921371
9  1e+00     2 0.235  0.1886355
10 1e+01     2 0.400  0.1900292
11 1e-03     3 0.195  0.1921371
12 1e-02     3 0.195  0.1921371
13 1e-01     3 0.195  0.1921371
14 1e+00     3 0.215  0.1795828
15 1e+01     3 0.350  0.1490712
16 1e-03     4 0.195  0.1921371
17 1e-02     4 0.195  0.1921371
18 1e-01     4 0.195  0.1921371
19 1e+00     4 0.215  0.1795828
20 1e+01     4 0.330  0.2162817

> pred_test<-predict(svmfit1,validation)
> table(pred_test,validation$lung)

pred_test       YES     NO
       YES      37      0
       NO       9       0
> mean(pred_test == validation$lung)
[1] 0.8043478
> confusionMatrix(validation$lung, pred_test)
Confusion Matrix and Statistics

          Prediction
Reference       YES     NO
       YES      37      0
       NO       9       0

               Accuracy : 0.8043
                 95% CI : (0.6609, 0.9064)
    No Information Rate : 1
    P-Value [Acc > NIR] : 1.000000

                  Kappa : 0
```

```
 Mcnemar's Test P-Value : 0.007661

            Sensitivity :     NA
            Specificity : 0.8043
         Pos Pred Value :     NA
         Neg Pred Value :     NA
             Prevalence : 0.0000
         Detection Rate : 0.0000
   Detection Prevalence : 0.1957
      Balanced Accuracy :     NA

       'Positive' Class : NO

###plot proses testing
> plot(svmfit1, validation)
```