PREDIKSI ASETILASI PADA *SEQUENCE* PROTEIN LISIN MENGGUNAKAN *SUPPORT VECTOR MACHINE*

(SKRIPSI)

Oleh

ESTER DEBORA PRISCILA SILALAHI
1517051235

JURUSAN ILMU KOMPUTER

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS LAMPUNG

BANDAR LAMPUNG

2019

ABSTRAK

Modifikasi pasca-translasi (PTM) adalah mekanisme penting yang terlibat dalam pengaturan fungsi protein yang terdiri dari berbagai macam seperti, fosforilasi, glikosilasi, ubiquitinasi, metilasi, asetilasi, dan lipidasi. Salah satu modifikasi pasca-translasi yang paling umum adalah modifikasi pasca-translasi asetilasi yang terjadi pada protein lisin. Asetilasi pada protein lisin adalah modifikasi besar pasca-translasi yang memainkan peran penting dalam berbagai proses biologis penting, seperti ekspresi gen, metabolisme. *Support Vector Machine* (SVM) adalah metode *machine learning* yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *input space* dan merupakan metode klasifikasi yang digunakan untuk mengolah data yang bersifat linear maupun non-linear. Oleh karena itu dilakukan penelitian ini yang bertujuan untuk mengklasifikasikan dan mendapatkan hasil prediksi dari data asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine*. Hasil prediksi dari 3 kernel *Support Vector Machine*, didapatkan akurasi pada kernel Linear 82.78%, kernel Polynomial 95.68% dan kernel Gaussian 97.52%.

Kata kunci: *Post-Translational Modification*, *Support Vector Machine*, Asetilasi

PREDIKSI ASETILASI PADA SEQUENCE PROTEIN LISIN MENGGUNAKAN SUPPORT VECTOR MACHINE

Oleh

Ester Debora Priscila Silalahi

Skripsi

Sebagai Salah Satu Syarat Untuk Memperoleh Gelar
SARJANA KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2019

Judul Skripsi	: **PREDIKSI ASETILASI PADA *SEQUENCE* PROTEIN LISIN MENGGUNAKAN *SUPPORT VECTOR MACHINE***
Nama Mahasiswa	: **Ester Debora Priscila Silalahi**
Nomor Pokok Mahasiswa	: 1517051235
Jurusan	: Ilmu Komputer
Fakultas	: Matematika dan Ilmu Pengetahuan Alam

Menyetujui,
Komisi Pembimbing

Favorisen R. Lumbanraja, Ph.D.
NIP 19830110 200812 1 002

Mengetahui,
Ketua Jurusan Ilmu Komputer

Dr. Ir. Kurnia Muludi, M.S.Sc.
NIP 19640616 198902 1 001

MENGESAHKAN

1. Tim Penguji

Ketua : **Favorisen R. Lumbanraja, Ph.D.** .....

Penguji I
Bukan Pembimbing : **Didik Kurniawan, S.Si., M.T.** .....

Penguji II
Bukan Pembimbing : **Dr. Eng Admi Syraif** .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

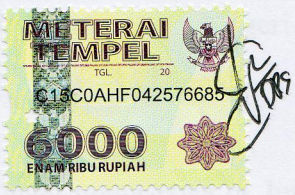
Drs. Suratman, M.Sc.,
NIP 19640604 199003 1 002

Tanggal Lulus Ujian Skripsi : **30 September 2019**

PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya yang berjudul "Prediksi Asetilasi Pada Sequence Protein Lisin Menggunakan *Support Vector Machine*" merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang di skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila dikemudian hari terbukti skripsi saya merupakan hasil penjiplakan atau dibuat orang lain, maka saya bersedia menerima sanksi berupa pencabutan gelar yang saya terima.

Bandar Lampung, 7 Oktober 2019



Ester Debora Priscila Silalahi

NPM. 1517051235

RIWAYAT HIDUP

Penulis dilahirkan pada tanggal 30 Mei 1997di Bandar Lampung, sebagai anak kedua dari empat bersaudara dengan Ayah yang bernama Almer Silalahi dan Ibu bernama Lasmaria Mesrawani Sagala. Penulis menyelesaikan pendidikan formal pertama kali di TK Xaverius 3 Bandar Lampung pada tahun 2002, kemudian melanjutkan pendidikan dasar di SD Xaverius 3 Bandar Lampung dan selesai pada tahun 2009. Pendidikan menengah pertama di SMP Fransiskus 1 Bandar Lampung diselesaikan penulis pada tahun 2012, kemudian melanjutkan ke pendidikan menengah atas di SMA Negeri 13 Bandar Lampung yang diselesaikan pada tahun 2015.

Pada tahun 2015 penulis terdaftar sebagai mahasiswa di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung. Selama dalam masa perkuliahan, penulis mengikuti organisasi internal jurusan yaitu Himakom (Himpunan Mahasiswa Jurusan Ilmu Komputer) periode 2015/2016 hingga periode 2016/2017. Selama menjadi mahasiswa beberapa kegiatan yang dilakukan penulis antara lain pada bulan Januari 2018 penulis melaksanakan kerja praktik di Tribun Lampung, dan pada bulan Agustus 2018 penulis melaksanakan Kuliah Kerja Nyata (KKN) di Desa Tanjung Qencono Kabupaten Lampung Timur.

PERSEMBAHAN

Dengan segala puji yyukur dan kerendahan hati meraih berkat dari Tuhan Yesus Kristus, kupersembahkan karya kecilku ini untuk orang-orang yang aku cintai dan sayangi.

Teruntuk Bapak dan Mama yang tak pernah putus-putusnya memberi nasihat, semangat, motivasi dan doanya. Terimakasih untuk segala kasih saying, perhatian, usaha dan segala dukungan moril maupun materi.

Teruntuk teman-teman, terimakasih untuk segala canda tawa, dukungan, perjuangan, dan kenangan yang telah terukir.

Almamater Tercinta,

UNIVERSITAS LAMPUNG

MOTTO

“Janganlah hendaknya kamu kuatir tentang apapun juga, tetapi nyatakanlah dalam segala hal keinginanmu kepada Allah dalam doa dan permohonan dengan ucapan syukur”

(Filipi 4:6)

“Sebab itu janganlah kamu kuatir akan hari besok, karena hari besok mempunyai kesusahannya sendiri. Kesusahan sehari cukuplah untuk sehari”

(Matius 6:34)

“Jadi mereka yang hidup dari iman, merekalah yang diberkati bersama-sama dengan Abraham yang beriman itu”

(Galatia 3:9)

“Berdoa, Berusaha, dan Berserah pada Tuhan Yesus Kristus”

(Penulis)

SANWACANA

Syalom dan Salam Sejahtera.

Puji syukur kepada Tuhan yang Maha Esa karena berkat kasih setiaNya dan anugerahNya penulis dapat menyelesaikan skripsi ini yang berjudul "Prediksi Asetilasi Pada *Sequence* Protein Lisin Menggunakan *Support Vector Machine*". Skripsi ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

Dalam pelaksanaan dan penyusunan skripsi ini penulis sangat berterima kasih dan memberikan penghargaan yang sedalam-dalamnya kepada seluruh pihak yang membantu penulis menyelesaikan skripsi ini. Penulis ingin mengucapkan terima kasih sengan setulus hati terutama kepada:

1. Kedua Orangtua tercinta, Bapak dan Mama, serta saudara-saudaraku yang ku kasihi yang selalu memberikan dukungan, motivasi, dan doa yang tak terhingga.
2. Bapak Dekan Drs. Suratman, M.Sc., selaku Dekan FMIPA Universitas Lampung.

3. Bapak Favorisen R. Lumbanraja, Ph.D., selaku dosen pembimbing skripsi atas kesediaannya, kesabaran, dan keikhlasannya untuk memberikan dukungan, bimbingan, nasihat, saran, dan kritik dalam proses penyelesaian skripsi ini.
4. Bapak Didik Kurniawan S.Si., M.T., selaku dosen pembahas utama yang telah memberikan saran dan masukan guna penyempurnaan dalam skripsi ini.
5. Bapak Dr. Eng Admi Syarif, selaku dosen pembahas kedua skripsi, yang telah memberikan saran dan masukan guna penyempurnaan penulisan skripsi ini.
6. Bapak Dr. Ir. Kurnia Muludi, M.S.Sc., selaku ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.
7. Ibu Anie Rose Irawati, M.Cs., selaku dosen pembimbing akademik penulis yang telah memberikan arahan dan masukan tentang matakuliah kepada penulis selama penulis menjalani perkuliahan.
8. Bapak dan Ibu Dosen Jurusan Ilmu Komputer Universitas Lampung yang telah memberikan ilmu dan pengalaman hidup selama penulis menjadi mahasiswa.
9. Ibu Ade Nora Maela dan Mas Nofal yang telah membantu memudahkan segala urusan administrasi penulis di Jurusan Ilmu Komputer.
10. Seluruh keluarga dan saudaraku yang telah membantu selama proses perkuliahan yang tidak bisa disebutkan satu persatu.
11. Kenny Hutabarat yang memberikan semangat, dukungan, dan doanya.

12. Wiwit, teman seperjuangan penulis dalam bimbingan dan menyelesaikan skripsi ini.
13. Teman-temanku, Wenti, Jannati, Alinda, Dina, Hany, Wiwit, terimakasih buat dukungan, doa, canda tawa selama perkuliahan dan dalam penyusunan skripsi ini.
14. Teman-temanku Ilkomp D, terimakasih canda tawa selama empat tahun ini yang kadang suka aneh-aneh tiap orang kelakuannya. Terimakasih kebersamaannya selama ini.
15. Saudara-saudaraku NHKBP Kedaton, terimakasih dukungan, canda tawa dan doanya.
16. Semua pihak yang secara langsung maupun tidak langsung yang telah membantu dalam penyelesaian skripsi ini.

Penulis telah berusaha semaksimal mungkin dalam penulisan skripsi ini untuk mencapai suatu kelengkapan yang baik. Penulis juga mengharapkan kritik dan saran yang bersifat membangun. Akhirnya dengan segala kerendahan hati penulis berharap skripsi ini dapat memberikan manfaat, baik kepada penulis khususnya kepada pembaca.

Bandar Lampung, 7 Oktober 2019

Penulis,

Ester Debora Priscila Silalahi
NPM. 1517051235

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar **Halaman**

DAFTAR TABEL

Tabel **Halaman**

DAFTAR KODE

Kode Program **Halaman**

I. PENDAHULUAN

A. Latar Belakang

Asam amino adalah unit dasar dari protein yang merupakan peran utama dalam mengatur beberapa proses yang berkaitan dengan ekspresi gen, dan berperan penting dalam pembentukan protein. Sehingga, protein terdiri dari rantai-rantai panjang asam amino, yang terikat satu sama lain dalam ikatan peptida (Akram et al., 2011).

Post-Translational Modification (PTM) atau yang sering dikenal dengan modifikasi pasca-translasi adalah mekanisme penting yang terlibat dalam pengaturan fungsi protein. Modifikasi pasca-translasi mengacu pada penambahan kovalen dan enzimatik modifikasi protein selama atau setelah biosintesis protein, yang memainkan peran penting dalam memodifikasi fungsi protein dan mengatur ekspresi gen (Minguez et al., 2013). Baru-baru ini studi komputasi pada pasca-translasi modifikasi protein telah menarik banyak perhatian. Modifikasi PTM yaitu terdiri dari berbagai macam seperti, fosforilasi, glikosilasi, ubiquitinasi, nitrosilasi, metilasi, asetilasi, dan lipidasi (Qiu et al., 2016).

Sebagai salah satu modifikasi pasca-translasi yang paling umum yaitu, modifikasi asetilasi. Asetilasi adalah salah satu modifikasi protein pasca-translasi yang paling signifikan, dan memainkan peran penting dalam berbagai proses seluler, seperti *cytokine signaling, transcriptional regulation* dan *apoptosis.* Asetilasi biasanya terjadi pada residu lisin yang menjelaskan proses memasukkan gugus asetil (CH3CO) ke dalam rantai samping asam amino dalam protein. Reaksi ini merupakan modifikasi reversibel yang sangat bergantung pada berbagai enzim. Asetilasi pada protein lisin adalah modifikasi besar pasca-translasi yang memainkan peran penting dalam berbagai proses biologis penting, seperti ekspresi gen, dan metabolisme (Wuyun, Zheng, Zhang, Ruan, & Hu, 2016). Asetilasi pada protein lisin dikatalisasi oleh *histone acetyltransferases* atau *lysine acetyltransferases* yang mentransfer gugus asetil ke gugus epsilon-amino residu lisin, sedangkan deasetilasi lisin oleh *histone deacetylases* atau *lysine deacetylases* yang menghilangkan gugus asetil (Y. Li et al., 2014).

Penelitian tentang komputasi pada PTM, banyak algoritma pembelajaran mesin untuk klasifikasi yang telah digunakan seperti, *Neural Network*, *Bayesian Algorithm*, *Random Forest,* dan *Support Vector Machine* (SVM). Klasifikasi adalah salah satu tugas yang paling penting untuk aplikasi seperti kategorisasi teks, klasifikasi citra, ekspresi gen, prediksi struktur protein, dan lain-lain. Dibandingkan dengan algoritma lain, klasifikasi SVM seringkali lebih baik untuk memprediksi modifikasi pasca-translasi pada protein. Metode SVM pertama diusulkan oleh Vapnik pada tahun 1995, dan sejak itu minat ketertarikan semakin tinggi dalam penelitian pembelajaran mesin dikarenakan metode SVM tidak mengalami keterbatasan dimensi data dan sampel terbatas. SVM memiliki

banyak fitur penting yang didukung untuk menghasilkan hasil empiris yang baik dan mendukung untuk klasifikasi dan estimasi fungsi non-linear (Ansari & Sutar, 2015). Contoh alat berbasis SVM termasuk MeMo, alat web untuk prediksi metilasi, SUMOsp, server web untuk prediksi situs sumoylation, dan PredPhospho yang memprediksi situs fosforilasi. Penggunaan fitur-fitur yang efektif telah semakin mempercepat pengembangan model prediksi PTM (S. Li et al., 2009).

Support Vector Machines (SVM) adalah metode *machine learning* yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah class pada *input space*. SVM merupakan sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur berdimensi tinggi, dilatih dengan algoritma pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning bias* yang berasal dari teori pembelajaran statistik (Cortes & Vapnik, 1995).

Penelitian ini, difokuskan pada dasar-dasar teori dari metode *Support Vector Machine* sebagai salah satu topik menarik yang tengah hangat dibicarakan dalam dunia *computer science.* Kelebihan SVM dibandingkan metode yang lain terletak pada kemampuannya untuk menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *feature space* yang ditunjang oleh strategi *Structural Risk Minimization* (SRM). Pernyataan yang dihasilkan oleh Vapnik adalah semakin luasnya penelitian yang membuktikan kehandalan SVM dari sudut teori maupun aplikasi, dimana salah satu aplikasinya adalah dalam bidang bioinformatika. Bioinformatika adalah suatu disiplin yang mengawinkan

teknologi informasi dan teknologi biologi, untuk menjawab permasalahan kompleks dalam bidang biologi (Nugroho, Witarto, & Handoko, 2004).

Penelitian menggunakan laboratorium basa untuk prediksi data protein yang cenderung memerlukan waktu yang lama, peralatan laboratorium yang mahal, dan peralatan yang cukup rumit, sehingga pada penelitian ini akan menggunakan aplikasi pada metode *Support Vector Machine* pada bidang bioinformatika khususnya pada data protein. Penelitian terdahulu yang dilakukan oleh Lee, dkk pada tahun 2010 yang berjudul *Using Solvent Accessibility and Physicochemical Properties to Identify Protein N-Acetylation Sites* digunakan penulis sebagai acuan dalam penelitian ini dengan menggunakan data yang sama dan *feature extraction* yang berbeda untuk meningkatkan akurasi prediksi asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine*. Dengan demikian, penulis akan melakukan penelitian yang berjudul Prediksi Asetilasi Pada *Sequence* Protein Lisin menggunakan *Support Vector Machine*.

B. Rumusan Masalah

Adapun rumusan masalah dalam penelitian ini adalah sebagai berikut:

1. Bagaimana hasil prediksi asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine.*
2. Bagaimana perbandingan hasil kinerja prediksi asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine* pada tiga kernel yaitu Linear, Polynomial, dan Gaussian.
3. Mengidentifikasi eksperimental situs asetilasi protein menggunakan laboratorium basa yang memerlukan waktu yang lama.

C. Tujuan Penelitian

Adapun tujuan pada penelitian ini, yaitu sebagai berikut:

1. Mengklasifikasikan dan mengukur hasil kinerja data asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine* dengan 3 kernel yaitu Linear, Polynomial, dan Gaussian.
2. Membandingkan hasil kinerja klasifikasi asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine* dengan penelitian sebelumnya oleh Lee, dkk pada tahun 2010 yang berjudul *Using Solvent Accessibility and Physicochemical Properties to Identify Protein N-Acetylation Sites*.

D. Manfaat Penelitian

Adapun manfaat penelitian ini adalah sebagai berikut:

1. Menambah pengetahuan tentang pengklasifikasian menggunakan metode *Support Vector Machine.*
2. Menambah pengetahuan mengatasi *feature extraction* yang cocok guna klasifikasi data protein.

E. Batasan Masalah

Adapun batasan masalah penelitian ini adalah sebagai berikut:

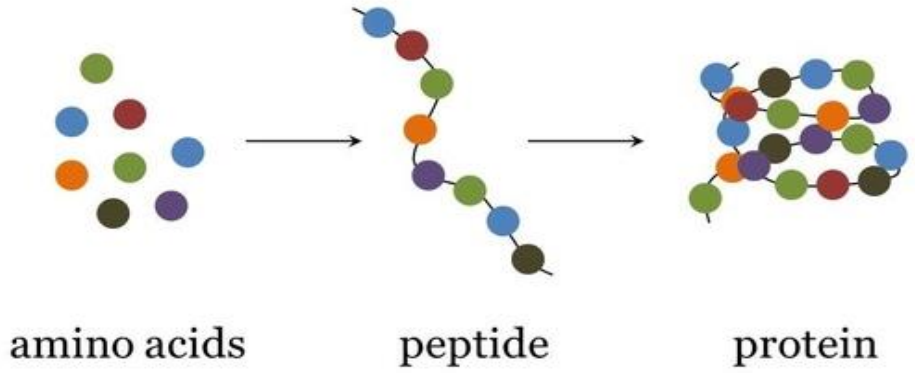
1. Penggunaan metode yang dipakai hanya menggunakan metode *Support Vector Machine* dan hanya tiga kernel *Support Vector Machine* yang digunakan yaitu Linear, Polynomial, dan Gaussian.

2. Lima *feature extraction* yang digunakan pada penelitian ini yaitu menggunakan *package* protr dan *package BioSeqClass.*
3. Berfokus pada protein Lisin.
4. Data yang digunakan yaitu data protein yang didapat dari *study literature*.

II. TINJAUAN PUSTAKA

A. Protein

Protein terdiri atas rantai-rantai panjang asam amino, yang terikat satu sama lain dalam ikatan peptida. Protein adalah polimer yang berfungsi sebagai penyususn protoplasma dan struktur tubuh lainnya, yang dapat berupa enzim atau hormon. Mekanisme sintesis pada protein terjadi melalui dua tahap utama yaitu transkripsi dan translasi (Jamilah et al., 2009).



Gambar.1 Ilustrasi Terjadinya Protein (Jamilah et al., 2009).

Transkripsi adalah pencetakan mRNA oleh DNA, sedangkan translasi adalah penerjemahan kode oleh tRNA berupa urutan yang dikehendaki. Translasi pada sintesis protein mengacu pada fase perakitan protein dalam sel yang melibatkan ribosom dimana RNA diterjemahkan untuk menghasilkan rantai asam

amino.Translasi bukan akhir jalur ekspresi genom. Polipeptida hasil translasi tidak langsung aktif, untuk menjadi protein aktif atau fungsional dalam sel dapat dilakukan modifikasi pasca-translasi atau yang sering disebut dengan *Post-Translational Modification* (PTM). Modifikasi pasca-translasi adalah perubahan yang terjadi pada struktur protein setelah menyelesaikan dan pelepasan polipeptida dari ribosom (Jamilah et al., 2009).

B. *Post-Tanslational Modification* (PTM)

Modifikasi pasca-translasi terjadi hampir pada semua protein dan memainkan peran penting dalam berbagai proses biologis dengan secara signifikan mempengaruhi struktur dan dinamika protein. Tempat terjadinya pasca-translasi yaitu terjadi di ribosom. Beberapa pendekatan komputasi telah dikembangkan untuk mempelajari PTM (fosforilasi, glikosilasi, ubiquitinasi, nitrosilasi, metilasi, asetilasi, lipidasi dan proteolisis) untuk menunjukkan pentingnya teknik dalam memprediksi situs yang dimodifikasi agar dapat diteliti lebih lanjut dengan pendekatan eksperimental (Audagnotto & Dal Peraro, 2017).

Modifikasi pasca-translasi (PTM) terjadi pada sejumlah besar protein *de facto* untuk meningkatkan kompleksitas sebenarnya dari proteom. PTM terdiri dalam modifikasi kovalen asam amino dari urutan protein utama dan memiliki efek untuk menciptakan banyak susunan yang lebih besar dari spesies protein yang bisa digunakan. Menanggapi persyaratan fisiologis spesifik, PTM memainkan peran penting dalam mengatur banyak fungsi biologis, seperti protein lokalisasi dalam sel, stabilitas protein, dan regulasi aktivitas enzimatik (Audagnotto & Dal Peraro, 2017).

C. Asetilasi

Asetilasi adalah PTM reversibel dengan peran yang terkenal dalam mengatur ekspresi gen melalui modifikasi ekor inti histon. Laporan berbagai individu menunjukkan bahwa asetilasi juga terlibat dalam beragam proses biologis lainnya yang menunjukkan fungsi pengaturan yang lebih luas (Gnad, 2010). Asetilasi lisin merupakan jenis penting dari modifikasi pasca-translasi reversibel (PTM) yang terjadi pada kelompok amino residu lisin dalam protein. Regulasi asetilasi lisin diaktifkan oleh sistem enzim yang sangat seimbang. Dalam sistem ini, *lysine acetyltransferases* mentransfer grup asetil ke grup amino lisin, sedangkan *lysine deacetylases* menghilangkan gugus asetil ini. Untuk memahami sepenuhnya mekanisme asetilasi, identifikasi substrat dan situs asetilasi spesifik sangat penting (Y. Li et al., 2014).

Asetilasi lisin adalah Post-Translasional Modifikasi (PTM) dinamis dan reversibel yang sangat terkonsentrasi pada prokariota dan eukariota. Proses ini menetralkan muatan positif pada asam amino dan mengatur ikatan DNA, interaksi protein-protein, dan stabilitas protein. Selain itu, asetilasi lisin terlibat dalam beragam konsekuensi biologis termasuk aktivitas transkripsi, kelangsungan hidup sel, dan lokalisasi subseluler. Yang paling penting, telah dilaporkan bahwa asetilasi lisin yang menyimpang terkait dengan banyak penyakit patologis, seperti kanker, penyakit neurodegeneratif, dan penyakit metabolisme (Lu, Lee, Chen, & Chen, 2014).

D. *SkipRedundant Sequence* Protein

Terdapat banyak *database squence* protein yang sangat redundan dan redundansi mereka harus dihilangkan dalam banyak studi berbeda. Redundansi

dalam *dataset* terjadi ketika beberapa data serupa hadir pada saat yang sama. Dalam bioinformatika, redundansi dalam sekumpulan *sequence* terjadi ketika satu atau lebih *sequence* serupa ada pada set data yang sama. Adanya *sequence* protein yang sama dalam analisis tertentu akan muncul bias yang tidak diinginkan, *Dataset* protein non-redundan sangat penting dalam bioinformatika. *Dataset* yang baik, berarti menghilangkan *sequence* protein yang melampaui ambang batas kesamaan tertentu Beberapa program seperti *Decrease Redundancy, cd-hit, Pisces, BlastClust,* dan *SkipRedundant* sudah tersedia. Pada penelitian ini, *dataset* yang akan digunakan dilakukan pembersihan redundansi dengan menggunakan *SkipRedundant* (Sikic & Carugo, 2010).

E. *Feature Extraction*

Ekstraksi fitur berfungsi untuk menghilangkan redundansi data dan mengambil data penting dari sebuah data mentah. Ekstraksi fitur juga dapat didefinisikan sebagai ekstraksi informasi yang paling representatif dari data mentah, yang meminimalkan variabilitas pola di dalam kelas. Tugas dasar dari ekstraksi fitur adalah untuk mencari sekelompok fitur yang paling efektif untuk diklasifikasikan yaitu, mengkoversi dari ruang fitur berdimensi tinggi hingga ruang fitur dimensi rendah, sehingga dapat merancang secara efektif (‘Arif, Hassan, Nasien, & Haron, 2015).

Ekstraksi fitur melakukan beberapa transformasi fitur asli untuk menghasilkan fitur lain yang lebih signifikan. Ekstraksi fitur dapat digunakan dalam konteks untuk mengurangi kompleksitas dan memberikan representasi sederhana dari data yang mewakili setiap variabel dalam ruang fitur, sebagai kombinasi linear dari variabel input asli (Khalid, Khalil, & Nasreen, 2014).

Ekstraksi fitur adalah sebuah langkah penting dalam pembangunan klasifikasi pola dan bertujuan mengekstraksi informasi yang relevan agar menjadi ciri masing-masing kelas. Dalam proses ini, fitur yang relevan diekstraksi dari objek untuk membentuk vektor fitur. Vektor fitur ini kemudian digunakan oleh pengklasifikasi untuk mengenali unit input dengan target unit output. Proses ini menjadi lebih mudah bagi pengklasifikasi untuk mengklasifikasikan antara kelas yang berbeda dengan melihat fitur-fitur karena memungkinkan cukup mudah dibedakan (Medhi et al., 2016).

Strategi ekstraksi fitur yaitu mengekstraksi komponen yang berbeda dari gambar seperti tepi, sudut dan sebagainya yang dapat digunakan untuk mencocokkan kesamaan sebagai estimasi transformasi relatif antar gambar. Maka, dalam ekstraksi fitur ini memfokuskan pada berbagai metode yang digunakan untuk ekstraksi fitur gambar, ekstraksi fitur tekstur dan ekstraksi fitur bentuk. Pada fase ekstraksi fitur, setiap karakter diwakili oleh vektor fitur yang menjadi identitasnya. Tujuan utama ekstraksi fitur adalah untuk mengekstraksi satu set fitur, yang memaksimalkan tingkat pengenalan dengan jumlah elemen terkecil dan untuk menghasilkan kumpulan fitur serupa untuk berbagai simbol yang sama (Medhi et al., 2016).

Dalam memprediksi berbagai atribut penting dari protein, banyak deskriptor yang berbeda untuk mewakili urutan sampel yang telah dikembangkan dan digunakan secara luas. Ekstraksi fitur yang digunakan dalam penelitian ini yaitu menggunakan protein deskriptor. Protein deskriptor yang dipakai yaitu *package* protr dan *package BioSeqClass* yang terdapat di dalam R *Programming*. *Package BioSeqClass* merupakan salah satu package ekstraksi fitur dari

Biological Sequences yang ada pada bahasa pemprograman R. Sedangkan *package* protr bertujuan untuk ekstraksi fitur dari data asli menjadi data dalam urutan protein (*sequence protein)* sehingga dapat dengan mudah diterapkan dalam penelitian di bidang bioinformatika. Secara umum, setiap jenis deskriptor dapat di ekstraksi dengan fungsi bernama *extractX()* dalam paket protr, di mana *X* adalah singkatan dari nama deskriptor (Xiao, Cao, & Xu, 2014). Berikut adalah Tabel 1 yaitu, jumlah dimensi dari setiap grup dekriptor protein pada *package* protr.

Tabel.1 Jumlah Dimensi Grup Deskriptor Protein (Xiao, Cao, Zhu, & Xu, 2015).

Descriptor Group	***Descriptor Name***	***Descriptor Dimension***
Amino Acid Composition	*Amino Acid Composition*	20
	Dipeptide Composition	400
	Tripeptide Composition	8000
Autocorrelation	*Normalized Moreau-Broto Autocorrelation*	240[1]
	Moran Autocorrelation	240[1]
	Geary Autocorrelation	240[1]
CTD	*Composition*	21
	Transition	21
	Distribution	105
Conjoint Triad	*Conjoint Triad*	343
Quasi-Sequence-Order	*Sequence-Order-Coupling Number*	60[2]
	Quasi-Sequence-Order Descriptor	100[2]
Pseudo-Amino Acid Composition	*Pseudo-Amino Acid Composition*	50[3]
	Amphiphilic Pseudo-Amino Acid Composition	804
Position Specific Scoring Matrix	*Position Specific Scoring Matrix*	20 x string

Adapun *package* yang digunakan pada tahap *feature extraction* dalam penelitian ini yaitu sebagai berikut:

1. APAAC (*Amphiphilic Pseudo Amino Acid Composition)*

   Pada *feature extraction* APAAC ini menggunakan *package* protr. *Feature extraction* APAAC berfungsi untuk menghitung Komposisi Asam Amino Pseudo Amino Amfifilik. Rumus dari *feature extraction* APAAC ini adalah (20 + (n * lambda)) yang sudah sesuai dengan ketentuan yang terdapat pada *package feature extraction* APAAC. Nilai n adalah 2 dan lambda adalah panjang *sequence* protein dikurang 1 pada masing-masing kelas data. Jadi, jika nilai tersebut dimasukkan dan dihitung (20+(2*(21-1)) maka didapatkan jumlah dimensi *feature extraction* APAAC yaitu 60.

   Dalam penelitian ini mengunakan *library* protr yang digunakan untuk menghitung dan mendapatkan hasil dari *feature extraction* APAAC. Jika diuraikan *library* protr pada *extractAPAAC* akan terlihat seperti pada Kode Program 1.

```
function (x, props = c("Hydrophobicity", "Hydrophilicity"), lambd
     a = 30, w = 0.05, customprops = NULL)
{
    if (protcheck(x) == FALSE) {
        stop("x has unrecognized amino acid type")
    }
    if (nchar(x) <= lambda) {
        stop("Length of the protein sequence must be greater than
          \"lambda\"")
    }
    AAidx <- read.csv(system.file("sysdata/AAidx.csv",
        package = "protr"), header = TRUE)
    tmp <- data.frame(AccNo = c("Hydrophobicity", "Hydrophilicity
          ",
        "SideChainMass"),
          A = c(0.62, -0.5, 15), R = c(-2.53, 3, 101), N = c(-0.7
          8, 0.2, 58), D = c(-0.9, 3, 59), C = c(0.29, -1, 47), E
          = c(-0.74, 3, 73), Q = c(-0.85, 0.2, 72),
         G = c(0.48, 0, 1), H = c(-0.4, -0.5, 82), I = c(1.38, -1
         .8, 57), L = c(1.06, -1.8, 57), K = c(-1.5, 3, 73), M =
         c(0.64, -1.3, 75), F = c(1.19, -2.5, 91),
          P = c(0.12, 0, 42), S = c(-0.18, 0.3, 31), T = c(-0.05,
          -0.4, 45), W = c(0.81, -3.4, 130), Y = c(0.26, -2.3,
            107), V = c(1.08, -1.5, 43))
    AAidx <- rbind(AAidx, tmp)
    if (!is.null(customprops))
        AAidx <- rbind(AAidx, customprops)
    aaidx <- AAidx[, -1]
    row.names(aaidx) <- AAidx[, 1]
    n <- length(props)
    H0 <- as.matrix(aaidx[props, ])
    H <- matrix(ncol = 20, nrow = n)
    for (i in 1:n) {
        H[i, ] <- (H0[i, ] - mean(H0[i, ]))/(sqrt(sum((H0[i,
            ] - mean(H0[i, ]))^2)/20))
    }
    AADict <- c("A", "R", "N", "D", "C",
        "E", "Q", "G", "H", "I",
        "L", "K", "M", "F", "P",
        "S", "T", "W", "Y", "V")
    dimnames(H) <- list(props, AADict)
    Theta <- vector("list", lambda)
    for (i in 1:lambda) Theta[[i]] <- vector("list", n)
    xSplitted <- strsplit(x, split = "")[[1]]
    N <- length(xSplitted)
    for (i in 1:lambda) {
        for (j in 1:n) {
            for (k in 1:(N - i)) {
                Theta[[i]][[j]][k] <- H[props[j], xSplitted[k]] *
                  H[props[j], xSplitted[k + i]]
            }
        }
    }
    tau <- sapply(unlist(Theta, recursive = FALSE), mean)
    fc <- summary(factor(xSplitted, levels = AADict), maxsum = 21
     )
    Pc1 <- fc/(1 + (w * sum(tau)))
    names(Pc1) <- paste("Pc1.", names(Pc1), sep = "")
    Pc2 <- (w * tau)/(1 + (w * sum(tau)))
    names(Pc2) <- paste("Pc2", as.vector(outer(props, 1:lambda, p
     aste, sep = ".")), sep = ".")
    Pc <- c(Pc1, Pc2)
    Pc
}
```

Kode Program.1 *Code* Implementasi *extractAPAAC* R Studio.

2. *AAIndex*

Feature extraction AAIndex berfungsi untuk mengembalikan matrik yang mengukur sifat fisikokimia dan biokimia asam amino yang jumlah dimeninya yaitu 21 variabel karena panjang *sequence* pada penelitian ini yaitu 21. Dalam penelitian ini mengunakan *library BioSeqClass* yang digunakan untuk menghitung dan mendapatkan hasil dari *feature extraction AAIndex*. Jika diuraikan *library BioSeqClass* pada *featureAAIndex* akan terlihat pada Kode Program 2.

```
function (seq, aaindex.name = "all")
{
    L = unique(sapply(seq, function(x) {
        length(unlist(strsplit(x, split = "")))
    }))
    if (length(L) > 1) {
        stop("Sequences in seq must have equal length") }
    data(aa.index)
    if (aaindex.name == "all") {
        index = sapply(aa.index, function(x) {
            x$I })
        index = index[, apply(index, 2, function(x) {
            sum(is.na(x)) == 0})]
        name = colnames(index)
        indexFeature = sapply(seq, function(x) {
            x2 = unlist(strsplit(x, split = ""))
            x3 = as.vector(apply(index, 2, function(y) {
                y[x2]
            }))
            names(x3) = paste(rep(name, each = length(x2)), rep(
            1:length(x2),
                ncol(index)), sep = "_")
            x3})}
    else {
        index = aa.index[[aaindex.name]]$I
        if (sum(is.na(index)) > 0) {
            stop(paste("aaindex.name", aaindex.name, "is not the
        property name in \n        AAindex or has NA value"))
        }
        indexFeature = sapply(seq, function(x) {
            x2 = unlist(strsplit(x, split = ""))
            x3 = index[x2]
            names(x3) = paste(aaindex.name, 1:length(x2), sep =
            "_")
            x3 })
    }
    indexFeature = t(indexFeature)
    colnames(indexFeature) = paste("AAindex:", colnames(indexFea
    ture),
        sep = "")
    indexFeature
}
```

Kode Program.2 *Code* Implementasi *featureAAIndex* R Studio.

3. CTD (*Composition, Transition, Distribution)*

   Dalam tahap ini menggunakan 3 jenis *feature extraction* CTD yaitu *Composition* (C), *Transition* (T) dan Distribution (D). *Composition* (C) merupakan jumlah asam amino dari sifat tertentu dibagi dengan jumlah total asam amino. *Transition* (T) mencirikan frekuensi persen asam amino dari sifat tertentu diikuti oleh asam amino dari sifat yang berbeda dan *Distribution* (D) mengukur panjang rantai di mana asam pertama 25, 50, 75 dan 100 yang terletak pada posisi masing-masing. Terdapat 21 dimensi pada *feature extraction* ini.

   Pada tahap ini dilakukan *feature extraction Composition, Transition,* dan *Distribution* menggunakan *library Bioseqclass.* Jika diuraikan *library BioSeqClass* pada *featureCTD* akan terlihat seperti pada Kode Program 3.

```
function (seq, class = elements("aminoacid"))
{
    k = 0
    pair = vector()
    for (i in 1:(length(class) - 1)) {
        for (j in (i + 1):length(class)) {
            k = k + 1
            pair[k] <- paste(names(class)[i], names(class)[j],
                sep = "_")
        }
    }
    name <- c(paste("CTD:C", names(class), sep = "_"),
        paste("CTD:T", pair, sep = "_"), paste("CTD:D",
            rep(names(class), each = 5), c("1st", "25%",
                "50%", "75%", "100%"), sep = "_"))
    binary2 <- rep(names(class), sapply(class, length))
    names(binary2) <- unlist(class)
    ctd <- sapply(seq, function(x) {
        y = binary2[unlist(strsplit(x, split = ""))]
        z = rep(0, length = length(name))
        names(z) = name
        tmp = table(y)/length(y)
        z[paste("CTD:C", names(tmp), sep = "_")] = tmp
        tmp1 = table(sapply(1:(length(y) - 1), function(i) {
            paste(y[i], y[i + 1], sep = "_")
        }))
        tmp2 = table(sapply(length(y):2, function(i) {
            paste(y[i], y[i - 1], sep = "_")
        }))
        if (length(intersect(pair, names(tmp1))) > 0) {
            z[paste("CTD:T", intersect(pair, names(tmp1)),
                sep = "_")] = tmp1[intersect(pair, names(tmp1))]
               /sum(tmp1)
        }
        if (length(intersect(pair, names(tmp2))) > 0) {
            z[paste("CTD:T", intersect(pair, names(tmp2)),
                sep = "_")] = z[paste("CTD:T", intersect(pair,
                names(tmp2)), sep = "_")] + tmp2[intersect(pair,
                names(tmp2))]/sum(tmp2)
        }
        for (c in unique(y)) {
            tmp = (1:length(y))[y == c]
            z[paste("CTD:D", rep(c, each = 5), c("1st",
                "25%", "50%", "75%", "100%"),
                sep = "_")] = c(tmp[1], tmp[ceiling(length(tmp)
               *
                c(0.25, 0.5, 0.75, 1))])/length(y)
        }
        z
    })
    t(ctd)
}
```

Kode Program.3 *Code* Implementasi *feature*CTD R Studio.

4. *Hydrophobicity*

Feature extraction Hydrophobicity merupakan urutan protein diurutkan berdasarkan hidrofobitasnya. Pada *feature extraction Hydrophobicity* ini

menggunakan *featureHydro* yang berguna untuk mengembalikan matriks yang mengukur efek hidrofobik. Parameter "*hydro.method*" mendukung metode pengkodean untuk menunjukkan efek hidrofobik dari asam amino. Setiap urutan dikodekan oleh vektor numerik dimensi N. Variabel N merupakan panjang dari *sequence* protein yang digunakan dalam penelitian ini. Terdapat 21 dimensi pada *feature extraction Hydrophobicity.*

Feature extraction Hydrophobicity menggunakan *library Bioseqclass.* Jika diuraikan *library BioSeqClass* pada *featureHydro* akan terlihat seperti pada Kode Program 4.

```
function (seq, hydro.method = "SARAH1")
{
    L = unique(sapply(seq, function(x) {
        length(unlist(strsplit(x, split = "")))
    }))
    if (length(L) > 1) {
        stop("Sequences in seq must have equal length")   }
    if (hydro.method == "kpm") {
        H = c(4.92, 4.92, 4.04, 4.04, 2.98, 2.35, 2.33, 1.81,
            1.28, 0.94, -0.14, -2.57, -3.4, -4.66, -5.54, -5.55,
            -6.64, -6.81, -8.72, -14.92)
        names(H) = c("I", "L", "V", "P",
            "F", "M", "W", "A", "C",
            "G", "Y", "T", "S", "H",
            "Q", "K", "N", "E", "D",
            "R")
        hydro = sapply(seq, function(x) {
            H[unlist(strsplit(x, split = ""))] })
        rownames(hydro) = paste("H:", 1:L, sep = "")}
    if (hydro.method == "SARAH1") {
        H = list(c(1, 1, 0, 0, 0), c(1, 0, 1, 0, 0), c(1, 0,
            0, 1, 0), c(1, 0, 0, 0, 1), c(0, 1, 1, 0, 0), c(0,
            1, 0, 1, 0), c(0, 1, 0, 0, 1), c(0, 0, 1, 1, 0),
            c(0, 0, 1, 0, 1), c(0, 0, 0, 1, 1), c(0, 0, 0, -1,
                -1), c(0, 0, -1, 0, -1), c(0, 0, -1, -1, 0),
            c(0, -1, 0, 0, -1), c(0, -1, 0, -1, 0), c(0, -1,
                -1, 0, 0), c(-1, 0, 0, 0, -1), c(-1, 0, 0, -1,
                0), c(-1, 0, -1, 0, 0), c(-1, -1, 0, 0, 0))
        names(H) = c("C", "F", "I", "V",
            "L", "W", "M", "H", "Y",
            "A", "G", "T", "S", "R",
            "P", "N", "D", "Q", "E",
            "K")
        hydro = sapply(seq, function(x) {
            unlist(H[unlist(strsplit(x, split = ""))])})
        rownames(hydro) = paste("H:", paste(rep(1:L, each = 5),
            1:5, sep = "_"), sep = "")  }
    t(hydro)
}
```

Kode Program.4 *Code* Implementasi *featureHydro* R Studio.

F. Klasifikasi

Klasifikasi adalah suatu metode analisis data dengan menentukan model yang menggambarkan *class* dari kumpulan data yang penting untuk menempatkan suatu objek pada suatu kategori. Modelnya dapat diambil berdasarkan analisis dari kumpulan *training data,* contohnya objek data yang *class*-nya sudah diketahui (Bui, Del Fiol, & Jonnalagadda, 2016) .

Kesalahan pada klasifikasi akan menyebabkan kesalahan pada hasil yang dikeluarkan. Sehingga pada bagian klasifikasi, banyak metoe-metode yag dikembangkan para peneliti seperti, *K-Nearest Neighbo*r (KNN), *Neural Network* (NN), *Support Vector Machine* (SVM), *Random Florest*. Kemudian hasil dari klasifikasi akan disimpan dan menjadi penentu untuk klasifikasi selanjutnya.

1. *K-Nearest Neighbo*r (KNN)

   K–Nearest Neighbor adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut. *K-Nearest Neighbor* termasuk algoritma *unsupervised learning* dimana hasil dari *query instance* yang baru akan diklasifikan berdasarkan mayoritas dari kategori pada *K-Nearest Neighbor* yang nantinya kelas yang paling banyak munculah yang akan menjadi kelas hasil klasifikasi (Wijaya, Chamidah, & Santoni, 2019).

2. *Artificial Neural Network* (ANN)

   Neural network merupakan sebuah metode sistem pembelajaran terhadap penerimaan informasi yang memiliki kinerja layaknya sebuah jaringan syaraf pada manusia (Dongare, Kharde, & Kachare, 2012).

   Karakteristik *Artificial Neural Network*:

   a. *Artificial Neural Network* dapat memetakan pola input ke pola output terkaitnya.
   b. *Artificial Neural Network* dapat mengidentifikasi objek baru yang sebelumnya tidak terlatih.
   c. *Artificial Neural Network* memiliki kemampuan untuk menggeneralisasi.
   d. *Artificial Neural Network* adalah sistem yang kuat dan toleran terhadap kesalahan.
   e. *Artificial Neural Network* dapat memproses informasi secara paralel, dengan kecepatan tinggi, dan secara terdistribusi.

3. *Support Vector Machine* (SVM).

   Support Vector Machine (SVM) adalah metode *machine learning* yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *input space*. Salah satu kelebihan yang dimiliki metode SVM yaitu menggunakan prinsip SRM ysng digunakan untuk penanganan *error* pada set data training. SRM dikatakan lebih baik karena tidak hanya meminimalkan *error* yang terjadi, tetapi meminimalkan faktor-faktor lainnya. SVM merupakan sistem pembelajaran yang menggunakan ruang hipotesis berupa fungsi-fungsi linier dalam sebuah ruang fitur berdimensi tinggi, dilatih dengan algoritma

pembelajaran yang didasarkan pada teori optimasi dengan mengimplementasikan *learning bias* yang berasal dari teori pembelajaran statistik (Cortes & Vapnik, 1995).

Support Vector Machine (SVM) merupakan metode klasifikasi yang diperkenalkan pertama kali oleh Vapnik pada tahun 1995. Pada dasarnya, metode ini bekerja dengan cara mendefinisikan batas antara dua kelas dengan jarak maksimal dari data yang terdekat. Untuk mendapatkan batas maksimal antar kelas maka harus dibentuk sebuah *hyperplane* (garis pemisah) terbaik pada input *space* yang diperoleh dengan mengukur margin hyperplane dan mencari titik maksimalnya. Margin merupakan jarak antara *hyperplane* dengan titik terdekat dari masing-masing kelas. Titik terdekat inilah yang disebut sebagai *support vector* (Bhavsar & Panchal, 2012).

Berikut adalah beberapa jenis fungsi kernel yang digunakan dalam penelitian ini:

a. Kernel Linear

   Kernel Linear merupakan fungsi kernel pada SVM paling sederhana. Kernel ini sering digunakan pada data set yang sebaran datanya dapat diklasifikasikan dengan cara linear. Fungsi kernel SVM Linear dapat dilihat pada persamaan 1.

   $$K(x_i, x_j) = x_i{}^T \cdot x_j + C \quad \text{.....(1)}$$

   Dengan x_i dan x_j merupakan vektor dari data set dan C adalah *constant*.

   Dalam penelitian ini, mengunakan *library* dari kernel Linear, yaitu

metodeSVM = "svmLinear". Jika diuraikan *library* kernel Linear akan terlihat seperti pada Kode Program 5.

```
# parameters needed for linear kernel
  if (kernel.function == "linear" & !is.null(rho)) {
    rho <- NULL

  warning("rho = ", call.names[call.order[4]], " is not used
with linear
kernel")
  }
  if (kernel.function == "linear" & !is.null(gamma)) {
    gamma <- NULL
 warning("gamma = ", call.names[call.order[5]], " is not used
with linear
kernel")
  }
  if (kernel.function == "linear" & !is.null(d)) {
    d <- NULL
    warning("d = ", call.names[call.order[6]], " is not used
with linear
kernel")

  }
```

Kode Program.5 *Code* Implementasi Kernel Linear R Studio.

b. Kernel Polynomial

Persamaan Polynomial merupakan sebuah persamaan yang terdiri dari variabel dan koefisien yang memiliki suku banyak. Kernel Polynomial pada SVM sering digunakan dalam masalah dimana semua *training* data nya dinormalisasi. Fungsi kernel SVM Polynomial dapat dilihat pada persamaan 2.

$$K(x_i, x_j) = (\gamma {x_i}^T \cdot x_j + r)^d, \gamma > 0 \quad \text{.....(2)}$$

Dengan x_i dan $x_j$merupakan vektor dari data set, *r*, γ adalah parameter untuk mengkontrol kecepatan proses *learning*, dan *d* merupakan pangkat polynomial yang digunakan.

Dalam penelitian ini mengunakan *library* dari kernel Polynomial, yaitu metodeSVM = "svmPolynomial". Jika diuraikan *library* kernel Polynomial akan terlihat seperti pada Kode Program 6.

```
# parameters needed for polynomial kernel
  if (kernel.function == "polynomial" & !is.null(rho)) {
    if (rho <= 0) {
      stop("rho should be positive")
    }
  }

if (kernel.function == "polynomial" & !is.null(d)) {
    if (kernel.function == "polynomial" & (d <= 0 |
!check.integer(d))) {
      stop("d should be an integer > 0")
    }
  }
  if (kernel.function == "polynomial" & is.null(d)) {
    stop("d is missing")
  }
```

Kode Program.6 *Code* Implementasi Kernel Polynomial R Studio.

c. Kernel Gaussian

Kernel Gaussian merupakan kernel pada SVM yang digunakan untuk menyelesaikan masalah yang tidak bisa diselesaikan dengan cara Linear. Fungsi kernel SVM Gaussian dapat dilihat pada persamaan 3.

$$K(x_i, x_j) = exp\ (-\gamma||x_i - x_j||^2), \gamma > 0 \quad \text{.....(3)}$$

Dengan x_i dan x_j merupakan vektor dari data set dan γ adalah parameter untuk mengkontrol kecepatan proses *learning* dan *exp* merupakan basis dari logaritma alami.

Dalam penelitian ini mengunakan *library* dari kernel Gaussian, yaitu metodeSVM = "svmRadial". Jika diuraikan *library* kernel Gaussian akan terlihat seperti pada Kode Program 7.

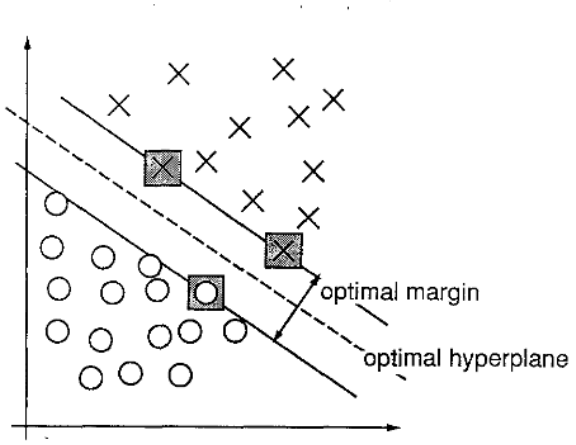
```
# parameters needed for gaussian kernel
  if (kernel.function == "gaussian" & !is.null(rho)) {
    if (rho <= 0) {
      stop("rho should be positive")
    }

}
  if (kernel.function == "gaussian" & !is.null(gamma)) {
    gamma <- NULL
    warning("gamma = ", call.names[call.order[5]], " is not
used with
gaussian kernel")
  }
  if (kernel.function == "gaussian" & !is.null(d)) {
    d <- NULL
    warning("d = ", call.names[call.order[6]], " is not used
with gaussian
kernel")
```

Kode Program.7 *Code* Implementasi Kernel Gaussian R Studio.

Support Vector Machine telah banyak diterapkan pada pola masalah klasifikasi dan regresi nonlinier. Untuk klasifikasi nonlinier, fungsi kernel sering digunakan untuk mengubah data input ke dimensi tinggi ruang fitur dimana data input menjadi lebih dapat dipisahkan dibandingkan dengan ruang input asli. *Support Vector Machine* telah banyak diterapkan untuk dua masalah klasifikasi pola pada bioinformatika. Salah satunya adalah diagnosis kanker berdasarkan ekspresi gen *microarray* data, dan yang lainnya adalah prediksi struktur sekunder dari protein (Ladwani, 2018).

Support Vector Machine digunakan sebagai rangkaian harmonis konsep-konsep unggulan dalam bidang *pattern recognition*. Sebagai salah satu metode pattern recognition, usia *Support Vector Machine* terbilang masih relatif muda. Walaupun demikian, evaluasi kemampuannya dalam berbagai aplikasi menempatkannya sebagai *state of the art* dalam *pattern recognition* dan merupakan salah satu tema yang berkembang dengan pesat (Nugroho et al., 2004).

Support Vector Machine adalah metode *machine learning* yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada input *space*. Berikut adalah Gambar.1 penentuan hyperplane pada SVM.



Gambar.2 Ilustrasi SVM 2 *Class* (Cortes & Vapnik, 1995).

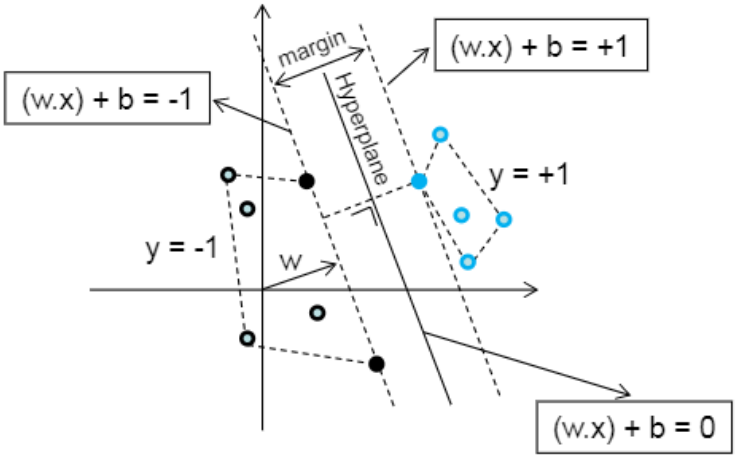
Gambar 2 memperlihatkan beberapa pattern yang merupakan anggota dari dua buah *class*. Permasalahan klasifikasi dapat diterjemahkan dengan usaha menemukan garis *(hyperplane*) yang memisahkan antara kedua kelompok tersebut. Berbagai alternatif garis pemisah *(discrimination boundaries)* ditunjukkan pada gambar 2. *Hyperplane* pemisah terbaik antara kedua class dapat ditemukan dengan mengukur margin *hyperplane* dan mencari titik optimum *hyperplane* tersebut. Margin adalah jarak antara *hyperplane* dengan *pattern* terdekat dari masing-masing *class*. *Pattern* yang paling dekat disebut sebagai *support vector*. Garis optimal *hyperplane* menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua *class*, sedangkan pada simbol dari masing-masing *class* data yaitu silang dan bulat

yang berada dalam kotak hitam tersebut adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM (Cortes & Vapnik, 1995).

Berbeda dengan strategi *neural network* yang berusaha mencari *hyperplane* pemisah antar *class*, *support vector machine* berusaha menemukan *hyperplane* yang terbaik pada *input space*. Prinsip dasar *support vector machine* adalah linear *classifier*, dan selanjutnya dikembangkan agar dapat bekerja pada problem non-linear dengan memasukkan konsep kernel *trick* pada ruang kerja berdimensi tinggi. Perkembangan ini memberikan rangsangan minat penelitian di bidang *pattern recognition* untuk investigasi potensi kemampuan *support vector machine* secara teoritis maupun dari segi aplikasi. *Support vector machine* telah berhasil diaplikasikan dalam problema dunia nyata, dan secara umum memberikan solusi yang lebih baik dibandingkan metode konvensional seperti *artificial neural network*. Kelebihan *support vector machine* dibandingkan metode yang lain terletak pada kemampuannya untuk menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *feature space* yang ditunjang oleh strategi *Structural Risk Minimization* (SRM) (Nugroho et al., 2004).

4. Pembahasan Mencari *Hyperplane Support Vector Machine*.

   Contoh ilustrasi mencari *hyperplane* dapat dilihat pada Gambar 3. ilustrasi mencari *hyperplane.*



Gambar 3. Ilustrasi Penjelasan *Hyperplane* (Cortes & Vapnik, 1995).

Dapat dilihat pada Gambar 3 diketahui bahwa terdapat 2 *class* data yaitu -1 dan 1, margin, serta *hyperplane* yang memisahkan dari kedua *class* data tersebut.

Tabel.2 Contoh Data Dengan 2 *Class.*

x_1	x_2	Y
1	1	1
1	-1	-1
-1	1	-1
-1	-1	-1

Karena terdapat 2 variabel data yaitu x_1 dan x_2, maka w juga akan memiliki fitur (w_1 dan w_2).

Formulasi yang digunakan untuk mencari *hyperplane* adalah sebagai berikut:

$$\frac{1}{2} \| w \|^2 = \frac{1}{2}({w_1}^2 + {w_2}^2) \quad \text{.....(4)}$$

Syarat:

$y_i\ (w\ .x_i + b) \geq 1$……(5)

Dengan $i = 1,2,3, \ldots, N$

Maka persamaan yang digunakan dengan 2 variabel yaitu:

$y_i\ (w_1.x_1 + w_2.x_2 + b) \geq 1$……..……...(6)

Sehingga didapatkan beberapa persamaan dari data pada Tabel 2, yaitu berikut:

$(w_1 + w_2 + b) \geq 1$……..……...(7)

Untuk $y_1 = 1,\ x_1 = 1,\ x_2 = 1$

$(-w_1 + w_2 - b) \geq 1$……..……(8)

Untuk $y_2 = -1,\ x_1 = 1,\ x_2 = -1$

$(w_1 - w_2 - b) \geq 1$……....(9)

Untuk $y_3 = -1,\ x_1 = -1,\ x_2 = 1$

$(w_1 + w_2 - b) \geq 1$……..…….(10)

Untuk $y_4 = -1,\ x_1 = -1,\ x_2 = -1$

Langkah selanjutnya yaitu:

a. Menjumlahkan persamaan (1) dan (2):

$(w_1 + w_2 + b) \geq 1$……..……(1)

$(-w_1 + w_2 - b) \geq 1$……(2) +

$2w_2 = 2$

Maka $w_2 = 1$

b. Menjumlahkan persamaan (1) dan (3):

$(w_1 + w_2 + b) \geq 1$……(1)

$(w_1 - w_2 - b) \geq 1$……(3) +

$2w_1 = 2$

Maka $w_1 = 1$

c. Menjumlahkan persamaan (2) dan (3):

$(-w_1 + w_2 - b) \geq 1$……..(2)

$(w_1 - w_2 - b) \geq 1$……..(3)

+

$-2b = 2$

Maka $b = -1$

Sehingga didapatkan persamaan *hyperplane*:

$w_1 x_1 + w_2 x_2 + b = 0$……..(11)

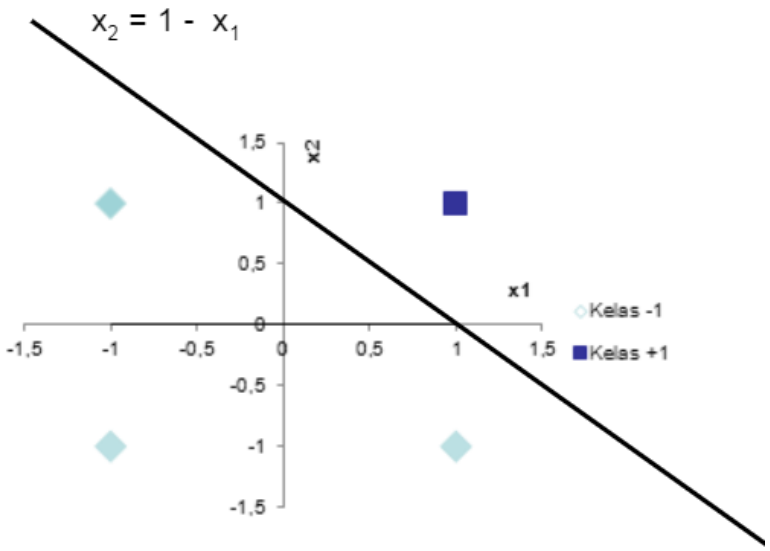
$x_1 + x_2 - 1 = 0$……..(12)

$x_2 = 1 - x_1$……..(13)

Maka hasil dari persamaan diatas, didapatkan visualisasi garis *hyperplane* yang dapat dilihat pada Tabel 3 dan Gambar 4 sebagai berikut:

Tabel.3 Hasil Visualisasi Garis *Hyperplane.*

x_1	$x_2 = 1 - x_1$
-2	3
-1	2
0	1
1	0
2	-1



Gambar 4. Hasil Visualisasi Garis *Hyperplane* (Cortes & Vapnik, 1995).

G. *Cross-Validation*

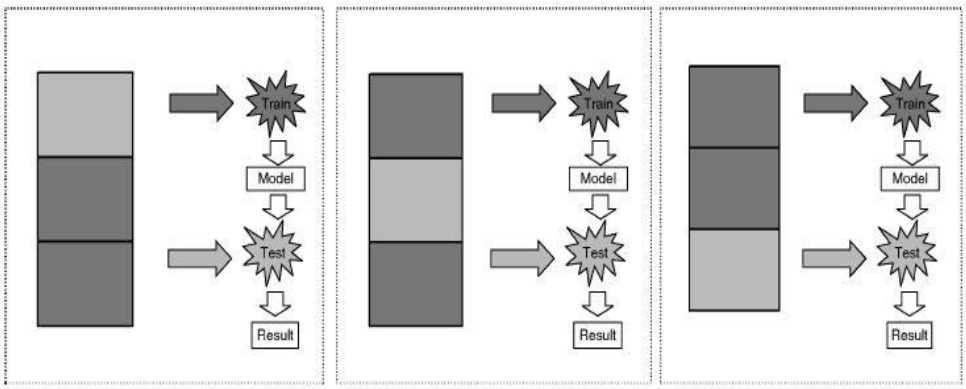
Cross-Validation adalah metode statistik untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua segmen yaitu, satu digunakan untuk belajar atau melatih model dan yang lain digunakan untuk memvalidasi model. Pada *cross-validation* yang khas, perangkat pelatihan dan validasi harus di *cross-over* dalam putaran berturut-turut sehingga setiap titik data memiliki peluang untuk divalidasi (Refaeilzadeh, Tang, & Liu 2008). *Cross-validation* meliputi:

1. *K-Fold Validation*

   Teknik *k-fold cross validation* adalah salah satu dari sekian pendekatan yang digunakan oleh praktisi untuk pemilihan model dan estimasi kesalahan pengklasifikasi. *K-fold cross validation* berfungsi membagi data set menjadi k himpunan bagian, kemudian dilakukan secara berulang sebanyak k iterasi. Beberapa dari data set digunakan untuk mempelajari model, sementara yang

lain dieksploitasi untuk menilai kinerjanya. Namun, terlepas dari keberhasilan *k-fold cross validation* hanya ada aturan praktis yang tersedia untuk memilih nomor dan kardinalitas dari himpunan bagian (Anguita, Ghelardoni, Ghio, Oneto, & Ridella, 2012).

K-fold cross validation merupakan improvisasi dari *holdout method.* Metode ini dilakukan dengan cara membagi *data set* menjadi *k subset* dan melakukan *holdout method* sebanyak *k* iterasi. Tiap iterasi, satu dari *k subsets* digunakan sebagai *testing set* dan *k-1 subsets* yang lain digunakan sebagai *training set*. Setelah itu rata-rata dari kesalahan semua percobaan dihitung. Keuntungan dari metode ini adalah pembagian *data set* tidak terlalu berpengaruh karena tiap poin data dilakukan tes minimal 1 iterasi. Namun, kekurangan yang dimiliki oleh metode ini adalah waktu yang diperlukan lebih lama dibandingkan *holdout method* karena tes yang dilakukan diulang sebanyak *k* iterasi (Refaeilzadeh, Tang, & Liu 2008). Contoh ilustrasi metode *k-fold cross validation* dapat dilihat pada Gambar 5 sebagai berikut:



Gambar.5 Contoh Ilustrasi Metode *K-Fold Cross Validation* (Refaeilzadeh, Tang, & Liu 2008).

2. *Leave-One-Out-Cross-Validation* (LOOCV)

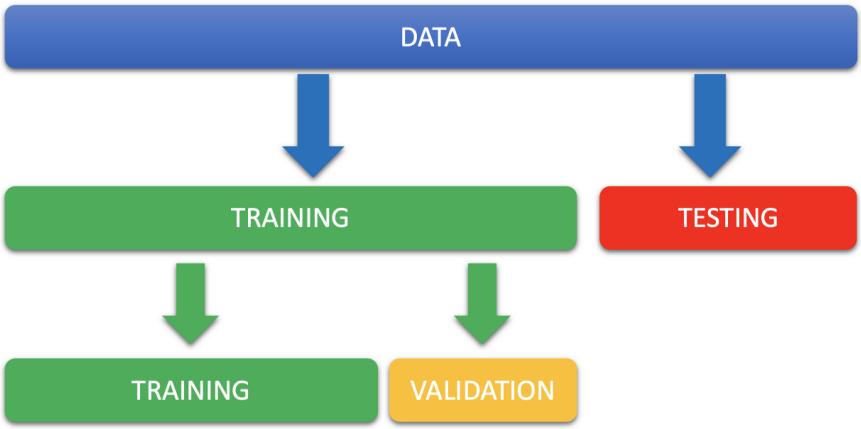
   Leave-One-Out-Cross-Validation (LOOCV) digunakan untuk mengukur akurasi setiap sistem, dimana satu subjek digunakan untuk pengujian dan sisanya digunakan untuk melatih pengklasifikasian. Setiap sistem terdiri dari pengaturan-pengaturan yang berbeda seperti ekstraksi fitur, pemilihan fitur, teknik klasifikasi, dan pelatihan berdasarkan subset data yang berbeda juga (Kohavi, 1995).

   LOOCV melibatkan pemotongan satu observasi dari data yang ditetapkan sebagai titik validasi dan selanjutnya menggunakan pengamatan yang tersisa agar sesuai dengan model dan memprediksi nilai untuk titik validasi. LOOCV diulangi sampai setiap pengamatan dalam set data telah divalidasi (Kohavi, 1995).

3. *Hold-Out Validation*

   Hold-Out Validation menghindari tumpang tindih antara data pelatihan dan data uji, menghasilkan estimasi yang lebih akurat untuk generalisasi kinerja algoritma. Kelemahannya adalah prosedur ini tidak menggunakan semua data yang tersedia dan hasilnya sangat tergantung pada pilihan untuk pelatihan. Selain itu, data dalam set tes mungkin berharga untuk pelatihan dan jika itu dilakukan, kinerja prediksi akan menurun, yang lagi mengarah pada hasil yang miring. Masalah-masalah ini dapat diatasi sebagian dengan mengulangi *hold-out* validasi berulangkali, tetapi kecuali jika pengulangan ini dilakukan secara sistematis, beberapa data dapat dimasukkan dalam data uji dan sementara yang lain tidak termasuk sama sekali, atau sebaliknya beberapa data mungkin selalu masuk kedalam data uji dan tidak pernah

mendapatkan kesempatan untuk berkontribusi pada fase pembelajaran (Refaeilzadeh, Tang, & Liu 2008). Berikut adalah contoh ilustrasi cara kerja *Hold-Out Validation* yang dapat dilihat pada Gambar 6:



Gambar.6 Contoh Ilustrasi *Hold-Out Validation* (Refaeilzadeh, Tang, & Liu 2008).

H. Evaluasi Matrik

Confusion matrix adalah alat ukur berbentuk matriks yang digunakan untuk mendapatkan jumlah ketepatan klasifikasi terhadap kelas dengan algoritma yang dipakai (Sasongko, 2016).

1. *Accuracy*

   Accuracy adalah suatu pengukuran yang dilakukan untuk mengevaluasi suatu model klasifikasi dengan tingkat kedekatan antara nilai prediksi dengan nilai aktual (Bekkar, Djemaa, & Alitouche, 2013). *Accuracy* dapat dihitung dengan persamaan 14.

$$Accuracy = \frac{\sum True\ positive + \sum True\ negative}{\sum Total\ population} \quad \text{.....(14)}$$

2. *Sensitivity*

   Sensitivity adalah suatu pengukuran yang dilakukan untuk mengetahui proporsi positif yang diidentifikasi dengan benar (Bekkar et al., 2013). *Sensitivity* dapat dihitung dengan persamaan 15.

$$Sensitivity = \frac{\sum True\ positive}{\sum True\ positive + \sum False\ negative} \quad (15)$$

3. *Recall / Specificity*

   Recall atau *Specificity* adalah suatu pengukuran yang dilakukan untuk mengetahui proporsi negatif yang diidentifikasi dengan benar (Bekkar et al., 2013). *Recall* dapat dihitung dengan persamaan 16.

$$Recall = \frac{\sum True\ positive}{\sum False\ positive + \sum True\ positive} \quad (16)$$

4. *Matthews Correlation Coefficient*

   Matthews correlation coefficient (MCC) adalah ukuran kualitas klasifikasi biner (dua kelas). Langkah ini diusulkan oleh ahli biokimia Brian W. Matthews pada tahun 1975. MCC memperhitungkan positif dan negatif pada nilai yang bernilai benar dan memperhitungkan positif dan negatif pada nilai yang bernilai salah. Pada umumnya MCC dianggap sebagai ukuran seimbang yang dapat digunakan bahkan jika kelas memiliki ukuran yang sangat berbeda. MCC pada dasarnya adalah koefisien korelasi antara klasifikasi biner yang diamati dan diprediksi. MCC menghasilkan nilai antara +1 dan -1, dimana +1 mewakili prediksi sempurna, 0 tidak lebih baik daripada prediksi acak dan −1 menunjukkan ketidaksepakatan total antara prediksi

dan observasi (Bekkar et al., 2013). MCC dapat dihitung dengan menggunakan persamaan 17.

$$MCC = \frac{(TP \times TN) - (FP \times FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad \text{.....(17)}$$

Pada persamaan diatas, TP adalah jumlah *true positive*, TN adalah jumlah *true negative*, FP adalah jumlah *false positive*, dan FN adalah jumlah *false negative*.

Tiap baris dalam matrik kinerja menggambarkan kategori prediksi dan tiap kolom menggambarkan kategori nyata (Sasongko, 2016). Matrik kinerja (*Confusion Matrix*) dapat dilihat pada Tabel 4. Pada matrik kinerja 2x2, dimana terdapat 2 klasifikasi yaitu positif dan negatif, terdapat beberapa istilah dalam penamaannya, yaitu:

Tabel.4 *Confusion Matrix* (Hossin & Sulaiman, 2015).

		True Condition	
	Total Population	*Condition Positive*	*Condition Negative*
Predicted Condition	*Predicted Condition Positive*	*True Positive* (TP)	*False Positive* (FP)
	Predicted Condition Negative	*False Negative* (FN)	*True Negative* (TN)

Berikut adalah penjelasan dari Tabel 2 yaitu *Confusion Matrix*:

1. *True condition*

   Menggambarkan keadaan nyata dalam klasifikasi.

5. *Predicted condition*

   Menggambarkan keadaan prediksi dalam klasifikasi.

6. *Total population*

   Total population merupakan jumlah keseluruhan data yang dianalisis.

7. *True positive*

   True positive adalah hasil dimana model klasifikasi memprediksi dengan benar kelas positif.

8. *True negative*

   True negative adalah hasil dimana model klasifikasi memprediksi dengan benar kelas negatif.

9. *False positive*

   False positive adalah hasil dimana model klasifikasi memprediksi dengan salah kelas positif.

10. *False negative.*

    False negative adalah hasil dimana model klasifikasi memprediksi dengan salah kelas negatif.

III. METODOLOGI PENELITIAN

A. Tempat dan Waktu Penelitian

1. Tempat Penelitian

   Penelitian Prediksi Asetilasi Pada *Sequence* Protein Lisin Menggunakan *Support Vector Machine* dilaksanakan di Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung, yang berada di jalan Soemantri Brojonegoro no 1 Gedong Meneng, Bandar Lampung. Penelitian ini dilakukan pada semester ganjil tahun ajaran 2018/2019.

2. Waktu Penelitian

   Waktu penelitian dilakukan dimulai pada bulan Maret hingga berakhirnya penelitian ini yaitu pada bulan Agustus 2019. Penelitian ini dimulai dari minggu pertama pada bulan Maret 2019, yang pertama dimulai dari pengumpulan jurnal yang berkaitan dengan judul penulis untuk digunakan sebagai referensi penulis dalam penyusunan laporan, pengumpulan data yang nantinya data tersebut digunakan sebagai bahan untuk dilakukan nya penelitian ini serta bimbingan untuk penyusunan laporan hingga pada tahap akhir yaitu pendaftaran wisuda pada bulan September 2019.

B. Data dan Alat

Dalam penelitian ini, dibutuhkan data dan alat sebagai berikut:

1. Data

   Adapun data yang diperlukan yaitu *sequence* protein lisin yang akan digunakan untuk melakukan Prediksi Asetilasi Pada *Sequence* Protein Lisin Menggunakan *Support Vector Machine.* Data yang digunakan dalam penelitian ini yaitu *sequence protein* lisin yang didapatkan dari jurnal Huang, dkk pada tahun 2016. Data terdiri dari data pada kelas negatif dan data pada kelas positif. Jumlah data yang terdapat dalam data kelas negatif yaitu berjumlah 8.704 dan data kelas positif berjumlah 14.407, sehingga jumlah semua dataset yang digunakan yaitu 23.111 data. Data yang digunakan dalam penelitian ini yaitu *sequence protein* lisin yang didapatkan dari sumber, yaitu https://www.ncbi.nlm.nih.gov/pubmed/26578568 .

2. Alat

   Adapun alat yang digunakan dalam penelitian ini yaitu sebagai berikut:

   a. Perangkat Keras (*Hardware)*

      Perangkat keras yang digunakan pada penelitian ini untuk mendukung dan menunjang pelaksanaan penelitian, yaitu:

      1. Prosesor Intel® Core™ i3-4030U CPU @ 1.90GHz
      2. Celeron ® CPU 1037U @ 1,80 GHz VGA Intel HD4000 RAM 6Gb
      3. DDR3 SDRAM
      4. Threads, Cache 3 MB
      5. *Harddisk* (HDD) 500GB

6. Intel® HD *Graphics* 520

b. Perangkat Lunak (*Software)*

Perangkat lunak yang digunakan pada penelitian ini untuk mendukung dan menunjang pelaksanaan penelitian, yaitu:

1. Sistem Operasi *Windows* 7
2. *R Programming x64 3.6.1*

   Bahasa pemrograman R adalah bahasa pemrograman dan perangkat lunak untuk analisis statistika dan grafik. R dibuat oleh Ross Ihaka dan Robert Gentleman di Universitas Auckland, Selandia Baru, dan kini dikembangkan oleh R Development Core Team, dimana Chambers merupakan anggotanya.

3. *R Studio 1.2.1335*

   Rstudio merupakan sebuah perangkat lunak dengan lisensi gratis dan *open source* yang dikembangkan oleh JJ Allaire. Rstudio digunakan sebagai *integrated development environment* (IDE) untuk bahasa pemrograman R sehingga dapat mempermudah pengguna bahasa pemrograman R untuk melakukan pekerjaan yang berkaitan dengan statistik dan grafik.

4. *Library e1071 1.7-2*

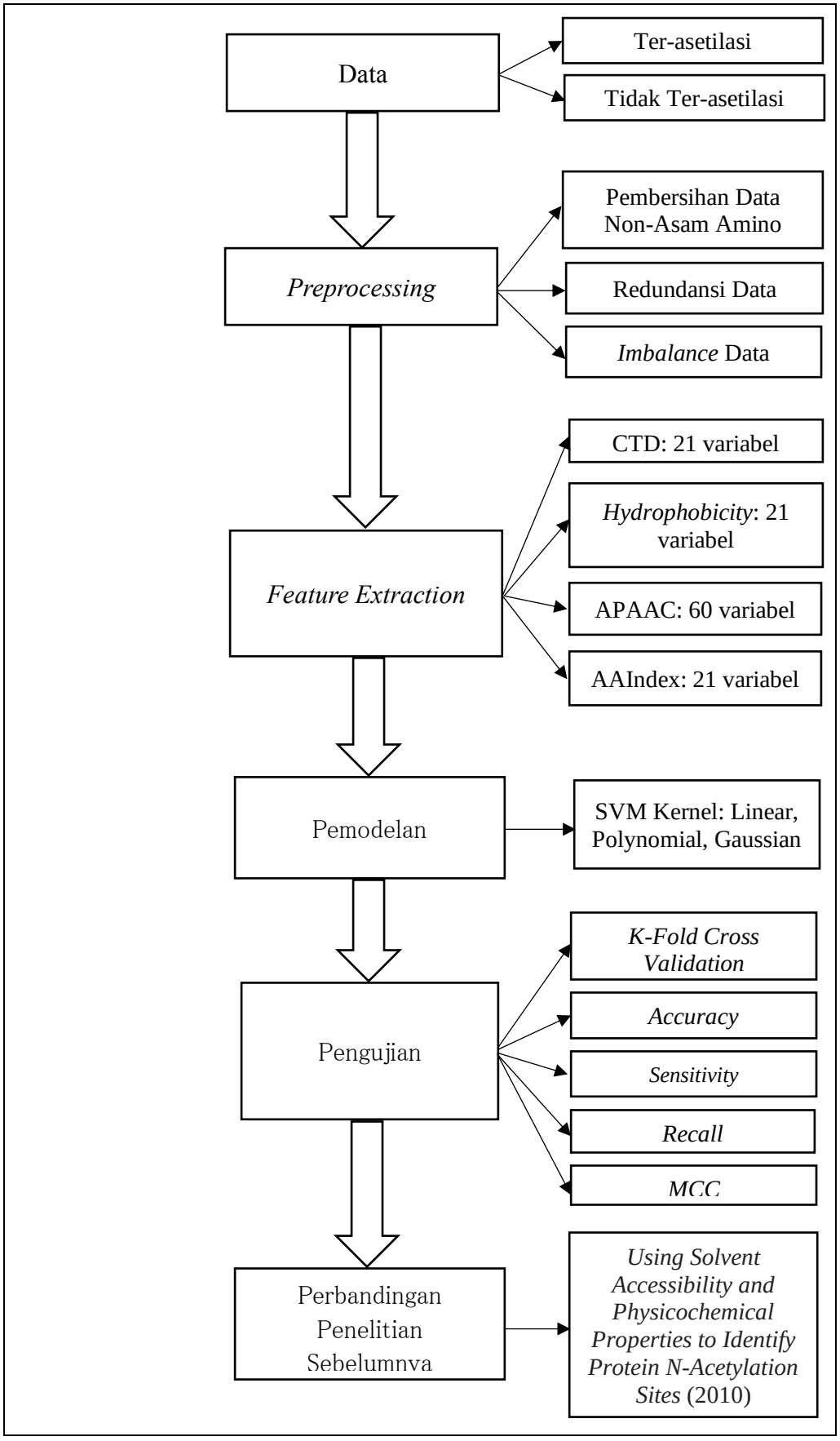
   Library e1071 merupakan fungsi untuk *Class Analysis, Fuzzy Clustering, Support Vector Machines, Naive Bayes Classifier, Random Forest.*

5. *Library Caret 6.0-84*

   Library caret (***C****lassification* ***A****nd* ***RE****gression* ***T****raining*) merupakan serangkaian fungsi yang berupaya merampingkan proses untuk membuat model prediksi.

C. Alur Kerja Penelitian

Alur kerja penelitian merupakan langkah-langkah yang dilakukan peneliti dalam melakukan penelitian. Adapun alur kerja penelitian yang dilakukan pada penelitian ini, dapat dilihat pada Gambar 7 sebagai berikut:



Gambar.7 Alur Kerja Penelitian.

Penjelasan dari alur implementasi pada Gambar 5 adalah sebagai berikut:

1. Data

   Data yang diperlukan yaitu *sequence* protein lisin yang akan digunakan untuk melakukan Prediksi Asetilasi Pada *Sequence* Protein Lisin Menggunakan *Support Vector Machine.* Data yang digunakan dalam penelitian ini yaitu *sequence protein* lisin yang didapatkan dari *study literature* penelitian Huang, dkk pada tahun 2016. Data terdiri dari data pada kelas negatif dan data pada kelas positif. Jumlah data yang terdapat dalam data kelas negatif yaitu berjumlah 8.704 dan data kelas positif berjumlah 14.407, sehingga jumlah semua dataset yang digunakan yaitu 23.111 data. Panjang *sequence* protein pada data negatif dan positif yaitu 21.

2. *Preprocessing*

   Adapun langkah pada tahap *preprocessing* yaitu:

   a. Pembersihan Data Non-Asam Amino

      Pembersihan *sequence* protein yang tidak termasuk dalam asam amino dilakukan pada masing-masing kelas data. *Sequence* protein yang terdapat huruf X, tidak dapat dibaca dan tidak dapat diolah pada saat *feature extraction* karena bukan bagian dari data *sequence* protein. Pembersihan data *sequence* protein dilakukan pada data negatif dan data positif. Berikut adalah contoh *sequence* protein yang terdapat huruf X, "RALPRQDTVIKHYQRPAXXXX" dan *sequence* protein seperti contoh tersebut harus dihapus atau dibersihkan.

b. *Skipredundant Sequence* Protein

Pada langkah ini, dilakukan *skipredundant sequence* protein agar data yang akan diklasifikasikan nantinya tidak terjadi banyak kemiripan. *Skipredundant* dilakukan sebesar 10% pada masing-masing data negatif dan data positif.

c. *Imbalance* Data

Terdapat ketidakseimbangan data pada data negatif dan positif yaitu 1479 data negatif dan 3588 data positif. Maka, diperlukan data yang seimbang antara data negatif dan positif agar hasil klasifikasi yang akan dilakukan memperoleh hasil yang seimbang dan maksimal. Oleh karena itu, dalam hal ini perlu dilakukan penghapusan sebagian data pada data positif agar banyaknya data negatif dan data positif seimbang.

3. *Feature Extraction*

Tahap *feature extraction* yaitu, untuk mengekstraksi data agar menjadi ciri masing-masing kelas. Ekstraksi fitur yang digunakan pada penelitian ini menggunakan *package* protr yang terdapat di dalam R Programming. Proses ini akan menjadi lebih mudah untuk mengklasifikasikan antara kelas yang berbeda dengan melihat fitur-fitur sehingga mudah dibedakan.

Pada langkah ini yaitu, menggunakan 4 *feature extraction* berbeda yang terdapat dalam *package* protr dan *package BioSeqClass* yaitu :

a. *Composition, Transition, Distribution (CTD)* pada *package BioSeqClass:*

Jumlah dimensi yang terdapat pada CTD yaitu 21 variabel.

b. *Hydrophobicity* pada *package BioSeqClass:*

Jumlah dimensi yang terdapat pada *Hydrophobicity* yaitu 21 variabel.

c. *AAindex* pada *package BioSeqClass*:

Jumlah dimensi yang terdapat pada *AAindex* yaitu 21 variabel.

d. *Amphiphilic Pseudo Amino Acid Composition* (APAAC) pada *package* protr:

Jumlah dimensi yang terdapat pada *Amphiphilic Pseudo Amino Acid Composition* yaitu 60 variabel.

4. Membuat Model Klasifikasi SVM dengan 10-*Fold Cross Validation.*

Pada langkah ini yaitu membuat model klasifikasi menggunakan metode SVM (*Support Vector Machine*). Langkah yang dilakukan yaitu data akan dibagi menjadi data *training* dan data *testing*. Pembagian data yaitu, 10% data sebagai data *test*, dan 90% data sebagai data *training* sebanyak 10 kali karena penelitian ini menggunakan aturan *10-Fold Cross Validation.* Kernel SVM yang dipakai yang dipakai dalam klasifikasi penelitian ini yaitu Kernel Linear, Kernel Polynomial, dan Kernel Gaussian.

5. Pengujian

Setelah data sudah diklasifikasikan menggunakan *support vector machine* dan sudah dilakukan *10-Fold Cross Validation*, maka hasil dari penelitian tersebut akan dilakukan pengujian untuk menguji hasil yang sudah dilakukan. Pengujian yang digunakan yaitu dengan menggunakan *confusion matrix* yang terdiri dari *Accuracy, Sensitivity, Recall,* dan *MCC.* Selanjutnya

yaitu, menghitung rata-rata *Accuracy* yang sudah dilakukan sebanyak 10-*Fold Cross Validation* untuk mendapatkan hasil kinerja prediksi asetilasi pada *sequence* protein lisin menggunakan *Support Vector Machine.*

V. SIMPULAN DAN SARAN

A. Simpulan

Dari hasil penelitian yang dilakukan, dapat diambil simpulan sebagai berikut:

1. *Feature extraction* yang digunakan menggunakan dua *package* yaitu *package* BioSeqClass dan *package* protr karena menggunkan dua *package* tersebut menghasilkan akurasi yang cukup baik untuk penelitian ini. Pada *package* BioSeqClass yaitu menggunakan CTD (*Composition, Transition, Distribution)* dan *Hydrophobicity* dan *AAindex* sedangkan pada *package* protr yaitu APAAC (*Amphiphilic Pseudo Amino Acid Composition)*. Masing-masing *feature extraction* memiliki variabel yang berbeda yaitu CTD 21 variabel, *Hydrophobicity* 21 variabel, APAAC 60 variabel, dan *AAindex* 21 variabel, sehingga jumlah variabel dari keempat *feature extraction* yang digunakan pada penelitian ini yaitu 123 variabel.
2. Pada penelitian ini mendapatkan hasil kinerja pada kernel Gaussian dengan akurasi lebih baik dibandingkan dengan penelitian sebelumnya yaitu 82.79% sedangkan penelitian sebelumnya oleh Lee, dkk pada tahun 2010 mendapatkan akurasi lebih rendah yaitu 75.00%.

3. Metode SVM pada kernel Gaussian, memiliki hasil kinerja terbaik dengan akurasi tertinggi dibandingkan dengan metode SVM pada kernel Polynomial dan Linear. Pada penelitian ini metode SVM (*Support Vector Machine)* kernel Gaussian memiliki nilai akurasi sebesar 97.52%.
4. Percobaan menggunakan empat *feature extraction* yaitu CTD (*Composition, Transition, Distribution)* dan *Hydrophobicity,* APAAC (*Amphiphilic Pseudo Amino Acid Composition)*, dan *AAindex* menghasilkan akurasi yang baik dibandingkan menggunakan *feature extraction* lainnya.

B. Saran

Adapun saran yang diberikan untuk penelitian selanjutnya tentang Prediksi Asetilasi Pada *Sequence* Protein Lisin Menggunakan *Support Vector Machine* ini yaitu sebagai berikut:

1. Pada penelitian selanjutnya diharapkan agar menggunakan metode klasifikasi lain seperti *Random Forest* sehingga tidak hanya menggunakan metode klasifikasi SVM (*Support Vector Machine*) saja dan dapat mengetahui perbandingan hasil akurasi dengan metode yang berbeda.
2. Dapat menggunakan *feature extraction* selain empat *feature extraction* yang digunakan pada penelitian ini, dan menggunakan lebih banyak lagi *feature extraction* agar dapat mendapatkan nilai rata-rata akurasi yang lebih baik.
3. Mencoba menggunakan data *sequence* protein lain untuk mengetahui hasil prediksi menggunakan *Support Vector Machine*.

DAFTAR PUSTAKA

A.D.Dongare, R.R.Kharde, Amit D.Kachare. (2012). *Introduction to Artificial Neural Network (ANN) Methods*. *2*(1), 327–352.

‘Arif, M., Hassan, H., Nasien, D., & Haron, H. (2015). A Review on Feature Extraction and Feature Selection for Handwritten Character Recognition. *International Journal of Advanced Computer Science and Applications*, *6*(2), 204–212. https://doi.org/10.14569/ijacsa.2015.060230

Akram, M., Asif, H. M., Uzair, M., Akhtar, N., Madni, A., Ali Shah, S. M., … Ullah, A. (2011). Amino acids: A review article. *Journal of Medicinal Plants Research*, *5*(17), 3997–4000.

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (2012). The ‘K’ in K-fold cross validation. *ESANN 2012 Proceedings, 20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, (April), 441–446.

Ansari, S., & Sutar, U. (2015). Devanagari Handwritten Character Recognition using Hybrid Features Extraction and Feed Forward Neural Network Classifier (FFNN). *International Journal of Computer Applications*, *129*(7), 22–27. https://doi.org/10.5120/ijca2015906859

Audagnotto, M., & Dal Peraro, M. (2017). Protein post-translational modifications: In silico prediction tools and molecular modeling. *Computational and Structural Biotechnology Journal*, *15*(April), 307–319. https://doi.org/10.1016/j.csbj.2017.03.004

Bekkar, M., Djemaa, H. K., & Alitouche, T. A. (2013). Evaluation Measures for

Models Assessment over Imbalanced Data Sets. *Journal of Information Engineering and Applications*, *3*(10), 27–38. Retrieved from http://www.iiste.org/Journals/index.php/JIEA/article/view/7633

Bhavsar, H., & Panchal, M. H. (2012). A Review on Support Vector Machine for Data Classification. *International Journal of Advanced Research in Computer Engineering & Technology*, *1*(10), 2278–1323.

Bui, D. D. A., Del Fiol, G., & Jonnalagadda, S. (2016). PDF text classification to leverage information extraction from publication reports. *Journal of Biomedical Informatics*, *61*, 141–148. https://doi.org/10.1016/j.jbi.2016.03.026

Cortes, C., & Vapnik, V. (1995). *Support-Vector Networks*. *297*, 273–297.

Hossin, M., & Sulaiman, N. (2015). A Review on Evaluation Metrics For Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), 1–11.

Jamilah, B., Mohamed, A., Abbas, K. A., Abdul Rahman, R., Karim, R., & Hashim, D. M. (2009). Protein-starch interaction and their effect on thermal and rheological characteristics of a food system: A review. *Journal of Food, Agriculture and Environment*, *7*(2), 169–174.

Khalid, S., Khalil, T., & Nasreen, S. (2014). A survey of feature selection and feature extraction techniques in machine learning. *Proceedings of 2014 Science and Information Conference, SAI 2014*, (October 2016), 372–378. https://doi.org/10.1109/SAI.2014.6918213

Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *International Joint Conference of Artificial Intelligence*, *5*.

Ladwani, V. M. (2018). Support vector machines and applications. *Computer Vision: Concepts, Methodologies, Tools, and Applications*, 1381–1390. https://doi.org/10.4018/978-1-5225-5204-8.ch057

Li, S., Li, H., Li, M., Shyr, Y., Xie, L., & Li, Y. (2009). *Improved Prediction of*

Lysine Acetylation by Support Vector Machines. 977–983.

Li, Y., Wang, M., Wang, H., Tan, H., Zhang, Z., Webb, G. I., & Song, J. (2014). Accurate in silico identification of species-specific acetylation sites by integrating protein sequence-derived and functional features. *Scientific Reports*, *4*. https://doi.org/10.1038/srep05765

Lu, C. T., Lee, T. Y., Chen, Y. J., & Chen, Y. J. (2014). An Intelligent System for Identifying Acetylated Lysine on Histones and Nonhistone Proteins. *BioMed Research International*, *2014*. https://doi.org/10.1155/2014/528650

Medhi, S., Ahmed, C., & Gayan, R. (2016). A Study on Feature Extraction Techniques in Image Processing. *Internasional Journal of Computer Sciences and Engineering*, *4*(7), 89–93. https://doi.org/10.1016/B978-1-84334-596-1.50008-0

Minguez, P., Letunic, I., Parca, L., & Bork, P. (2013). PTMcode: A database of known and predicted functional associations between post-translational modifications in proteins. *Nucleic Acids Research*, *41*(D1), 306–311. https://doi.org/10.1093/nar/gks1230

Nugroho, A. S., Witarto, A. B., & Handoko, D. (2004). Support Vector Machine-Support Vector Machine. *Kuliah Umum IlmuKomputer.Com*. Retrieved from http://www.dbpia.co.kr/Journal/ArticleDetail/401581

Qiu, W. R., Sun, B. Q., Xiao, X., Xu, Z. C., & Chou, K. C. (2016). iPTM-mLys: Identifying multiple lysine PTM sites and their different types. *Bioinformatics*, *32*(20), 3116–3123. https://doi.org/10.1093/bioinformatics/btw380

Sasongko, T. B. (2016). Komparasi dan Analisis Kinerja Model Algoritma SVM dan PSO-SVM. *Politeknik Harapan Bersama Tegal*, *2*(2), 244–253.

Sikic, K., & Carugo, O. (2010). *Protein sequence redundancy reduction : comparison of various methods Bioinformation*. *5*(6).

Wijaya, A. S., Chamidah, N., & Santoni, M. M. (2019). Pengenalan Karakter Tulisan Tangan Dengan K-Support Vector Nearest Neighbor. *IJEIS*

(Indonesian Journal of Electronics and Instrumentation Systems), *9*(1), 33. https://doi.org/10.22146/ijeis.38729

Wuyun, Q., Zheng, W., Zhang, Y., Ruan, J., & Hu, G. (2016). Improved species-specific lysine acetylation site prediction based on a large variety of features set. *PLoS ONE*, *11*(5), 1–21. https://doi.org/10.1371/journal.pone.0155370

Xiao, N., Cao, D. S., Zhu, M. F., & Xu, Q. S. (2015). Protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences. *Bioinformatics*, *31*(11), 1857–1859. https://doi.org/10.1093/bioinformatics/btv042

Xiao, N., Cao, D., & Xu, Q. (2014). *Rcpi : R / Bioconductor Package as an Integrated Informatics Platform in Drug Discovery*.

Zupan, J. (1994). *Introduction to Artificial Neural Network (ANN) Methods*. *2*(1), 327–352.