BAB II
TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

2.1.1. Penelitian Terkait

Dalam penyusunan Laporan ini, penulis sedikit banyak terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang berkaitan dengan latar belakang masalah pada Laporan ini. Berikut ini penelitian terdahulu yang berhubungan dengan laporan ini antara lain.

Penelitian yang dilakukan Taghfirul Azhima Yoga Siswa, Prihandoko 2018, Hasil kinerja terbaik yang diuji menggunakan T-Test pada algoritma C4.5 dan Naïve Bayes berbasis Particle Swarm Optimization (PSO) dapat dihasilkan bahwa algoritma Naïve Bayes (PSO) memiliki nilai tertinggi sebesar 0,980 dilanjutkan algoritma C4.5 (PSO) sebesar 0,943. Dengan demikian algoritma Naïve Bayes Particle Swarm Optimization (PSO) dapat memberikan solusi terbaik terhadap akurasi pendeteksian penyakit kanker payudara.[3]

Penelitian yang dilakukan Wiwit Supriyanti, Kusrini, Armadyah Amborowati 2016, Hasil uji kinerja algoritma klasifikasi untuk kasus ketepatan pemilihan konsentrasi mahasiswa untuk algoritma C4.5 tanpa penambahan seleksi fitur forward selection diperoleh nilai akurasi sebesar 84,43%, kemudian setelah ditambahkan seleksi fitur forward selection meningkat menjadi 84,98%. Sedangkan pada algoritma Naive Bayes tanpa penambahan seleksi fitur forward selection diperoleh nilai akurasi sebesar 78,47%, setelah ditambahkan seleksi fitur forward selection meningkat menjadi 82,01%.[8]

Penelitian yang dilakukan Irene Lishania, Rito Goejantoro, dan Yuki Novia Nasution 2019, Berdasarkan hasil analisis dan pembahasan, didapatkan bahwa hasil ketepatan klasifikasi penyakit stroke pada data pasien di RSUD Abdul Wahab Sjahranie bulan November dan Desember 2017 dengan metode naive

Bayes adalah 81,25% dan metode decision tree algoritma (J48) diperoleh tingkat akurasi sebesar 87,5%. Hal ini menunjukkan bahwa pada penelitian ini, metode decision tree algoritma (J48) memberikan ketepatan prediksi klasifikasi yang lebih baik.[9]

Penelitian yang dilakukan Kelvin Leonardi Kohsasih, Zakarias Situmorang 2022, Berdasarkan hasil penelitian yang telah dilakukan yaitu dengan membagi dataset menjadi 60% data training dan 40% data testing maka dapat simpulkan bahwa algortitma C4.5 memiliki performa yang lebih baik yaitu dengan tingkat akurasi sebesar 95% serta nilai presisi, recall dan f1-score masing masing yaitu 90%, 95% dan 93%. sedangkan algoritma naïve bayes mendapatkan tingkat akurasi sebesar 91%, presisi 92%, recall 91% dan f1-score sebesar 92%. selain itu hasil log loss dan specificity dari algorima naïve bayes yaitu 0.205 dan 0.213 sedangkan algoritma C4.5 mendapatkan nilai masing masing yaitu 0.190 dan 0.047.[10]

Lalu penelitian yang dilakukan Annisa Puspitawuri, Edy Santoso, Candra Dewi 2019, Metode kombinasi K-Nearest Neighbor dan Naïve Bayes dapat digunakan untuk melakukan diagnosis tingkat risiko penyakit stroke dengan 8 atribut dan 3 jenis tingkatan risiko penyakit stroke, yaitu tingkat risiko rendah, sedang dan tinggi. Langkah pertama yang dilakukan adalah menggunakan atribut dengan tipe numerik sebagai data uji dan data latih untuk proses klasifikasi menggunakan algoritme K-Nearest Neighbor. Kemudian menentukan K Neighbor terdekat untuk proses algoritme K-Nearest Neighbor menggunakan atribut yang bersifat numerik. K tetangga terdekat yang telah didapatkan kemudian digunakan untuk langkah berikutnya, yaitu sebagai data latih dari model algoritme Naïve Bayes menggunakan atribut yang bersifat kategoris. Kemudian akan didapatkan hasil klasifikasi dari model-model yang telah dibangun. Hasil klasifikasi berupa nilai tertinggi pada proses posterior Naïve Bayes, kemudian menjadi label kelas untuk data yang di uji. Jika terjadi kesamaan nilai tertinggi pada posterior, maka nilai mayor pada proses K-Nearest Neighbor akan menjadi label kelas data uji.[11]

2.1.2. Perbandingan dengan penelitian terdahulu.

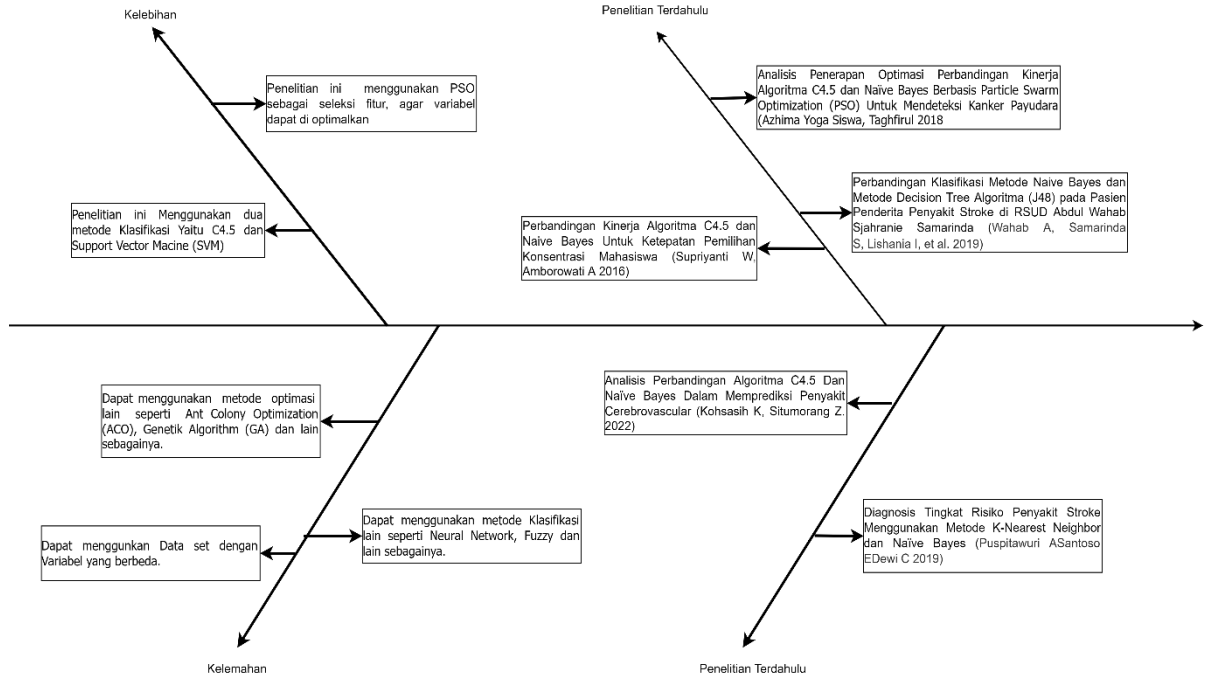
Perbandingan dengan penelitian terdahulu ditampilkan pada table dibawah ini:

Tabel 2.1 Tabel Perbandingan Dengan Penelitian Terdahulu

No	Peneliti	Judul	Perbandingan
1.	Tagfirul Azhima Yoga Siswa, Prihandoko (2018)	Analisis Penerapan Optimasi Perbandingan Kinerja Algoritma C4.5 dan Naïve Bayes Berbasis Particle Swarm Optimization (PSO) Untuk Mendeteksi Kanker Payudara.	Penelitian ini memiliki kesamaan pada *machine learning* namun berbeda pada algoritma yang digunakan. Penelitian ini menggunkan PSO dan komparasi algortima.
2.	Ramdhan Saepul Rohman, Rizal Amegia Saputra, Dasya Arif Firmansaha (2020)	Komparasi Algoritma C4.5 Berbasis PSO dan GA dalam Diagnosa Penyakit Stroke	Penelitian ini menggunakan dataset Public dari kaggel tentang stroke. Algortima yang digunakan hanya C4.5 dengan menggabungkan dua seleksi fitur.
3.	Irene Lishania, Rito Goejantoro, dan Yuki Novia Nasution (2019)	Perbandingan Klasifikasi Metode Naive Bayes dan Metode Decision Tree Algoritma (J48) pada Pasien Penderita Penyakit Stroke di RSUD Abdul Wahab Sjahranie	Menggunakan 2 metode klasifikasi dengan dataset bertipe private dari Rumah Sakit. Tidak menggunkan seleksi fitur apapun algortima Decission yang

		Samarinda	digunakan merupakan penerapan pada WEKA.
4.	Annisa Puspitawuri, Edy Santoso, Candra Dewi (2019)	Diagnosis Tingkat Risiko Penyakit Stroke Menggunakan Metode K-Nearest Neighbor dan Naïve Bayes.	Penelitian menggunakan dataset yang sudah pernah di uji dengan system wawancara dengan jumlah 150 data. Peneliti tidak menggunakan seleksi fitur dan perbedaan algoritma klasifikasi.
5.	Kelvin Leonardi Kohsasih, Zakarias Situmorang. (2022)	Analisis Perbandingan Algoritma C4.5 Dan Naïve Bayes Dalam Memprediksi Penyakit Cerebrovascular	Dataset yang digunkan sama bersumber dari kaggel. Peneliti juga malakukan komparasi antar algorima klasifikasi, namun berbeda pada algoritma pembanding dan tak menggunakan seleksi fitur.

Adapun peneliti menggunakan diagram *fishbone* untuk mengetahui kelebihan dan kekurangan dengan penelitian terdahulu yang pernah di lakukan:



Gambar 2.1 Diagram *Fishbone* Penelitian Terdahulu

2.2. Landasan Teori

2.2.1. Data Mining

Data mining adalah suatu istilah yang digunakan untuk menemukan pengetahuan yang tersembunyi di dalam database.

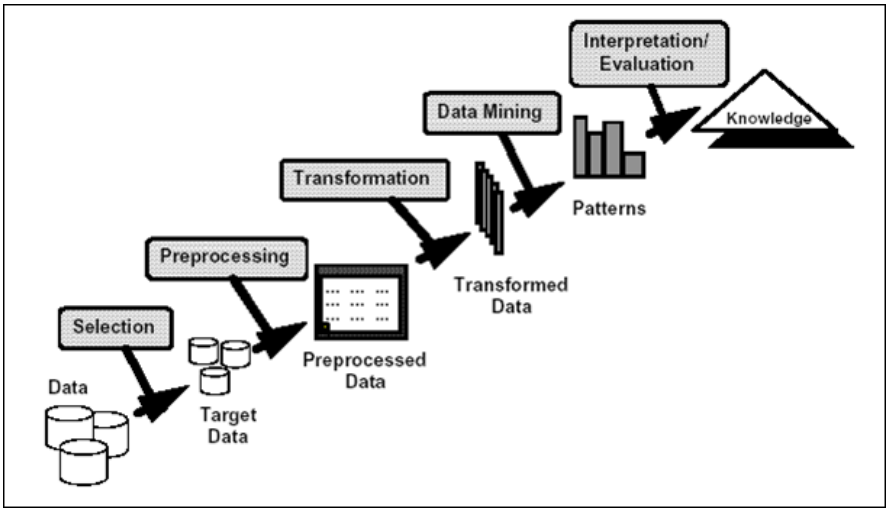
Menurut (Turban et al, 2005) data mining merupakan proses semi otomatik yang menggunakan teknik statistik, matematika, kecerdasan buatan, dan machine learning untuk mengekstraksi dan mengidentifikasi informasi pengetahuan potensial dan berguna yang bermanfaat yang tersimpan di dalam database besar.

Menurut (Larose, 2006) data mining adalah suatu proses menemukan hubungan yang berarti, pola, dan kecenderungan dengan memeriksa dalam sekumpulan besar data yang tersimpan dalam penyimpanan dengan menggunakan teknik pengenalan pola seperti teknik statistik dan matematika.

Menurut (Witten, Ian H. Frank, 2011) Definisi data mining adalah proses ekstraksi suatu data/pola (sebelumnya tidak diketahui, bersifat implisit, dianggap

tidak berguna) menjadi informasi atau pengetahuan dari data yang jumlahnya besar

Penelitian (Kusrini & Taufiq, 2009). *Data Mining* dan *Knowledge Discovery in Database (KDD)* banyak digunakan untuk menginformasikan proses pengambilan informasi yang tersembunyi di *database* besar. Namun pada dasarnya kedua istilah ini memiliki konsep yang berbeda, namun saling terkait, dan salah satu langkah dari proses *KDD* adalah *data mining*. Berikut ini adalah garis besar proses *KDD.*



Gambar 2.2 Proses KDD

1. *Data Selection*

pemilihan data dari kumpulan data operasional yang harus dilakukan sebelum mulai mengekstraksi data dari *KDD.* Data hasil pemilihan digunakan dalam proses *data mining* dan disimpan dalam *file* terpisah dari *database* operasional.

2. *Pre-processing/Cleaning*

Langkah pembersihan ini termasuk menghapus data duplikat, memeriksa data yang tidak konsisten, dan memperbaiki kesalahan pada data

3. *Transformation*

merupakan proses kreatif dan sangat bergantung pada database, jenis atau model data yang akan diambil. Data dikonversi atau digabungkan menjadi format khusus sebelum dapat digunakan. Misalnya, beberapa metode standar seperti asosiasi dan analisis klaster hanya dapat menerima data input kategorikal. Oleh karena itu, data berupa data numerik kontinu harus dibagi menjadi beberapa interval waktu, misalnya atribut data umur dapat diubah menjadi kelompok umur.

4. *Data Mining*

Proses menemukan pola informasidalam data terpilih dengan menggunakan teknik metode tertentu. Memilih metode algoritme yang sesuai tergantung pada tujuan dan proses *KDD.*

5. *Interpretation*/Evaluasi

Sampel informasi yang dihasilkan dari proses data *mining* harus disajikan dalam format yang dapat dipahami oleh pemangku kepentingan. Tahap ini mengkaji apakah pola dan informasi yang ditemukan bertentangan dengan fakta dan hipotesis yang ada.

2.2.2. Algoritma C4.5.

Algoritma C4.5 merupakan salah satu algoritma pohon keputusan yang dapat digunakan untuk menghasilkan aturan yang mudah diinterpretasikan dan tercepat diantara algoritma lainnya. Algoritma tersebut juga mampu menghasilkan subsistem model dasar yang dapat digunakan untuk mendukung sistem pendukung keputusan. Sehingga penelitian tentang peningkatan performansi algoritma C4.5 masih sangat menarik untuk dilakukan. Ada banyak fitur yang terlibat dalam Algoritma C4.5 yaitu data, data atribut, instance, dan kelas atribut. Dalam beberapa kasus klasifikasi, algoritma ini masih menghasilkan akurasi yang kurang maksimal. Oleh karena itu, penelitian ini bertujuan untuk meningkatkan kinerja Algoritma C4.5 dengan menerapkan proses modifikasi persamaan dan penambahan perlakuan untuk meningkatkan akurasi pemilihan node yang akan

dipangkas. Beberapa metode pengembangan algoritma C4.5 fokus pada fase pruning yang masih memungkinkan untuk trimming node dengan informasi bernilai tinggi atau kontributif. Cara mengatasinya adalah dengan memodifikasi fungsi pruning dan akan memastikan bahwa proses pruning dilakukan terhadap cabang yang benar-benar non-kontributif, sehingga meningkatkan akurasi hasil.[12]

Menurut Sukma, dkk (2019, p.23), algoritma C4.5 merepresentasikan algoritma konstruksi pohon keputusan yang dikembangkan oleh *Ross Qiunlan,* yaitu membangun pohon keputusan: atribut mana yang memiliki prioritas tertinggi, atau mana yang mungkin memiliki nilai boost tertinggi, berdasarkan nilai entropy atribut sebagai sumbu dari klasifikasi atribut. Pada fase tersebut, algoritma C4.5 memiliki dua prinsip operasi. membangun pohon keputusan dan membangun aturan (*rule model*). Aturan yang dibentuk dari pohon keputusan membentuk kondisi dengan cara "jika-maka".

Proses membangun pohon keputusan dengan algoritma C4.5 memiliki empat langkah:

a. Pemilihan atribut sebagai root didasarkan pada nilai gain tertinggi dari atribut yang ada.

b. Mengambil cabang untuk setiap nilai berarti mengambil cabang sesuai dengan nilai variabel gain maksimum.

c. Bagilah pada setiap kasus pada cabang berdasarkan perhitungan nilai gain tertinggi, lakukan perhitungan setelah menghitung nilai gain tertinggi pertama, dan jalankan kembali proses perhitungan gain tertinggi tanpa menggunakan nilai gain pertama. variabel.

d. Mengulangi proses tersebut pada setiap cabang sedemikian rupa sehingga semua *case* di dalam cabang tersebut memiliki kelas yang sama, dan mengulangi proses perhitungan keuntungan maksimum untuk setiap cabang *case* sampai proses perhitungan gagal/sampai terisi penuh.

Menurut Haryati, dkk (2015, p. 132), pohon keputusan menyerupai struktur pohon yang di dalamnya terdapat simpul-simpul internal (bukan daun) yang menggambarkan atribut, dan setiap cabang mewakili salah satu atribut yang diuji, setiap daun menggambarkan sebuah kelas. Sebuah pohon keputusan dimulai dari akar paling atas. Satu set data uji yang diberikan. Jika kelas data Y tidak diketahui, pohon keputusan ditelusuri dari akar ke node dan setiap nilai atribut oleh data Y diuji untuk melihat apakah mengikuti aturan pohon keputusan. Sebuah pohon keputusan memprediksi kelas dari sebuah tuple Y. Algoritma C4.5 dan pohon keputusan adalah dua model atom. Karena algoritma C4.5 membangun pohon keputusan. (dikotomi berulang) Namun, proyek ini sebenarnya dikerjakan lebih awal oleh *E.B. Hunt, J Marin,* dan *P.T. Stone. Quinlan* kemudian membuat sebuah algoritma dari pengembangan ID3 yang disebut C4.5, berdasarkan pembelajaran yang diawasi. Ada beberapa tahap dalam membuat sebuah pohon keputusan dengan algoritma C4.5, yaitu:

1) Siapkan data latih. Data latih biasanya terdiri dari data historis yang terjadi sebelumnya dan dikelompokkan ke dalam kelas tertentu.

2) Tentukan akar pohon. *Root* diperoleh dari atribut yang dipilih dengan menghitung nilai gain untuk setiap atribut. Nilai gain tertinggi akan menjadi *root* pertama. Sebelum menghitung nilai gain untuk suatu atribut, hitung terlebih dahulu nilai entropinya. Dengan persamaa berikut:

$$entropy\ (S) = \sum_{i=1}^{n} - \pi * log2\ \pi \qquad \textbf{(2.1)}$$

Keterangan:

S: himpunan kasus.

n: jumlah partisi S.

π: proporsi dari Si terhadap S.

3) Kemudian hitung nilai Gain dengan metode information gain:

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

Keterangan:

S = Himpunan Kasus.

A = Atribut.

n = Jumlah Partisi Atribut.

A | Si | = Jumlah Kasus pada partisi ke-i.

| S | = Jumlah Kasus dalam S.

4). Gain ratio

Untuk menghitung gain ratio kita perlu ketahui suatu term baru yang disebut split information. Split information dihitung dengan formula sebagai berikut:

$$SplitInfo(S,A) = -\sum_{i=n}^{n} . \frac{si}{s} \log 2 \frac{si}{s} \quad \textbf{(2.3)}$$

S1 sampai Sc adalah c subset yang dihasilkan dari pemecahan S dengan menggunakan atribut A yang mempunyai banyak C nilai.

Selanjutnya gain ratio dihitung dengan cara:

$$Gainratio = \frac{Gain(S,A)}{Splitinformation(S,A)} \quad \textbf{(2.4)}$$

5). Ulangi langkah ke (2) *Entropy* hingga semua tupel terpisah-pisah.

6) Pemisahan pohon keputusan berakhir ketika:

a. Semua tupel dari node N mendapatkan kelas yang sama.

b. Sebuah tuple yang dipartisi tidak memiliki atribut.

c. Cabang kosong tidak memiliki tupel.

Secara umum untuk membangun pohon keputusan algoritma C4.5 adalah sebagai berikut:

1. Pilih atribut sebagai akar.
2. Buat cabang untuk tiap-tiap nilai.
3. Bagi kasus dalam cabang.
4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama.

Kelebihan utama Algoritma C4.5 dapat membuat pohon keputusan (decision tree) yang efisien menangani atribut tipe diskrit dan tipe diskrit- numerik, mudah untuk diinterprestasikan dan memiliki tingkat akurasi yang dapat diterima.

Kekurangannya Algoritma C4.5 hanya dapat digunakan untuk menangani sampel-sampel yang dapat disimpan secara keseluruhan dan pada waktu yang bersamaan di memori.

2.2.3. Support Vector Machine

Support Vector Machine (SVM) metode pembelajaran mesin yang telah menjadi sangat populer untuk analisis neuroimaging dalam beberapa tahun terakhir. Karena kesederhanaan dan fleksibilitasnya yang relatif untuk mengatasi berbagai masalah klasifikasi, SVM secara khusus memberikan kinerja prediksi yang seimbang, bahkan dalam studi di mana ukuran sampel mungkin terbatas. Dalam penelitian gangguan otak, SVM biasanya digunakan menggunakan analisis pola multivoxel (MVPA) karena kesederhanaan relatifnya membawa risiko overfitting yang lebih rendah bahkan menggunakan data pencitraan dimensi tinggi. Baru-baru ini, SVM telah digunakan dalam konteks psikiatri presisi, terutama untuk aplikasi yang melibatkan prediksi diagnosis dan prognosis penyakit otak seperti penyakit Alzheimer, skizofrenia, dan depresi. Di bagian terakhir bab ini, kami meninjau sejumlah studi terbaru yang menggunakan SVM untuk aplikasi tersebut.[13]

Support Vector Machines (SVM) dikembangkan oleh Boser, Guyon, dan Vapnik dan pertama kali dipresentasikan pada Lokakarya Tahunan 1992 tentang Teori Pembelajaran Komputasi. Konsep dasar metode SVM sebenarnya merupakan gabungan dari teori-teori komputasi awal seperti: Pengganda *Lagrange* ditemukan oleh Joseph Louis Lagrange pada tahun 1766, bersama dengan konsep pendukung lainnya.

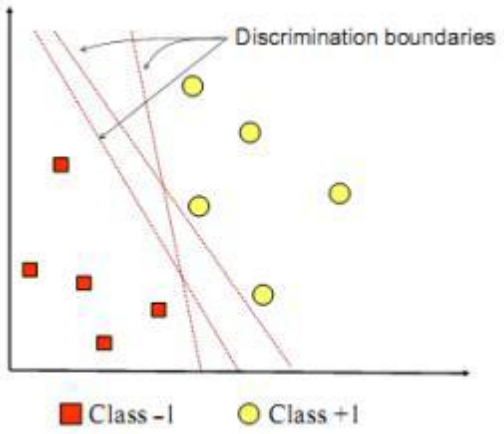
Menurut Fachrurrazi (2011), SVM adalah teknik untuk membuat prediksi, baik untuk regresi maupun klasifikasi. Metode SVM digunakan untuk mendapatkan fungsi pemisahan (hyperplane) yang optimal untuk memisahkan observasi dengan nilai variabel target yang berbeda (William, 2011). Hyperplane ini bisa berupa garis dalam 2D atau bidang dalam berbagai dimensi. Menurut Nugroho (2003), karakteristik umum SVM dapat diringkas sebagai berikut:

1. Mengubah data di ruang input menjadi ruang berdimensi lebih tinggi (ruang fitur) untuk pengenalan pola dan optimalisasi di ruang vektor baru. Ini membedakan SVM dari solusi pengenalan pola umum yang melakukan optimasi parameter pada hasil transformasi dengan dimensi lebih kecil dari dimensi ruang input.
2. Prinsip kerja SVM pada dasarnya hanya dapat menangani klasifikasi dua kelas, namun dikembangkan untuk mengklasifikasikan lebih dari dua kelas dengan menggunakan pengenalan pola.
3. *Catastrophe* didefinisikan sebagai masalah yang dihadapi oleh metode pengenalan pola dalam memperkirakan parameter karena jumlah sampel data yang relatif kecil dibandingkan dengan ruang dimensi vektor.
4. *Feasibility* SVM dapat diimplementasikan relatif lebih mudah, karena proses penentuan support vector dapat dirumuskan dalam Quadratic Programing (QP).

Kelemahan metode support vector machine adalah:

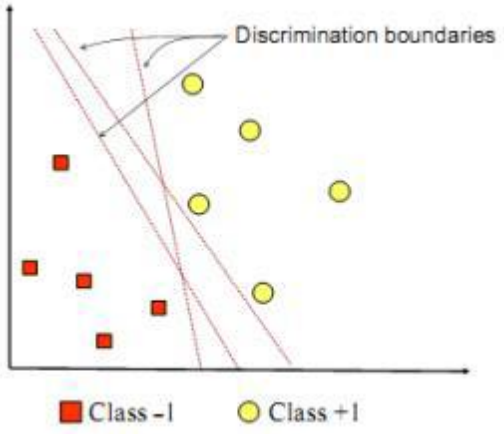
1. Sulit diterapkan pada masalah skala besar. Dalam hal ini berarti jumlah sampel yang diproses.
2. SVM secara teoritis dikembangkan untuk masalah klasifikasi dua kelas. Namun, SVM sekarang dimodifikasi untuk menyelesaikan lebih dari tiga kelas masalah (Nugroho, 2003).

Diagram berikut menunjukkan beberapa pola yang termasuk dalam dua kelas, 1 dan -1. Pola kelas -1 dilambangkan dengan warna kotak merah dan pola kelas 1 dengan lingkaran kuning. Masalah klasifikasi dapat diartikan dengan mencoba mencari garis (hyperplane) yang memisahkan dua kelompok.



Gambar 2.3 *SVM* berusaha untuk menemukan hyperplane terbaik yang memisahkan kedua kelas

Hyperplane pemisah yang optimal antara dua kelas dapat ditemukan dengan mengukur tepi hyperplane. dan temukan titik maksimumnya. Margin adalah jarak antara hyperplane dan pola terdekat untuk setiap kelas. Pola berikut disebut vektor dukungan. Garis solid pada gambar menunjukkan hyperplane yang optimal. Persis di tengah antara dua kelas, dan titik merah dan kuning di dalam lingkaran hitam adalah vektor pendukung. Mencoba menemukan lokasi hyperplane ini merupakan inti dari proses pembelajaran SVM.



Gambar 2.4. *Hyperplane* terbentuk diantara ke-dua kelas

Ada 2 metode yang umum digunakan dalam support vector machine:

1. Metode Linier

Model linier memiliki sifat penting baik dari aspek komputasi dan analitis. Penggunaan model linier dengan pendekatan parametrik dalam metode klasik memiliki aplikasi praktis yang terbatas karena kutukan dimensionalitas.

Terdapat 2 pendekatan dalam metode linier: pendekatan alternatif adalah membuat fungsi basis adaptif terhadap data latih dengan sejumlah fungsi basis yang telah ditentukan, pendekatan nonparametrik yaitu mendefinisikan data latih sebagai basis pusat.

2. Metode kernel

Sebagai salah satu fungsi linear, bentuk *SVM* secara umum dinyatakan dalam persamaan berikut

$(x) = wYx + b$ **(2.5)**

Di mana x adalah vektor input, w adalah parameter bobot, dan b adalah bias.

Fungsi kernel adalah untuk mengimplementasikan model dalam ruang dimensi yang lebih tinggi tanpa harus menentukan fungsi pemetaan dari ruang masukan ke ruang fitur. Salah satu contoh fungsi kernel yang banyak digunakan adalah *Gaussian radial basis function* (RBF) dan Karnel *Polinominal*, yaitu:

$K\left(x_i, x_j\right) = e^{-i\gamma||x_i - x_j||^2}, \gamma \geq 0$ **(2.6)**

di mana x ‘adalah “inti” yang dipilih dari data pelatihan.

Dan persamaan untuk Karnel *Polinominal*,

$(x^T x + 1).^q$ **(2.7)**

Kemudian diturunkan dengan fungsi berikut:

$K\left(x_i, x_j\right) = \left(x_i^T x_j + 1\right).^q$ **(2.8)**

di mana q adalah derajat polinomial

Ada 2 cara penggunaan metode kernel dalam pembelajaran mesin yaitu: Penggunaan langsung yaitu fungsi kernel yang digunakan sebagai fungsi dasar model *SVM*, contoh: fungsi jaringan basis radial, penggunaan tidak langsung merepresentasikan model menjadi beberapa representasi yang berisi produk dalam fungsi pemetaan, misalnya: kernel regresi linier, *kernel Perceptron*, mesin *vektor* dukungan, dll.

2.2.4. Stroke

Stroke juga dikatakan sebagai gangguan fungsi syaraf akut yang disebabkan karena gangguan peredaran darah otak secara mendadak (dalam hitungan detik) atau secara cepat (dalam hitungan jam) timbul gejala dan tanda yang sesuai dengan daerah fokal yang terganggu. Stroke terbagi dalam dua tipe, tipe pertama adalah stroke iskemik disebabkan kurangnya suplai darah ke otak dikarenakan menyempitnya atau tersumbatnya pembuluh darah oleh deposit lemak yang disebut plak sehingga jaringan otak mengalami iskemik. Tipe yang kedua adalah stroke hemoragik yang disebabkan pemecahan aneurisma pada parenchyma otak atau pada rongga antara otak dan tengkorak sehingga menyebabkan terjadinya iskemik dan desakan pada jaringan otak.[14] Gangguan lain berupa masalah fungsi syaraf lokal dan/atau global, munculnya mendadak, progresif, dan cepat. Gangguan fungsi syaraf pada stroke disebabkan oleh gangguan peredaran darah otak non traumatik. Gangguan syaraf tersebut menimbulkan gejala antara lain: kelumpuhan wajah atau anggota badan, bicara tidak lancar, bicara tidak jelas (pelo), mungkin perubahan kesadaran, gangguan penglihatan, dan lain-lain. Didefinisikan sebagai stroke jika pernah didiagnosis menderita penyakit stroke oleh tenaga kesehatan (dokter/perawat/bidan) atau belum pernah didiagnosis menderita penyakit stroke oleh nakes tetapi pernah mengalami secara mendadak keluhan kelumpuhan pada satu sisi tubuh atau kelumpuhan pada satu sisi tubuh yang disertai kesemutan atau baal satu sisi tubuh atau mulut menjadi mencong tanpa kelumpuhan otot mata atau bicara pelo atau sulit bicara/komunikasi dan atau tidak mengerti pembicaraan.

2.2.5. *Particel Sward Optimazion*

Particle swarm optimization, disingkat sebagai PSO, [15] didasarkan pada perilaku sebuah kawanan serangga, seperti semut, rayap, lebah atau burung. Algoritma PSO meniru perilaku sosial organisme ini. Perilaku sosial terdiri dari tindakan individu dan pengaruh dari individu-individu lain dalam suatu kelompok. Kata partikel menunjukkan, misalnya, seekor burung dalam kawanan burung. Setiap individu atau partikel berperilaku secara terdistribusi dengan cara menggunakan kecerdasannya (intelligence) sendiri dan juga dipengaruhi perilaku kelompok kolektifnya. Dengan demikian, jika satu partikel atau seekor burung menemukan jalan yang tepat atau pendek menuju ke sumber makanan, sis kelompok yang lain juga akan dapat segera mengikuti jalan tersebut meskipun lokasi mereka jauh di kelompok tersebut. Dalam konteks *optimisasi multivariat*, *swarm* diasumsikan berukuran pasti atau tetap, dengan setiap partikel memiliki posisi awal pada lokasi acak dalam ruang *multidimensi*. Semua partikel diyakini memiliki dua sifat: posisi dan kecepatan. Setiap partikel melintasi ruang tertentu dan mengingat posisi terbaik yang terjadi atau ditemukan dalam kaitannya dengan sumber makanan atau nilai fungsi tujuan. Setiap partikel mentransmisikan informasinya atau posisi yang disukainya ke partikel lain dan menyesuaikan posisi dan kecepatannya berdasarkan informasi yang diterima tentang posisi yang disukainya. Misalnya perilaku burung dalam kawanan burung. Setiap burung memiliki batasnya dalam hal kecerdasan,, biasanya ia akan mengikuti kebiasaan *(rule)* seperti berikut :

1. Seekor burung tidak berada terlalu dekat dengan burung yang lain.
2. Burung tersebut akan mengarahkan terbangnya ke arah rata-rata keseluruhan burung.
3. Seekor burung akan memposisikan diri dengan rata-rata posisi burung yang lain, sehingga jarak antar burung dalam kawanan itu tidak terlalu jauh.

Dengan demikian perilaku kawanan burung akan didasarkan pada kombinasidari 3 faktor simpel berikut:

1. Kohesi - terbang Bersama.

2. Separasi - jangan terlalu dekat.
3. Penyesuaian*(alignment*) - mengikuti arah bersama.

Oleh karena itu PSO dikembangkan berdasarkan model berikut:

1. Burung lain mengikuti arah makan, tetapi tidak secara langsung.

2. Ada satu faktor yang bergantung pada pikiran masing-masing burung. Ini adalah memori tentang apa yang terjadi di masa lalu. Model disimulasikan dalam ruang dimensi tertentu menggunakan beberapa iterasi sehingga posisi partikel pada setiap iterasi secara progresif mengarah ke tujuan yang diinginkan (minimisasi atau maksimalisasi fungsi). Ini berjalan hingga jumlah iterasi maksimum tercapai atau kriteria penghentian lainnya juga tersedia.

2.2.6. *Confusion Matrix*

Pengujian *Confusion Matrix* Pada tahap ini, model penelitian diuji dengan menggunakan metodefusion matrix. Metode ini menggunakan tabel matriks untuk menyajikan hasil evaluasi model. Jika kumpulan data terdiri dari dua kelas, kelas pertama dianggap positif dan kelas kedua dianggap negatif. Menskor menggunakan matriks kebingungan menghasilkan presisi, presisi, daya ingat, dan nilai-F. Akurasi klasifikasi adalah akurasi kumpulan data yang diklasifikasikan dengan benar setelah memeriksa hasil klasifikasi. Akurasi juga berlaku untuk data nyata. *Recall* adalah persentase kasus positif yang diprediksi dengan benar menjadi positif. *True positive* (*TP*) adalah jumlah catatan positif dalam kumpulan data yang diklasifikasikan sebagai positif. *True Negative (TN)* adalah jumlah catatan negatif yang diklasifikasikan sebagai positif dalam kumpulan data. *False Positif (FP)* adalah jumlah catatan negatif dalam kumpulan data yang diklasifikasikan sebagai positif. *False Negatives (FN*) adalah jumlah catatan positif dalam kumpulan data yang diklasifikasikan sebagai negatif. Di bawah ini adalah rumus model untuk matriks konfusi.[16]

Accuracy adalah jumlah perbandingan data yang benar dengan jumlah keseluruhan data.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \textbf{(2.9)}$$

Precision digunakan untuk mengukur seberapa besar proporsi dari kelas data positif yang berhasil diprediksi dengan benar dari keseluruhan hasil prediksi kelas positif.

$$Precision = \frac{TP}{FP+TP} \quad \textbf{(2.10)}$$

Recall digunakan untuk menunjukkan presentase kelas data positif yang berhasil diprediksi benar dari keseluruhan data kelas positif.

$$Recall = \frac{TP}{FN+TP} \quad \textbf{(2.11)}$$

2.3. Evaluasi Model

2.3.1. *Accuracy*

Accuracy menggambarkan seberapa akurat model dapat mengklasifikasikan dengan benar. Maka, accuracy merupakan rasio prediksi benar (positif dan negatif) dengan keseluruhan data. Dengan kata lain, accuracy merupakan tingkat kedekatan nilai prediksi dengan nilai aktual (sebenarnya). Nilai accuracy dapat diperoleh dengan persamaan (2.9).[17]

Akurasi adalah metrik yang digunakan untuk menilai kinerja model klasifikasi *machine learning*. *Accuraxy* juga merupakan metrik pembelajaran mesin yang paling sederhana dan paling banyak dipahami oleh *Final User* dan *Data Sciencetis*. Namun, kesederhanaannya juga merupakan kelemahannya karena kesulitan menyampaikan poin kesalahan dalam model *machine learning*

Sisi positif dari akurasi sebagai metrik kesalahan adalah:

1. Mudah diimplementasikan
2. Mudah dipahami oleh banyak orang

Negatif akurasi sebagai metrik kesalahan adalah:

1. Tidak berfungsi dengan baik pada kumpulan data yang tidak seimbang
2. Tidak dapat membedakan antara presisi dan kemampuan mengingat

Faktor negative inilah yang memberikan alasan kenapa harus berhati-hati dalam penggunaanya. Akurasi hanya boleh digunakan pada kumpulan data seimbang dan dalam konteks metrik lain yang menyediakan aspek lain dari kinerja model *machine learning*.[18]

Ada nilai batas umum untuk memahami skor akurasi:

1. *Over 90% - Very good*
2. *Between 70% and 90% - Good*
3. *Between 60% and 70% - OK*
4. *Below 60% - Poor*

2.3.2. *AUC (Area Under Curve)*

Menurut [19] *AUC* dapat diartikan sebagai probabilitas bahwa, ketika kita secara acak memilih satu contoh positif dan satu contoh negatif, pengklasifikasi akan memberikan skor yang lebih tinggi pada contoh positif daripada negative. Oleh karena itu, nilai AUC yang lebih tinggi menyiratkan kinerja klasifikasi yang lebih baik, menjadikannya sebagai tujuan maksimalisasi. Namun, setiap upaya untuk meringkas kurva ROC menjadi satu angka kehilangan informasi tentang pola trade-off dari algoritma diskriminator tertentu.

Performance keakurasian AUC dapat diklasifikasikan menjadi lima kelompok yaitu:

1. 0,90 – 1,00 = *Exellent Classification*
2. 0,80 – 0,90 = *Good Classification*
3. 0,70 – 0,80 = *Fair Classification*
4. 0,60 – 0,70 = *Poor Classification*
5. 0,50 – 0,60 = *Failure Classification*

ROC adalah kurva probabilitas dan AUC mewakili tingkat atau ukuran keterpisahan. Ini memberi tahu seberapa banyak model mampu membedakan antar kelas. Semakin tinggi AUC, semakin baik model dalam memprediksi kelas 0 sebagai 0 dan kelas 1 sebagai 1. Dengan analogi, semakin tinggi AUC, semakin baik model dalam membedakan antara pasien dengan penyakit dan tanpa penyakit.[20]

2.3.3. *F-Measurement (f-Score)*

F1-Score atau juga dikenal sebagai *F-measure*, atau *Balance F-Score* adalah metrik *Error* yang skornya berkisar dari 0 hingga 1, di mana 0 adalah skor kurang baik dan 1 adalah skor terbaik, nilai tertinggi menunjukan hasil lebih baik dari system klasifikasi.

$$F_1 = 2\frac{Precision * Recall}{Precision + Recall} \quad \textbf{(2.12)}$$

Perbedaan utama antara *F1-Score* dan *Accuracy* adalah kinerjanya pada kumpulan data yang tidak seimbang dan kemudahan mengkomunikasikan hasil kepada pengguna akhir. [21]

F1-score adalah rata-rata harmonik dari Precision dan Recall dan memberikan ukuran yang lebih baik untuk kasus yang salah diklasifikasikan daripada Accuracy Metric.

Untuk lebih singkatnya perbedaan *F1-Score* dengan *Acuraccy* ialah,

1. Akurasi digunakan ketika True Positives dan True Negatives lebih penting, sedangkan F1-score digunakan ketika False Negatives dan False Positives sangat penting
2. Akurasi dapat digunakan ketika distribusi kelasnya mirip sedangkan skor F1 adalah metrik yang lebih baik ketika ada kelas yang tidak seimbang seperti pada kasus di atas.

3. Dalam sebagian besar masalah klasifikasi kehidupan nyata, ada distribusi kelas yang tidak seimbang dan dengan demikian skor F1 adalah metrik yang lebih baik untuk mengevaluasi model kami. [22]