KLASIFIKASI EKSPRESI GENETIKA PADA KANKER PROSTAT MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

SKRIPSI

Oleh

Salik Alfi Komarudin

NIM 161810101001

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2020

KLASIFIKASI EKSPRESI GENETIKA PADA KANKER PROSTAT MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

SKRIPSI

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

Oleh
Salik Alfi Komarudin
NIM 161810101001

JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS JEMBER
2020

PERSEMBAHAN

Puji syukur dengan menyebut nama Allah SWT yang Maha Pengasih dan Maha Penyayang dan sholawat serta salam senantiasa tercurahkan kepada junjungan Nabi Muhammad SAW sehingga terselesaikanlah skripsi ini dan saya persembahkan untuk:

1. Keluarga saya yang tercinta, Bapak Djoko Heri Purnomo, Ibu Anis Siswati, kakak saya Siti Khuri, adik saya Intan Nurfadillah dan seluruh keluarga yang telah memberikan do'a, semangat dan dukungan terhadap saya;
2. Seluruh jajaran guru dan dosen yang telah membimbing dan memberikan ilmunya kepada saya mulai dari TK Dharma Wanita P8 Sidowayah, SDN Sidowayah, SMPN 1 Bangil, SMAN 1 Bangil dan Jurusan Matematika FMIPA Universitas Jember dengan penuh kebaikan, kesabaran dan kasih sayang;
3. Teman-teman dari MISDIRECTON'16 yang memberikan semangat serta dukungan selama masa perkuliahan.

MOTTO

"Jangan sengaja pergi agar dicari, jangan sengaja lari agar dikejar.
Karena berjuang tak sebercanda itu."
(Sujiwo Tejo)

"Tuhan tidak menuntut kita untuk sukses.
Tuhan hanya menyuruh kita berjuang tanpa henti."
(Emha Ainun Nadjib)

"Jika do'a bukan sebuah permintaan, setidaknya itu adalah sebuah pengakuan kelemahan diri manusia di hadapan Tuhannya."
(Pidi Baiq)

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Salik Alfi Komarudin

NIM : 161810101001

menyatakan dengan sesungguhnya bahwa skripsi yang berjudul "Klasifikasi Ekspresi Genetika Pada Kanker Prostat Menggunakan Metode *Support Vector Machine*" 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, Januari 2020

Yang menyatakan,

Salik Alfi Komarudin
NIM 161810101001

SKRIPSI

KLASIFIKASI EKSPRESI GENETIKA PADA KANKER PROSTAT MENGGUNAKAN METODE *SUPPORT VECTOR MACHINE*

Oleh

Salik Alfi Komarudin
NIM 161810101001

Pembimbing

Dosen Pembimbing Utama : Dr. Alfian Futuhul Hadi, S.Si., M.Si.
Dosen Pembimbing Anggota : Abduh Riski, S.Si., M.Si.

PENGESAHAN

Skripsi berjudul "Klasifikasi Ekspresi Genetika Pada Kanker Prostat Menggunakan Metode *Support Vector Machine*" karya Salik Alfi Komarudin telah diuji dan disahkan pada:

hari, tanggal :

tempat : Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Jember

Tim Penguji:

Ketua,	Anggota I,
Dr. Alfian Futuhul Hadi, S.Si., M.Si. NIP. 197407192000121001	Abduh Riski, S.Si., M.Si. NIP. 199004062015041001
Anggota II,	Anggota III,
Ahmad Kamsyakawuni, S.Si., M.Kom. NIP. 197211291998011001	Dian Anggraeni, S.Si., M.Si. NIP. 198202162006042002

Mengesahkan
Dekan,

Drs. Achmad Sjaifullah, M.Sc., Ph.D.
NIP. 196102041987111001

RINGKASAN

Klasifikasi Ekspresi Genetika Pada Kanker Prostat Menggunakan Metode *Support Vector Machine*; Salik Alfi Komarudin, 161810101001; 2020; 47 halaman; Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Jember.

Kanker prostat telah lama menjadi perhatian ahli genetika manusia dalam penelitian kesehatan. Namun, penjelasan tentang penyebab utama kanker prostat tidak dapat diketahui secara metabolik-biologis, kecuali yang paling umum adalah faktor keturunan. Penjelasan risiko tertular kanker prostat dicari melalui penjelasan genetik sel-sel kanker prostat dan sel-sel prostat yang sehat dari sekuensing DNA dalam bentuk data *microarray* atau dalam bentuk nilai Gleason.

Data genetik sel kanker memiliki dimensi tinggi di mana jumlah variabel yang diamati jauh lebih banyak daripada individu yang diamati. Itu membuat teknik klasifikasi multivariat biasa gagal menangani data ini karena matriks singularitas. Selain itu, pengamatan jumlah pasien terkena kanker sangat sedikit karena jumlah mereka jarang ditemukan. Dengan dua fakta ini, maka dalam tulisan ini akan digunakan pendekatan *machine learning* untuk mempelajari klasifikasi, yaitu SVM. Karena SVM secara matematis memiliki solusi trivial, penelitian ini bertujuan untuk menyediakan fungsi pembagian yang akurat untuk membedakan sel-sel kanker prostat dari sel-sel prostat yang sehat.

Konsep SVM menjelaskan bagaimana upaya sederhana untuk menemukan fungsi pemisah terbaik (*hyperplane*) dari beberapa garis pemisah alternatif yang mungkin terjadi dalam suatu kasus. Fungsi pemisah terbaik adalah untuk menemukan nilai optimal dari batas *margin* untuk memisahkan setiap kelas dengan *hyperplane* antara kelas positif dan kelas negatif. Pada 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.

Sampel data yang digunakan pada penilitian ini terdiri dari 102 orang dengan 2135 variabel genetik yang kemudian dibagi menjadi data *training* dan

data *testing*. Data tersebut dibagi menjadi 75:25 dengan proporsi sama tiap-tiap kelas klasifikasi. Pengujian data *training* dan data *testing* dilakukan menggunakan metode *k-fold cross validation* dengan pembagiannya sebesar 5 *fold* dan 10 *fold*.

Hasil pengujian pada data training menghasilkan tingkat akurasi sebesar 100% untuk fungsi kernel *linear*, 83,11% untuk fungsi kernel *polynomial*, 94,8% untuk fungsi kernel *radial*, dan 92,2% untuk fungsi kernel *sigmoid*. Kemudian dilakukan *tuning* parameter untuk mencari parameter terbaik dan nilai *error* terkecil dari setiap fungsi kernel yang nantinya akan digunakan sebagai fungsi kernel pengujian data *testing*. *Tuning* parameter yang dilakukan berupa parameter *cost* yang bernilai 0.001, 0.01, 0.1, 1, 10, 100 terhadap setiap fungsi kernel menggunakan metode 5 *fold* dan 10 *fold cross validation.*

Dari hasil pengujian dan proses *tuning* didapatkan bahwa kernel *linear* merupakan fungsi kernel terbaik dibandingkan dengan ketiga kernel lainnya, ditandai dengan performa klasifikasinya yang lebih besar diantara kernel lain. Kernel *linear* juga memiliki nilai *error* terkecil pada 5 *fold* yang sudah dilakukan dengan *cost* parameternya sebesar 0,001. Pengujian data *testing* akan digunakan fungsi kernel *linear* dengan metode 5 *fold cross validation.*

Hasil pengujian data *testing* didapatkan akurasi sebesar 92%. Klasifikasi yang dihasilkan menyatakan bahwa terdapat 23 data terklasifikasikan secara benar dan 2 data yang mengalami kesalahan klasifikasi. Data tersebut diantaranya adalah 1 data yang seharusnya tergolong kedalam kelas normal tetapi terklasifikasikan kedalam kelas tumor, dan 1 data yang seharusnya termasuk kedalam kelas tumor tetapi terklasifikasikan kedalam kelas normal.

PRAKATA

Puji syukur kehadirat Allah SWT atas segala rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Ekspresi Genetika Pada Kanker Prostat Menggunakan Metode *Support Vector Machine*". 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. Dr. Alfian Futuhul Hadi, S.Si., M.Si. selaku Dosen Pembimbing Utama dan Abduh Riski, S.Si., M.Si. selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, tenaga, pikiran, dan perhatian dalam penulisan skripsi ini;
2. Ahmad Kamsyakawuni, S.Si., M.Kom. dan Dian Anggraeni, S.Si., M.Si. selaku Dosen Penguji yang telah memberikan kritik dan saran yang membangun demi kesempurnaan skripsi ini;
3. Prof. Drs. Kusno, DEA., Ph.D. dan Dr. Kristiana Wijaya, 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, Januari 2020

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

DAFTAR LAMPIRAN

Halaman

BAB 1. PENDAHULUAN

1.1 Latar Belakang

Kanker prostat telah lama menjadi perhatian ahli genetika manusia dalam penelitian kesehatan. Selain itu, pengamatan mengenai kanker juga jarang ditemukan karena jumlah pasien yang terbilang sedikit. Namun, penjelasan tentang penyebab utama kanker prostat tidak dapat diketahui secara pasti, kecuali yang paling umum adalah faktor keturunan. Ada beberapa faktor yang dapat mempengaruhi terjangkitnya kanker prostat. Kemudian penjelasan mengenai risiko kanker prostat dapat dicari dari salah satu faktor penyebabnya yaitu mutasi ekspresi genetika sel prostat.

Genetika sel prostat dapat dianalisa melalui sel prostat yang sehat dan sel prostat berisiko kanker dari sekuensing DNA yang datanya berupa *microarray* dalam bentuk nilai Gleason. Dari nilai Gleason dapat dibedakan antara sel risiko kanker dan sel prostat normal. Dataset *microarray* merupakan data yang dimana jumlah variabel pengamatan lebih banyak dari jumlah individu pengamatan. Dengan demikian, teknik klasifikasi multivariat biasa tidak dapat diterapkan pada bentuk data seperti ini karena terdapat matriks singularitas. Dalam hal ini, diperlukan pendekatan menggunakan *machine learning* untuk klasifikasi nilai Gleason pada *microarray* dataset.

Klasifikasi menggunakan *machine learning* dilakukan ketika data yang digunakan dengan skala besar, dan dapat juga digunakan untuk data dengan jumlah variabelnya lebih banyak dari jumlah pengamatan. Ada beberapa metode yang dapat digunakan untuk menentukan kasus klasifikasi, salah satunya *Support Vector Machine* (SVM). SVM adalah salah satu metode yang akhir-akhir ini mendapat perhatian lebih dari para peneliti dalam kasus klasifikasi karena metode ini memberika kemampuan generalisasi yang tinggi daripada metode lainnya.

Pratama (2018) membahas mengenai metode SVM untuk memprediksi ketepatan waktu kelulusan mahasiswa dengan akurasinya sebesar 80,55%. Damanik (2015) membandingkan metode RLB (Regresi Logistik Biner) dengan SVM dengan ketepatan klasifikasinya 70% untuk RLB dan 90% untuk metode

SVM. Andari (2013) menggunakan SVM untuk klasifikasi kanker payudara, menunjukkan bahwa penggunaan SSVM (*Smooth Support Vector Machine*) dengan akurasi 99,63% lebih baik dibandingkan dengan metode MARS (*Multivariate Adaptive Regression Splines*) dengan akurasi sebesar 95,88%.

Konsep SVM menjelaskan bagaimana upaya sederhana untuk menemukan fungsi pemisah terbaik (*hyperplane*) dari beberapa garis pemisah alternatif yang mungkin terjadi dalam suatu kasus. Fungsi pemisah terbaik adalah untuk menemukan nilai optimal dari batas *margin* untuk memisahkan setiap kelas dengan *hyperplane* antara kelas positif dan kelas negatif. Karena SVM secara matematis memiliki solusi trivial, penelitian ini bertujuan untuk mencari fungsi pemisah yang akurat dengan membedakan antar sel-sel risiko kanker dan sel-sel normal yang sehat.

1.2 Rumusan Masalah

Kanker prostat sampai saat ini belum diketahui penyebabnya, oleh karena itu perlu dilakukan adanya analisis terkait faktor risiko terjangkitnya. Analisis dari ekspresi genetika sel prostat dapat diketahui bahwa individu tersebut berisiko terkena kanker prostat atau tidak. Data tersebut diperoleh dari *microarray dataset*, dimana bentuk datannya merupakan high dimensional dengan jumlah prediktor lebik banyak dari jumlah sampel. Hal ini menyebabkan analisis multivariat biasa tidak dapat menyelesaikan kasus ini karena terjadi multikolinearitas. Oleh karena itu dibutuhkan pendekatan melalui *machine learning* salah satunya adalah SVM.

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah bagaimana hasil klasifikasi dari ekspresi genetika terkait sel risiko terjangkit kanker prostat dan sel prostat normal yang akurat dengan metode SVM.

1.4 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah memberikan informasi terkait hasil klasifikasi menggunakan SVM pada data *microarray*.

BAB 2. TINJAUAN PUSTAKA

2.1 Kanker Prostat

Kanker prostat merupakan kanker yang terjadi pada kelenjar prostat di sistem reproduksi laki-laki. Di tahun 1999 terdapat kasus lebih dari 179.000 tentang diagnosa kanker prostat di Amerika Serikat. Penyebab terjadinya kanker prostat masih belum bisa diketahui secara pasti, namun ada beberapa faktor yang memengaruhi timbulnya kanker prostat. Faktor-faktor yang mempengaruhi resiko terjadinya kanker prostat adalah usia, genetik, ras, pola makan, dan perilaku seks bebas (Basch dkk., 2012). Gejala yang dialami biasanya terjadi penyumbatan pada saluran kencing bagian bawah. Gejala atau tanda lain yang mungkin terjadi resiko kanker prostat antara lain adanya darah pada sperma, berkurangnya cairan ejakulat, dan keluhan disfungsi ereksi. Pemeriksaan untuk mencurigai adanya kanker prostat pada seseorang sangat dianjurkan agar pengobatan pada orang yang beresiko kanker prostat pada stadium dini akan memberikan hasil yang terbaik (Umbas, 2008). Faktor genetik dapat mempengaruhi faktor resiko terjadinya kanker prostat melalui ekspresi genetika dari kelenjar prostat pada sistem reproduksi.

Ekpresi genetika merupakan penentuan dari sifat suatu organisme genetika tersebut. Sifat yang terbentuk pada organisme merupakan hasil dari proses metabolisme pada sel. Genetika yang tersusun dari molekul DNA yang melalui proses transkripsi dan translasi kemudian diterjemahkan pada molekul mRNA sehingga dapat menentukan sifat dari suatu organisme. Seperti halnya pada kasus kanker prostat, genetika dari kelenjar prostat dapat memprediksi seseorang apakah beresiko terkena kanker prostat melalui beberapa analisis pada genetika yang tebentuk. Analisis ekspresi genetika dapat digunakan untuk menentukan perbedaan biologis global yang mendasari patologis umum dari kanker prostat sehingga dapat mengidentifikasi gen yang mungkin menjadi resiko terkena penyakit ini. Meskipun tidak ada korelasi dengan faktor usia dan serum prostat spesifik antigen (PSA), satu set genetika diidentifikasi dapat beresiko menjadi tumor atau tidak dengan diukur dari nilai Gleason. (Singh dkk., 2002).

2.2 Klasifikasi

Klasifikasi adalah aspek penting yang ada pada *data mining*. Teknik klasifikasi telah banyak digunakan di berbagai permasalahan dalam suatu penelitian. Klasifikasi merupakan suatu metode pengelompokan data yang akan mempelajari data latih dengan menggunakan algoritma pengklasifikasian. Adapun beberapa algoritma klasifikasi, antara lain *Bayesian Classification*, *K-Nearest Neighbor*, *Decision Tree Induction*, *Case-Based Reasoning*, *Genetic Algorithms*, dan *Support Vector Machine* (Khan dkk., 2010).

Menurut Han dkk. (2011) bahwa pengukuran terhadap kinerja klasifikasi dapat menggambarkan seberapa baik *classifier* tersebut dalam mengklasifikasikan data. *Confusion matrix* merupakan salah satu metode yang dapat menganalisa seberapa baik *classifier* mengenali tupel dari setiap kelas klasifikasi. Pada *confusion matrix* terdapat empat istilah sebagai representasi hasil klasifikasi. Keempat istilah tersebut adalah *True Positive* (TP) yang merepresentasikan kelas positif terdeteksi benar, *True Negative* (TN) merupakan data kelas negatif yang terdeteksi benar, *False Positive* (FP) adalah jumlah kelas negatif yang terdeteksi sebagai kelas positif dan *False Negative* (FN) merupakan data kelas positif namun terdeteksi sebagai kelas negatif. *Confusion matrix* dapat dilihat pada Tabel 2.1 berikut.

Tabel 2.1 *Confusion Matrix*

Kelas Asli	Kelas Prediksi	
	Negatif	Positif
Negatif	*True Negative* (TN)	*False Positive* (FP)
Positif	*False Negative* (FN)	*True Positive* (TP)

Berdasarkan tabel tersebut dapat diperoleh performa klasifikasi antara lain akurasi, presisi, *sensitivity* dan *specitivity*. Nilai akurasi merupakan nilai seberapa besar hasil keakuratan dari klasifikasi data tersebut. Presisi merupakan rasio perbandingan antara kelas benar positif dengan semua kelas hasil positif.

Sensitifity merupakan proporsi kelas positif diidentifikasi benar, sedangkan *specitifity* adalah proporsi kelas negatif yang diidentifikasi benar. Perhitungan tersebut didapatkan melalui persamaan berikut ini.

$$\text{Akurasi} = \frac{TP + TN}{TP + FP + TN + FN} * 100\% \quad (2.1)$$

$$\text{Presisi} = \frac{TP}{TP + FP} * 100\% \quad (2.2)$$

$$\text{Sensitivity} = \frac{TP}{TP + FN} * 100\% \quad (2.3)$$

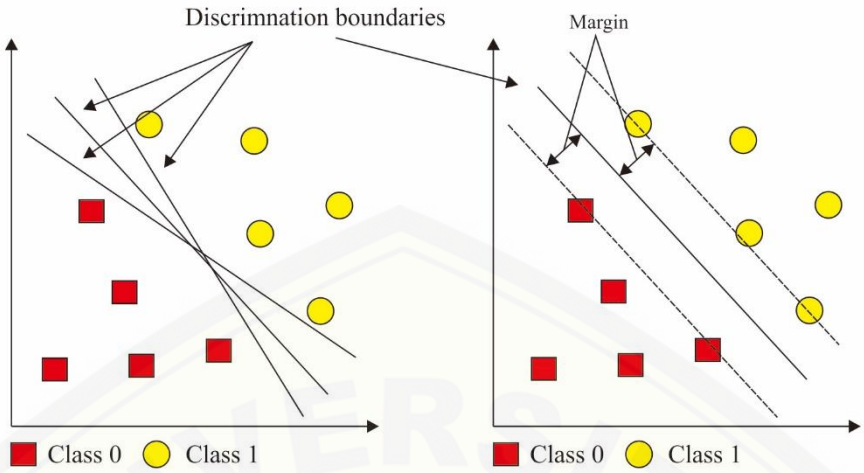
$$\text{Specitifity} = \frac{TN}{TN + FP} * 100\% \quad (2.4)$$

2.3 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) pertama kali diperkenalkan oleh Vapnik pada tahun 1995 sebagai rangkaian harmonis konsep-konsep unggulan dalam bidang *pattern recognition*. SVM merupakan salah satu metode terbaik yang bisa dipakai dalam permasalahan klasifikasi. SVM adalah metode learning machine yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *inputspace*. Pada 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 (Nugroho dkk., 2003).

2.3.1 *Linear Support Vector Machine*

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah *class* pada *input space*. Gambar 2.1 memperlihatkan beberapa *pattern* yang merupakan anggota dari dua buah *class* : 1 dan 0. *Pattern* yang tergabung pada class 0 disimbolkan dengan warna merah (kotak), sedangkan *pattern* pada class 1, disimbolkan dengan warna kuning (lingkaran).



Gambar 2.1 *Hyperplane* yang memisahkan kedua *class*.

Problem klasifikasi dapat diterjemahkan dengan usaha menemukan garis pemisah (*hyperplane*) yang memisahkan antara kedua kelompok tersebut. Berbagai alternatif garis pemisah (*discrimination boundaries*) ditunjukkan pada Gambar 2.1. *Hyperplane* pemisah terbaik antara kedua *class* dapat ditemukan dengan mengukur margin *hyperplane* tersebut dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing *class*. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Garis solid pada Gambar 2.1 sebelah kanan menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua *class*, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah *support vector*. Usaha untuk mencari titik *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM (Gunn, 1998).

Tiap data dinotasikan sebagai $x_i \in R^p,\ i = 1,2,\dots,n.$ dimana n adalah banyaknya data. *Positive class* dinotasikan sebagai 1, dan *negative class* sebagai 0. Dengan demikian, tiap data dan label *class*-nya dinotasikan sebagai : $y_i \in \{0,1\}$. Diasumsikan bahwa kedua *class* tersebut dapat dipisahkan secara sempurna oleh *hyperplane* di *D-dimensional feature space*. *Hyperplane* tersebut didefinisikan sebagai berikut :

$$\vec{w}.\vec{x_i} + b = 0 \qquad (2.5)$$

Data x_i yang tergolong ke dalam *negative class* adalah mereka yang memenuhi pertidaksamaan berikut :

$$\vec{w}.\vec{x_i} + b \leq -1 \tag{2.6}$$

Adapun data x_i yang tergolong ke dalam *positive class* yang memenuhi pertidaksamaan berikut :

$$\vec{w}.\vec{x_i} + b \geq 1 \tag{2.7}$$

(Suykens, 1999).

Optimal *margin* dihitung dengan memaksimalkan jarak antara *hyperplane* dan *pattern* terdekat. Jarak ini dirumuskan sebagai $1/\|\vec{w}\|$ ($\|\vec{w}\|$ adalah *norm* dari vektor *w*). Selanjutnya, masalah ini diformulasikan ke dalam *Quadratic Programming* (QP) problem, dengan meminimalkan invers persamaan seperti berikut.

Meminimalkan :

$$\|\vec{w}\|^2 = w^T w \tag{2.8}$$

Dengan syarat:

$$y_i(\vec{w}.\vec{x_i} + b) - 1 \geq 0, \forall_i \tag{2.9}$$

Optimisasi ini dapat diselesaikan dengan Lagrange Multipliers:

$$L(\vec{w}, b, \alpha) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{n} \alpha_i y_i(\vec{w}.\vec{x_i} + b - 1) \tag{2.10}$$

α_i adalah *Langrange multiplier* yang berkorespondensi dengan x_i. Nilai α_i adalah nol atau positif. Untuk menyelesaikan masalah tersebut. Pertama-tama meminimalkan L terhadap w, dan memaksimalkan L terhadap α_i. Dengan memodifikasi persamaan (2.10), memaksimalkan masalah di atas dapat direpresentasikan dalam α_i

Memaksimalkan:

$$L = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j \tag{2.11}$$

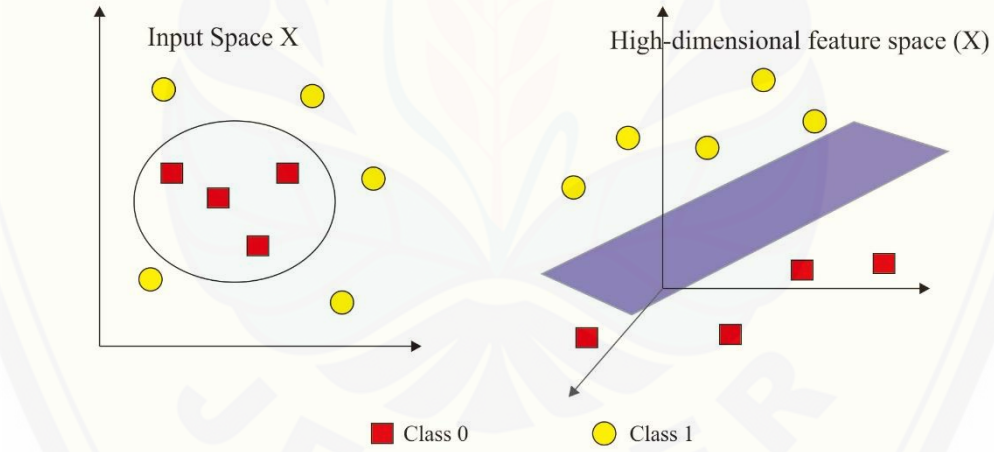
Dengan syarat:

$$\alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.12}$$

Solusi dari permasalahan di atas menghasilkan banyak nilai α_i nol. Data yang berkorespondensi dengan α_i yang tidak nol, merupakan *support vectors*, yaitu data yang memiliki jarak terdekat dengan *hyperplane* (Prasetyo, 2012).

2.3.2 *Non Linear Support Vector Machine*

SVM merupakan salah satu varian dari *linear machine* sehingga hanya dapat dipakai untuk menyelesaikan masalah yang sifatnya *linear separable*. Untuk dapat dipakai dalam kasus *non-linear*, pertama-tama data yang berada pada ruang vektor awal ($\{\overrightarrow{x_i} \in \Re^D\}$) berdimensi D, harus dipetakan ke ruang vektor baru yang berdimensi lebih tinggi ($\{\overrightarrow{x_i'} \in \Re^Q\}$). Fungsi pemetaan tersebut dinotasikan sebagai $\Phi(x)$. Pemetaan ini bertujuan untuk merepresentasikan data ke dalam format yang *linear separable* pada ruang vektor yang baru. Ilustrasi proses ini ditunjukkan pada Gambar 2.2.



Gambar 2.2 Memetakan data ke ruang vektor yang lebih tinggi.

$$\Phi : \Re^D \rightarrow \Re^Q, D < Q \tag{2.13}$$

(Suykens, 1999).

Proses optimisasi pada fase ini memerlukan perhitungan *dot product* dua buah variabel pada ruang vektor baru. *Dot product* kedua buah vektor ($\overrightarrow{x_i}$) dan ($\overrightarrow{x_j}$) dinotasikan sebagai $\Phi(\overrightarrow{x_i}).\Phi(\overrightarrow{x_j})$. Nilai *dot product* kedua buah vektor ini dapat dihitung secara tak langsung, yaitu tanpa mengetahui fungsi transformasi Φ. Teknik komputasi ini disebut *Kernel Trick*, yaitu menghitung *dot product* dua

buah vektor di ruang vektor baru dengan memakai komponen kedua buah vektor tersebut di ruang vektor asal sebagai berikut.

$$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = \Phi(\overrightarrow{x_i}).\Phi(\overrightarrow{x_j}) \quad (2.14)$$

Berbagai jenis fungsi dapat dipakai sebagai kernel K, sebagaimana tercantum pada Tabel 2.2.

Tabel 2.2 Fungsi Kernel dalam SVM

Nama Kernel	Definisi
Linear	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = (\overrightarrow{x_i}.\overrightarrow{x_j})$
Polynomial	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = (\overrightarrow{x_i}.\overrightarrow{x_j} + 1)^p$
Gaussian RBF	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = \exp(-\frac{\|\overrightarrow{x_i}-\overrightarrow{x_j}\|^2}{2\sigma^2})$
Sigmoid	$K(\overrightarrow{x_i}, \overrightarrow{x_j}) = \tanh(\alpha\overrightarrow{x_i}.\overrightarrow{x_j} + \beta)$

Selanjutnya klasifikasi non linear pada SVM terhadap *test sample* dirumuskan sebagai berikut.

$$f(\Phi(x)) = w.\Phi(x) + b \quad (2.15)$$

$$= \sum_{i=1, x_i \in SV}^{n} \alpha_i y_i \Phi(x).\Phi(x_i) + b \quad (2.16)$$

$$= \sum_{i=1, x_i \in SV}^{n} \alpha_i y_i \mathrm{K}(x, x_i) + b \quad (2.17)$$

Support vector SV adalah subset dari data training set, dengan α yang tidak negatif $\{x_i | \alpha_i > 0\}$ $(i = 1,2, \ldots, l)$ (Kecman, 2005).

2.4 *K-Fold CrossValidation*

K-fold cross validation merupakan pengujian yang berfungsi untuk menilai kinerja proses algoritma dengan membagi data sampel secara acak dan mengelompokkan data tersebut sebanyak nilai *k* pada *k-fold.* Kemudian satu kelompok *k-fold* dijadikan sebagai data testing dan kelompok lainnya sebagai data training (Rohani dkk., 2017). Penggunaan *k-fold cross validation* memungkinkan

untuk melakukan estimasi kesalahan generalisasi (Anguita dkk., 2009). Hutapea dkk. (2018) menerangkan pada hasil penelitiannya bahwa hasil pengujian terbaik menggunakan *k-fold cross validation* dengan menghasilkan nilai akurasi terbesar menggunakan 5 *fold*. Menurut Lidya dkk. (2015) *k-fold cross validation* 4 *fold*, 5 *fold*, 6 *fold*, 10 *fold*, 11 *fold*, 12 *fold*, 13 *fold*, 14 *fold* dan 15 *fold* yang digunakan pada penelitiannya menghasilkan bahwa pada nilai k = 10 adalah nilai yang optimal dengan akurasi terbesar.

2.5 *Support Vector Machine* pada R

R merupakan bentukan kolaborasi dari ahli statistika dan matematika di dunia yang dibuat oleh Ross Ihaka dan Robert Gentleman. Bahasa dalam R merupakan basis dari bahasa S yang dikembangkan oleh Chambers di Bell Laboratory. R merupakan pemrograman yang dapat melakukan impor data (*big data*) dan juga dapat membaca data dari program lain (*data base management program*). R tidak hanya digunakan untuk statistik, dengan dilengkapi berbagai paket yang telah ditambahkan pada program ini sehingga R juga dapat digunakan dalam banyak bidang dengan segala jenis data. Salah satu paket yang ada dalam R adalah SVM. Terdapat empat paket terkait SVM yang ada pada program R yaitu paket `e1071`, `kernlab`, `klaR`, dan `svmpath`. Paket `e1071` merupakan paket SVM yang berfungsi untuk metode penyelesaian parameter dan visualisai. Paket `kernlab` untuk mengoptimalkan basis kernel sehingga memberikan hasil implementasi SVM yang fleksibel dan diperluas. Paket `klaR` mengimplementasikan SVM dengan klasifikasi seperti analsis pemisah reguler. Paket terakhir yaitu `svmpath` yang berfungsi untuk menyediakan algoritma yang sesuai dengan hasil solusi pada SVM (Karatzouglou dkk., 2006).

BAB 3. METODOLOGI PENELITIAN

Dalam bab ini akan dibahas dua hal terkait dengan metodologi penelitian yaitu jenis dan bentuk data yang akan digunakan serta langkah-langkah dalam melakukan penelitian, keduanya dijelaskan sebagai berikut.

3.1 Jenis dan Bentuk Data

Jenis data yang digunakan pada penelitian ini merupakan data sekunder yang didapatkan dari situs https://ico2s.org/datasets/microarray.html. Bentuk data genetik pada sel prostat merupakan hasil mutasi genetika pada DNA melalui proses analisis *microarray* membentuk nilai Gleason. Analisis *microarray* yang membentuk nilai Gleason digunakan untuk membantu diagnosis risiko kanker prostat dengan skor berkisar antara 1 sampai 10. Karakteristik data *microarray* adalah tipe data dengan jumlah variabel penelitian lebih banyak dari jumlah individu yang diteliti. Pada penelitian ini data yang digunakan berupa 102 individu dengan 2135 variabel yang akan diklasifikasikan kedalam kelas risiko tumor dan kelas normal.

3.2 Langkah-langkah dan Metode Penelitian

Metode yang digunakan dalam penilitian ini adalah SVM dengan menggunakan *software* R studio. Langkah-langkah yang dilakukan dalam penelitian ini adalah sebagai berikut:

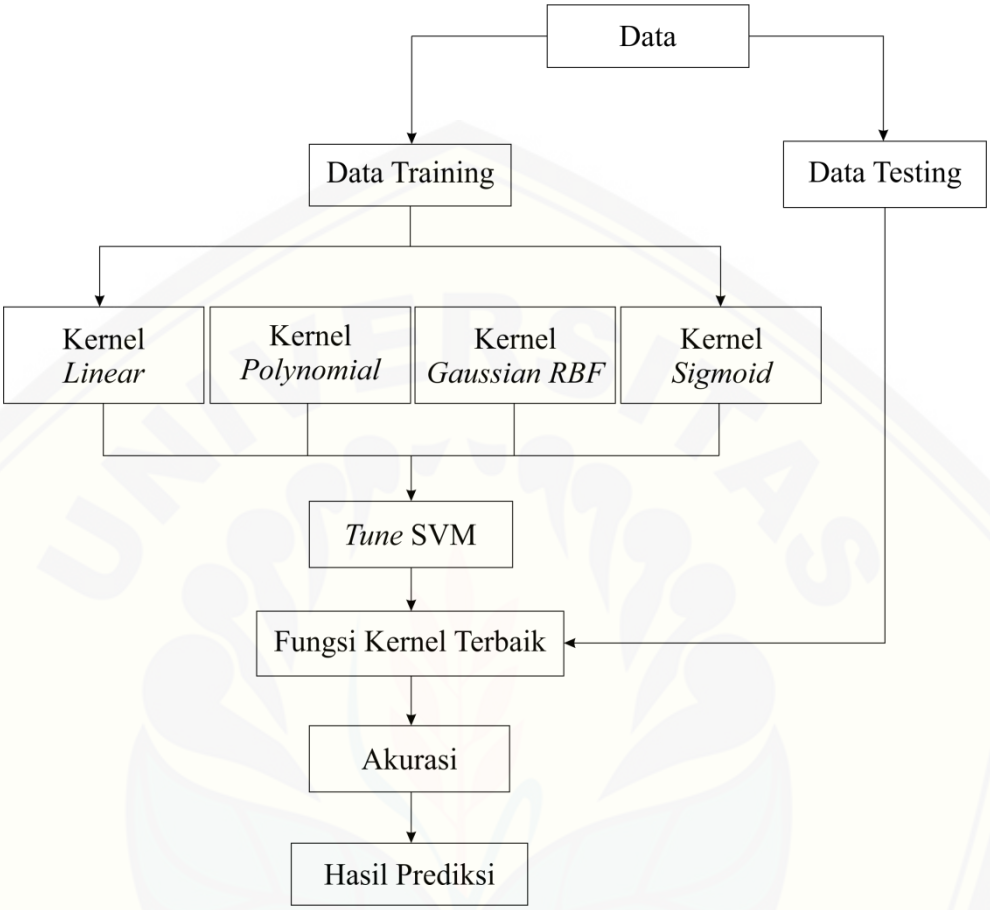
1. Melakukan pengolahan data.

   Data yang diperoleh dari situs di 3.1 disesuaikan menjadi bentuk matriks terlebih dahulu melalui program *excel*. Data yang akan digunakan adalah varibel x dengan total 2135 variabel sebagai *input* dan variabel y dengan kelas normal dan kelas tumor sebagai *output*. Berikutnya data diinputkan kedalam program R melalui paket `readxl`.

2. Membagi data menjadi dua bagian, *training* dan *testing*

   Melakukan *spliting* data 75:25 dengan proporsi sama setiap kelas. Pembagian data menggunakan paket *caret* dengan *function* `createDataPartition`. Kemudian membuat data frame dari data training dan data testing untuk kemudian diolah dalam SVM.
3. Melakukan uji data *training* pada program R menggunakan paket `e1071` dengan menggunakan fungsi kernel
   i. Menjalankan uji *training* dengan fungsi kernel *linear* menggunakan *k-fold cross validation* dengan nilai *k* adalah 5 dan 10.
   ii. Menjalankan uji *training* dengan fungsi kernel *polynomial* menggunakan *k-fold cross validation* dengan nilai *k* adalah 5 dan 10.
   iii. Menjalankan uji *training* dengan fungsi kernel *radial* menggunakan *k-fold cross validation* dengan nilai *k* adalah 5 dan 10.
   iv. Menjalankan uji *training* dengan fungsi kernel *sigmoid* menggunakan *k-fold cross validation* dengan nilai *k* adalah 5 dan 10.
4. Melakukan tuning parameter untuk mencari nilai *cost* terbaik dari hasil uji data *training* pada semua kernel untuk digunakan pada data *testing*.
5. Melakukan pengujian pada *data testing* menggunakan fungsi kernel yang memiliki nilai *cost* terbaik.
6. Menentukan hasil akurasi dan hasil prediksi klasifikasi kelas normal dan tumor pada eskpresi genetik kanker prostat.

Secara singkat langkah-langkah penelitian diuraikan sebagai berikut :



Gambar 3.1 Langkah-langkah penelitian.

BAB 5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan analisis yang telah dilakukan sebelumya, dapat ditarik kesimpulan bahwa pengujian terbaik menggunakan fungsi kernel linear yang memiliki nilai error terkecil dengan cost parameternya sebesar 0,001 dan menggunakan metode 5 fold cross validation. Hasil pengujian pada data testing memberikan kemampuan generalisasi terbaik dengan tingkat akurasinya sebesar 92%. Klasifikasi dari sel kanker prostat menghasilkan 12 data terklasifikasi secara benar sebagai kelas normal dan ada 1 data kesalahan klasifikisasi yang termasuk kedalam kelas tumor atau *False Positive*. Pada kelas tumor terdapat 11 data terklasifikasikan secara benar dan 1 data terklasifikasikan sebagai kelas normal atau *False Negative*.

5.2 Saran

Berdasarkan hasil dan kesimpulan yang diperoleh, hal yang sebaiknya dikembangkan untuk penelitian selanjutnya adalah melakukan pengujian dengan menambah parameter lain, sehingga hasil yang didapatkan lebih baik.

DAFTAR PUSTAKA

Andari, S. 2013. *Smooth Support Vector Machine Dan Multivariate Adaptive Regression Splines Untuk Mendiagnosis Kanker Payudara*. Surabaya : Institut Teknologi Sepuluh Nopember.

Anguita, D., A. Ghio, S. Ridella dan D. Sterpi. 2009. K-fold Cross Validation for Error Rate Estimate in Support Vector Machine. *Proceedings of The 2009 International Conference on Data Mining (DMIN)*, pp. 291-297.

Basch, E., T. K. Oliver, A. Vickers, I. Thompson, P. Kantoff, H. Parnes, D. A. Lowblaw, B. Roth, J. Williams dan R. K. Nam. 2012. Screening for Prostate Cancer With Prostate-Specific Antigen Testing: American Society of Clinical Oncology Provisional Clinical Opinion. *Journal of Clinical Oncologi*. Vol. 30, No. 24, 20 Agustus 2012.

Damanik, S. M. S., D. Ispriyanto dan Sugito. 2015. Klasifikasi Lama Studi Mahasiswa FSM Universitas Diponegoro Menggunakan Regresi Logistik Biner dan Support Vector Machine. *Jurnal Gaussian*. Vol. 4, hal. 123-132.

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

Han, J., M. Kamber dan J. Pei. 2012. *Data Mining: Concept and Techniques, Third Edition*. Waltham: Morgan Kaufmann Publisher.

Hutapea, A., M. T. Furqon dan Indriati. 2018. Penerapan Algoritme Modified K-Nearest Neighbour Pada Pengklasifikasian Penyakit Kejiwaan Skizofrenia. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. Vol 2, No. 10, hlm. 3957-3961.

Karatzouglou, A., D. Meyer dan K. Hornik. 2006. Support Vector Machine in R. *Journal of Statistic Software*. Vol. 15, Issue 9.

Kecman, V. 2005. *Support Vector Machine – An Introduction*. Netherlands: Springer -Verlag Berlin Heidelberg.

Khan, Aurangzeb, B. Baharudin, L. H. Lee dan K. Khan. 2010. A Review of Machine Learning Alghoritms for Text-Documents Classification. *Journal of Advances in Infromation Technology*. Vol. 1, No. 1 hal. 4-20.

Lidya, S. K., O. S. Sitompul dan S. Efendi. 2015. Sentiment Analysis Pada Teks Bahasa Indonesia Menggunakan Support Vector Machine (SVM) dan K-Nearest Neighbour (K-NN). *Seminar Nasional Teknologi Informasi dan Komunikasi 2015 (SENTIKA 2015)*, pp 1-8.

Nugroho, A.S., A. B. Witarto dan D. Handoko. 2003. *Suport Vector Machines* : Teori Aplikasinya dalam Bioinformatika.

Pratama, A., R. C. Wihandika dan D. E. Ratnawati. 2018. Implementasi Algoritme Support Vector Machine (SVM) untuk Prediksi Ketepatan Waktu Kelulusan Mahasiswa. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. Vol. 2, No. 4, April 2018, hal. 1704-1708.

Prasetyo, E. 2012. *Data Mining Konsep dan Aplikasi Menggunakan MATLAB*. Yogyakarta: Andi.

Rohani, A., M. Taki dan M. Abdollahpour. 2017. A Novel Soft Computing Model (Gaussian Process Regression with K-Fold Cross Validation) for Daily and Monthly Solar Radiation Forecasting (Part: I). *Renewable Energy*. Vol 115, hal. 411-422.

Singh, D., P. G. Febbo, K. Ross, D. G. Jackson, J. Manola, C. Ladd, P. Tamayo, A. A. Renshaw, A. V. D'Amico, J. P. Richie, E. S. Lander, M. Loda, P. W. Kantoff, T. R. Golub dan W. R. Sellers. 2002. Gene Expression Correlates of Clinical Prostate Cancer Behavior. *Cancer Cell* 1: 203-209.

Suykens, J. A. K. dan J. Vandewalle. 1999. Least Squares Support Vector Machine Classifier. *Journal Neural Processing Letters*, 9: 293-300.

Umbas, R. 2008. Penanganan Kanker Prostat saat ini dan Beberapa Pengembangan Baru. *Jurnal Kanker Indonesia 3*, 114-119.

LAMPIRAN

Lampiran A. Data Penelitian

No	1005_at	1007_s_at	1008_f_at	1020_s_at	…	999_g_at	Output
1	5,690595	5,887783	1,888793	6,188889	⋯	2,626398	0
2	2,751449	4,748083	9	4,019577	…	1,827826	0
3	2,974932	4,092879	9	4,126551	…	1,798521	0
4	2,924624	4,816713	9	4,000468	…	1,894559	0
5	2,405802	4,331438	9	4,253253	…	1,9248	0
6	2,699247	4,554692	9	4,0561	…	1,945649	0
7	2,729224	4,449875	9	4,031817	…	1,837177	0
8	2,786266	4,372375	9	4,174509	…	1,894491	0
9	2,914426	4,426925	9	4,057026	…	1,88538	0
10	2,632984	5,282323	9	4,540055	…	1,721393	0
11	2,772272	4,893885	9	4,052497	…	1,806207	0
12	7,418538	5,220845	9	5,882558	…	2,18226	0
. . .	. . .	. . .	. . .	. . .	… … …	. . .	. . .
48	7,531677	6,079702	4,374225	7,736808	⋯	1,672575073	0
49	7,619853	6,173804	3,20478	7,912641	…	1,615876888	0
50	7,509991	6,265975	4,159739	7,659103	…	1,590849808	0
51	7,635141	5,946033	7,326978	8,709325	…	1,721618329	1
52	2,595446	4,044766	9	8,956182	…	1,840458253	1
53	2,721188	4,822558	9	7,988582	…	1,836245331	1
. . .	. . .	. . .	. . .	. . .	… … …	. . .	. . .
97	8,285381	6,135713	4,833327	8,20467	⋯	1,710686	1
98	6,776298	6,580912	5,368542	9	…	1,610204	1
99	4,934895	6,213611	7,075594	7,960663	…	1,709556	1
100	6,627963	7,111646	4,399441	7,943872	…	1,697315	1
101	5,343997	6,729185	4,221106	8,332856	…	1,583947	1
102	7,443523	6,838929	4,164413	8,111479	…	1,548268	1

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

```
> library(e1071)
> library(caret)
Loading required package: lattice
Loading required package: ggplot2
> library(readxl)
> data=read_excel("G:\\Tugas Akhir\\Prostate.xlsx")
> split = (createDataPartition(y=data$Output, p=0.75,
list=FALSE))
> split
Resample1
```

[1,]	3	[21,]	27	[41,]	57	[61,]	84
[2,]	4	[22,]	29	[42,]	58	[62,]	85
[3,]	5	[23,]	30	[43,]	60	[63,]	86
[4,]	6	[24,]	31	[44,]	62	[64,]	87
[5,]	7	[25,]	32	[45,]	63	[65,]	88
[6,]	8	[26,]	33	[46,]	65	[66,]	89
[7,]	9	[27,]	34	[47,]	66	[67,]	90
[8,]	12	[28,]	36	[48,]	67	[68,]	91
[9,]	13	[29,]	38	[49,]	69	[69,]	93
[10,]	14	[30,]	39	[50,]	70	[70,]	94
[11,]	15	[31,]	41	[51,]	71	[71,]	95
[12,]	16	[32,]	42	[52,]	72	[72,]	97
[13,]	17	[33,]	43	[53,]	73	[73,]	98
[14,]	18	[34,]	46	[54,]	74	[74,]	99
[15,]	20	[35,]	47	[55,]	76	[75,]	100
[16,]	21	[36,]	48	[56,]	78	[76,]	101
[17,]	22	[37,]	50	[57,]	79	[77,]	102
[18,]	23	[38,]	51	[58,]	80		
[19,]	25	[39,]	53	[59,]	81		
[20,]	26	[40,]	55	[60,]	83		

```
> training_set = data[split,]
> dim(training_set)
[1]   77 2136
```

```
> test_set = data[-split,]
> dim(test_set)
[1]   25 2136
```

Lampiran C1. Pengujian Data *Training* Kernel *Linear*

```
> tc <- tune.control(cross = 5)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'linear', trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "linear", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1
Number of Support Vectors:  48
 ( 28 20 )
Number of Classes:  2
Levels:  0 1
>      y_train_pred     =     predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 37  0
  1  0 40
Success ratio on training set :  100 %
> tc <- tune.control(cross = 10)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'linear', trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "linear", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1
Number of Support Vectors:  48
 ( 28 20 )
Number of Classes:  2
Levels:
 0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 37  0
  1  0 40
Success ratio on training set :  100 %
```

Lampiran C2. Pengujian Data *Training* Kernel *Polynomial*

```
> tc <- tune.control(cross = 5)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                                kernel  =  'polynomial',
trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "polynomial", trainControl
= tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  polynomial
       cost:  1  degree:  3
Number of Support Vectors:  61
 ( 31 30 )
```

```
Number of Classes:  2
Levels:  0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 24 13
  1  0 40
Success ratio on training set :  83.1168831168831 %
> tc <- tune.control(cross = 10)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                                kernel  =  'polynomial',
trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "polynomial", trainControl
= tc)
Parameters:
 SVM-Type:  C-classification
```

```
 SVM-Kernel:  polynomial
       cost:  1   degree:  3
Number of Support Vectors:  61
 ( 31 30 )
Number of Classes:  2
Levels:  0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 24 13
  1  0 40
Success ratio on training set :  83.1168831168831 %
```

Lampiran C3. Pengujian Data *Training* Kernel *Radial*

```
> tc <- tune.control(cross = 5)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'radial', trainControl=tc)
> summary(classifier)
Call:
```

```
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "radial", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
Number of Support Vectors:  68
 ( 35 33 )
Number of Classes:  2
Levels: 0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 37  0
  1  4 36
Success ratio on training set :  94.8051948051948 %
> tc <- tune.control(cross = 10)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
```

```
+                   kernel = 'radial', trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "radial", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
Number of Support Vectors:  68
 ( 35 33 )
Number of Classes:  2
Levels: 0 1
>       y_train_pred        =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 37  0
  1  4 36
Success ratio on training set :  94.8051948051948 %
```

Lampiran C4. Pengujian Data *Training* Kernel *Sigmoid*

```
> tc <- tune.control(cross = 5)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'sigmoid', trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "sigmoid", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
Number of Support Vectors:  50
 ( 26 24 )
Number of Classes:  2
Levels:  0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
```

```
     0  1
  0 36  1
  1  5 35
Success ratio on training set :  92.2077922077922 %
> tc <- tune.control(cross = 10)
> classifier = svm(formula = Output ~ .,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'sigmoid', trainControl=tc)
> summary(classifier)
Call:
svm(formula = Output ~ ., data = training_set, type =
"C-classification", kernel = "sigmoid", trainControl =
tc)
Parameters:
 SVM-Type:  C-classification
 SVM-Kernel:  sigmoid
       cost:  1
Number of Support Vectors:  50
 ( 26 24 )
Number of Classes:  2
Levels: 0 1
>       y_train_pred       =       predict(classifier,
newdata=training_set[-2136])
> cm_train = table(training_set$Output, y_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 :
   y_train_pred
     0  1
  0 36  1
  1  5 35
Success ratio on training set :  92.2077922077922 %
```

Lampiran D. Proses *Tuning* SVM

```
#Tune SVM 5 fold
> tc <- tune.control(cross = 5)
> tuning <- tune(svm, Output~., data = training_set,
kernel = "linear", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method: 5-fold cross validation
- best parameters:
  cost
 0.001
- best performance: 0987119
- Detailed performance results:
   cost     error
1 1e-03 0.0987119
2 1e-02 0.1053421
3 1e-01 0.1053421
4 1e+00 0.1053421
5 1e+01 0.1053421
6 1e+02 0.1053421
```

```
> tuning <- tune(svm, Output~., data = training_set,
kernel = "polynomial", ranges = list(cost = c(0.001,
0.01, 0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
5-fold cross validation
- best parameters:
 cost
  100
- best performance:
0.1494830
- Detailed performance results:
   cost     error
1 1e-03 0.5627112
2 1e-02 0.4521978
3 1e-01 0.2846272
4 1e+00 0.2137213
5 1e+01 0.1588865
6 1e+02 0.1494830
> tuning <- tune(svm, Output~., data = training_set,
kernel = "radial", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
5-fold cross validation
- best parameters:
 cost
   10
- best performance: 0.1196087
```

```
- Detailed performance results:
   cost     error
1 1e-03 0.5278105
2 1e-02 0.5112508
3 1e-01 0.3697355
4 1e+00 0.1383781
5 1e+01 0.1196087
6 1e+02 0.1196087
> tuning <- tune(svm, Output~., data = training_set,
kernel = "sigmoid", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
5-fold cross validation
- best parameters:
 cost
  0.1
- best performance:
0.1132515
- Detailed performance results:
   cost       error
1 1e-03   0.4191112
2 1e-02   0.3855034
3 1e-01   0.1969137
4 1e+00   0.1132515
5 1e+01   2.0373428
6 1e+02 187.2775952
```

```
#Tune SVM 10 fold
> tuning <- tune(svm, Output~., data = training_set,
kernel = "linear", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
10-fold cross validation
- best parameters:
 cost
 0.001
- best performance:
0.1097182
- Detailed performance results:
   cost     error
1 1e-03 0.1097182
2 1e-02 0.1175649
3 1e-01 0.1175649
4 1e+00 0.1175649
5 1e+01 0.1175649
6 1e+02 0.1175649
> tuning <- tune(svm, Output~., data = training_set,
kernel = "polynomial", ranges = list(cost = c(0.001,
0.01, 0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
10-fold cross validation
- best parameters:
 cost
 0.01
```

```
- best performance:
0.1461275
- Detailed performance results:
   cost     error
1 1e-03 0.4672688
2 1e-02 0.3692147
3 1e-01 0.2631901
4 1e+00 0.1987705
5 1e+01 0.1535774
6 1e+02 0.1461275
> tuning <- tune(svm, Output~., data = training_set,
kernel = "radial", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
10-fold cross validation
- best parameters:
 cost
 10
- best performance:
0.1150083
- Detailed performance results:
   cost     error
1 1e-03 0.5392827
2 1e-02 0.5148097
3 1e-01 0.3171059
4 1e+00 0.1321449
5 1e+01 0.1150083
6 1e+02 0.1150083
```

```
> tuning <- tune(svm, Output~., data = training_set,
kernel = "sigmoid", ranges = list(cost = c(0.001, 0.01,
0.1, 1, 10, 100)), tunecontrol = tc)
> summary(tuning)
Parameter tuning of ‘svm’:
- sampling method:
10-fold cross validation
- best parameters:
 cost
 1
- best performance:
0.1216425
- Detailed performance results:
   cost       error
1 1e-03   0.5184955
2 1e-02   0.4615622
3 1e-01   0.2062234
4 1e+00   0.1216425
5 1e+01   1.8650912
6 1e+02 162.9760953
```

Lampiran E. Pengujian Data *Testing*

```
#Pengujian 5 fold
> folds = createFolds(training_set$Output, k = 5)
> cv = lapply(folds, function(x) {
+     training_fold = training_set[-x, ]
+     test_fold = training_set[x, ]
+     classifier = svm(formula = Output ~ .,
+                      data = training_fold,
+                      type = 'C-classification',
+                      kernel = 'linear',cost=0.001)
```

```
+           y_pred = predict(classifier, newdata =
test_fold[-2136])
+     cm_test= table(test_fold$Output, 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)
+   })
```

cv	list [5]	List of length 5
Fold1	double [1]	0.9375
Fold2	double [1]	1
Fold3	double [1]	0.8
Fold4	double [1]	0.9333333
Fold5	double [1]	0.9375

```
> accuracy = mean(as.numeric(cv))
> accuracy
[1] 0.9216667
```

Lampiran F. Prediksi Klasifikasi

```
>   prediksi   =   predict(tuning$best.model,newdata   =
test_set[-2136])
> table(test_set$Output,prediksi)
   prediksi
     0  1
  0 12  1
  1  1 11
> prediksi
 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
19 20 21 22 23 24 25
 0  0  0  0  0  0  0  1  0  0  0  0  0  1  1  1  1  1
1  1  1  1  1  0  1
Levels: 0 1
```

Lampiran G. Data Hasil Klasifikasi

No	Kelas Asli	Kelas Klasifikasi
1	Normal	Normal
2	Normal	Normal
3	Normal	Normal
4	Normal	Normal
5	Normal	Normal
6	Normal	Normal
7	Normal	Normal
8	Normal	Tumor
9	Normal	Normal
10	Normal	Normal
11	Normal	Normal
12	Normal	Normal
13	Normal	Normal
14	Tumor	Tumor
15	Tumor	Tumor
16	Tumor	Tumor
17	Tumor	Tumor
18	Tumor	Tumor
19	Tumor	Tumor
20	Tumor	Tumor
21	Tumor	Tumor
22	Tumor	Tumor
23	Tumor	Tumor
24	Tumor	Normal
25	Tumor	Tumor