BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Support vector machine dapat digunakan untuk peramalan radiasi matahari *global* (GSR) yang menghasilkan minimal *error* dan memiliki akurasi yang tinggi (Ramedani dkk.,2014). Model SVM juga dikembangkan untuk klasifikasi radiasi matahari melalui variabel meteorologi terukur (Chen dan Li, 2014). Penggabungan SVM juga digunakan untuk meramalkan rata-rata bulanan GSR horizontal berdasarkan tiga variabel meteorologi di tiga tempat yang berbeda (Olatomiwa dkk., 2015), dan estimasi radiasi matahari di zona subtropis (Chen, 2015).

Support vector machine dengan menggunakan *radial basis function* (RBF) dapat digunakan untuk klasifikasi radiasi matahari dengan menganalisis faktor-faktor yang mempengaruhinya seperti temperatur, kelembaban dan radiasi matahari yang menghasilkan akurasi 89% (Piri dkk., 2015). Model SVM dapat juga dikembangkan untuk klasifikasi kejadian radiasi matahari berdasarkan data selama beberapa bulan dengan akurasi yang sesuai (Deo dkk., 2016). Fungsi regresi SVM pernah digunakan untuk memfokuskan klasifikasi rata-rata tahunan radiasi matahari global tahunan yang memberikan akurasi sangat baik (Baser dan Demirhan, 2017). Regresi maju pada *support vector machine* kernel kuadrat dapat digunakan untuk membangun metode regresi kuadrat menggunakan regresi maju dan memilih variabel *input* yang paling penting untuk memperkirakan radiasi horizontal *global* dan memberikan akurasi sesuai (Jiang dan Dong, 2017).

Beberapa contoh penelitian yang menerapkan *support vector machine* yaitu perkiraan harga listrik dapat menghasilkan prediksi yang sesuai dalam memperkirakan kelompok kelas (Ziel dan Steinert, 2018), penaksir daya baterai dengan model prediksinya menunjukkan hasil yang bagus (Antón dkk., 2013). Metode tersebut merupakan pembelajaran mesin yang relatif baru yang didasarkan pada *structural risk minimization* (SRM) yang muncul sebagai salah satu teknik terkemuka untuk klasifikasi pola dan pendekatan fungsi (Pan dkk., 2008). *Support*

vector machine juga telah berhasil digunakan untuk mengatasi banyak masalah perkiraan, seperti untuk memperkirakan radiasi matahari setiap hari berdasarkan durasi paparan sinar matahari yang menunjukkan hasil yang bagus dalam memperkirakan radiasi matahari (Chen dkk., 2013).

Klasifikasi produksi listrik pada panel surya dapat dilakukan dengan beberapa cara, salah satunya menggunakan *support vector machine* (SVM) dengan menggunakan *input* data suhu radiasi matahari dan lingkungan. Adapun dengan menggunakan algoritma linier (RBF), data masukan dalam ruang *fitur* berdimensi tinggi melatih set data masukan yang digunakan untuk mensimulasi dan mengklasifikasi *output* dari sistem energi matahari serta mengukur *output* produksi energi panel surya. Kegunaan lain dari *support vector machine* yaitu: penilaian kualitas *real-time* untuk proses pemberian *primer-sealer* dari jalur perakitan *sunroof* yang menghasilkan rata-rata akurasi sebesar 82% (Oh dkk., 2018), memperkirakan radiasi matahari harian dan rata-rata harian menggunakan suhu udara menghasilkan akurasi yang sesuai (Chen dkk., 2011).

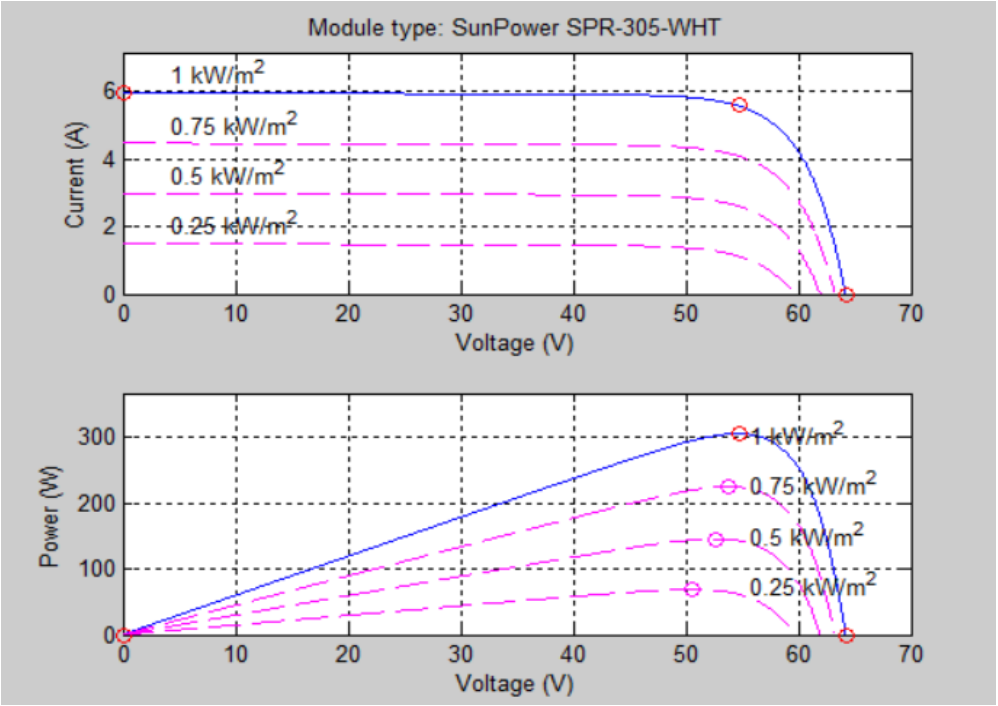
2.2 Dasar Teori

2.2.1 Panel Surya

Panel surya perlu dipantau untuk menjaga kondisi di lapangan agar setiap gangguan dapat ditangani secara tepat waktu (Suryono dan Khuriati, 2017). Panel surya bekerja dengan mengubah energi radiasi matahari menjadi energi listrik arus searah (DC) dan Intensitas sinar matahari diubah menjadi arus listrik oleh bahan semikonduktor yaitu: *monocrystalline silicon, polycrystalline silicon, amorphous silicon, telluride cadmium*, dan *indium gallium* melalui efek fotolistrik. Kinerja panel surya tergantung pada nilai *efisiensi* (η) dengan membandingkan daya listrik *output* ($P_{\max}$) dengan E merupakan energi *input* radiasi matahari (kW/m^2), A merupakan area pada panel surya (m^2) yang dapat dirumuskan pada persamaan (2.1).

$$\eta = \frac{P_{max}}{E.A} \tag{2.1}$$

Persamaan (2.1) dapat digunakan untuk mengetahui beberapa ketidakpastian yang terjadi saat panel surya beroperasi dari bermacam–macam faktornya seperti perbedaan panel surya, musim, lokasi geografis, cuaca, posisi jam matahari, tanggal pengamatan, waktu dan awan sebagai parameter yang menghambat kinerja panel surya (Zang dan Zang, 2017). Adapun gangguan lain yang mungkin terjadi seperti bayangan dari awan, pohon, rumah dan tiang antena (Ngoc dkk., 2017). Apabila o*utput* daya listrik yang terganggu, maka dapat didiagnosa dengan mengukur kinerja *input* dari panel surya (Denholm dkk., 2007). Karakteristik solar sel dalam kinerja matahari yang memancar tampak pada Gambar 2.1



Gambar 2.1 Karakteristik solar sel (Arumsari dan Pamuji, 2017)

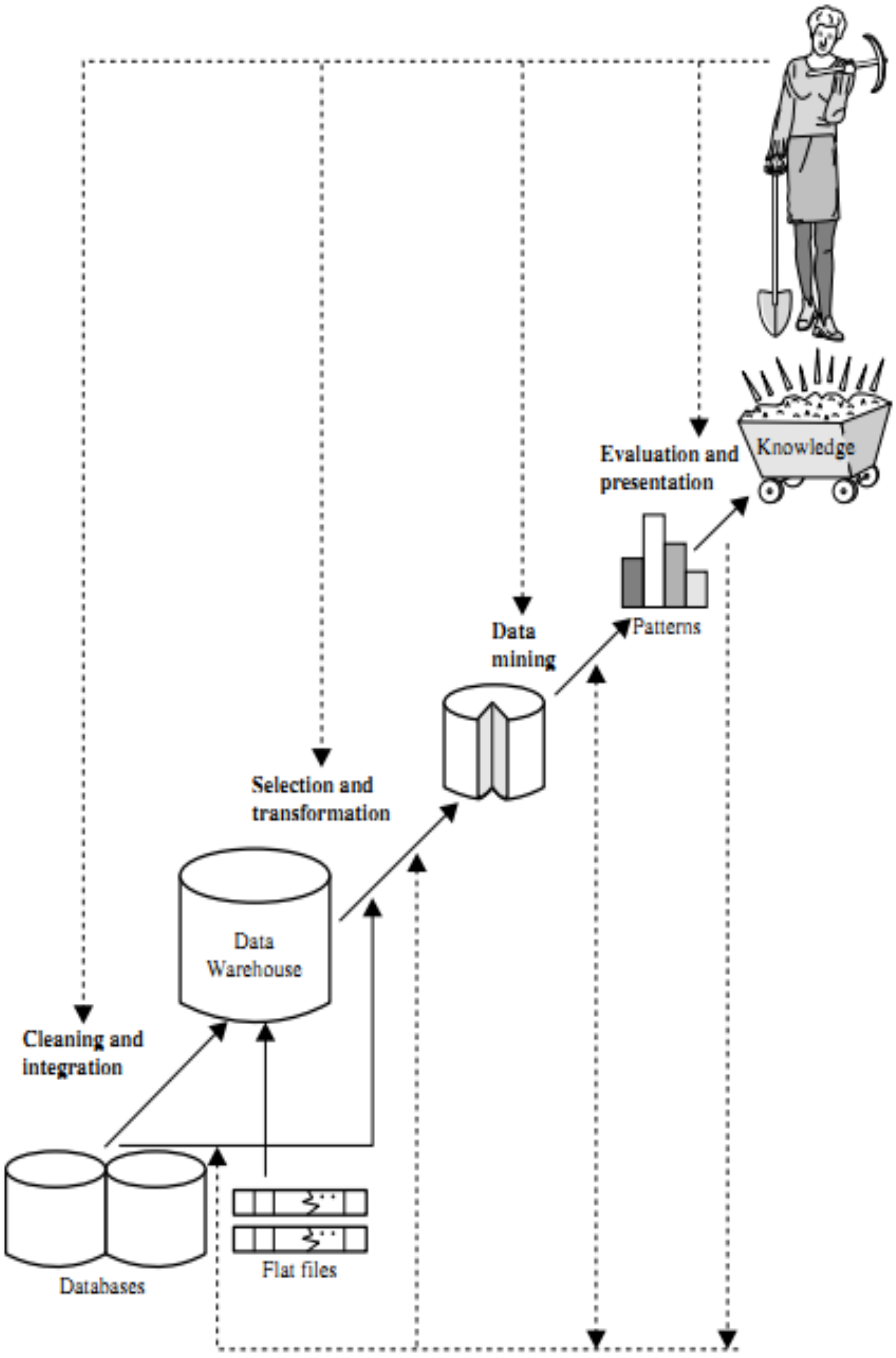
Gambar 2.1 menjelaskan bahwa pemancaran matahari yang berubah akan merubah daya panel surya. Pada tengah hari ketika matahari tepat di atas kepala, pemancaran dari cahaya matahari akan mencapai maksimum dan jika cuaca mendung maka pemancaran matahari akan berkurang karena terhalang oleh sinar

matahari yang mengakibatkan efisiensi dari surya sel dapat turun (Arumsari dan Pamuji, 2017).

2.2.2 *Data Mining*

Data mining digunakan untuk menemukan informasi yang berguna dari jumlah data yang besar. *Data Mining* juga merupakan suatu proses untuk menemukan pola dari suatu data. Proses pencarian pola pada *data mining* dilakukan secara otomatis (suatu bentuk, sifat, keadaan, kondisi, susunan tanpa keikutsertaan manusia secara aktif dalam proses pembuatan keputusan) dan semi-otomatis. *Data mining* juga disebut sebagai *Knowledge Discovery in Database* (KDD). Tahapan yang digunakan untuk mendapatkan pengetahuan atau pola dari *data mining* ditunjukkan pada Gambar 2.2 (Han dkk., 2011). Gambar 2.2 dapat dijelaskan bahwa proses untuk mendapatkan pengetahuan terdiri dari beberapa tahapan yang berurutan dan berulang-ulang yaitu: pembersihan data, integrasi data, pemilihan data, transformasi data, penambangan data, evaluasi pola dan presentasi pengetahuan.

Tahap pembersihan data dilakukan untuk menghapus data yang tidak konsisten dan *noise*. Tahap integrasi data digunakan untuk mengintegrasikan berbagai sumber data yang diolah. Tahap pemilihan data digunakan untuk proses pengolahan selanjutnya setelah memperoleh data yang terpilih. Data yang terpilih kemudian ditransformasi ke dalam format yang sesuai dengan menggunakan operasi agregasi (mendapatkan nilai dari sekumpulan data yang telah dikelompokkan misal maksimal, minimum, *count* (mencari cacah data), jumlah dan rata-rata). Selanjutnya dilakukan penambangan data yang merupakan tempat metode dari *support vector machine* digunakan untuk mendapatkan pola dari suatu data. Evaluasi pola digunakan untuk mengidentifikasi pola yang didapatkan dan merepresentasikan sebuah pengetahuan ataupun dengan berdasarkan beberapa pengukuran dan tahap yang terakhir yaitu presentasi pengetahuan yang digunakan untuk memvisualisasikan dan menggunakan beberapa teknik representasi data sehingga penyajian pola pengetahuan didapatkan untuk dapat dimengerti oleh pengguna.



Gambar 2.2 Langkah-langkah *data mining* (Han dkk., 2011)

2.2.3 *Normalization*

Normalization dilakukan untuk mengatasi variabel yang tidak konsisten dan mencegah terjadinya jumlah data yang lebih besar mendominasi jumlah data yang lebih kecil. Data dari panel surya dinormalisasi untuk mengatasi variabel yang

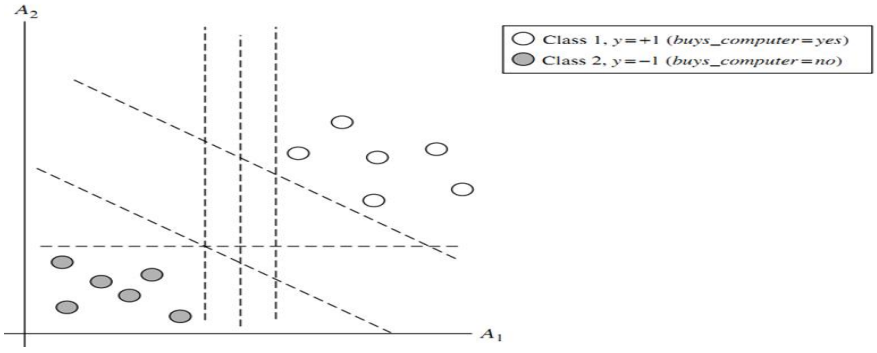
tidak konsisten dari *dataset* panel surya dengan menggunakan nilai rata-rata dari masing-masing variabel. Terdapat beberapa cara untuk dilakukan *data normalization* salah satunya adalah normalisasi minimal maksimal atau normalisasi. Dengan melakukan normalisasi hasil yang didapatkan akan lebih baik dan untuk penggunaan rentan nilai atribut yang dianjurkan yaitu [0,1]. Adapun rumus normalisasi untuk pengelompokan data panel surya ditunjukkan dengan persamaan (2.2).

$$x_i = \frac{x_i - x_{min}}{x_{max} - x_{min}} \tag{2.2}$$

dengan x_i adalah data *input* asli dan x_{max} dan x_{min} adalah setiap data *input* maksimum dan minimum pada masing-masing data (Zhang dkk., 2017). Jika x_{max} sama dengan x_{min}, maka normalisasi akan diubah menjadi 0,5 (Kumar dkk., 2015).

2.2.4 *Support Vector Machine*

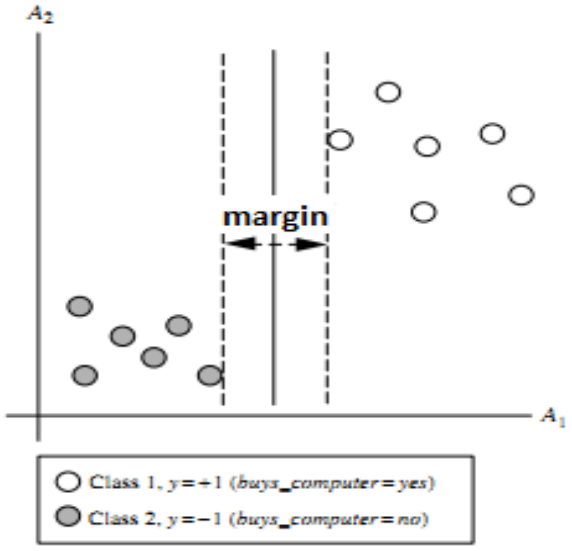
Support vector machine merupakan salah satu metode klasifikasi yang efektif dengan kinerja generalisasi tinggi dengan mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua kelas data (Wang dkk., 2018). Pemisahann dengan mencari *hyperplane* terbaik di tunjukkan pada Gambar 2.3.



Gambar 2.3 Pemisahan data pelatihan 2-D dengan banyak *hyperplane* (Han dkk., 2011)

Support vector machines merupakan kombinasi yang harmonis dari berbagai teori komputasi yang telah ada sebelumnya, misalkan *margin hyperplane* (Miller dan Vapnik, 1999). Jika *Hyperplane* pada *neural network* mencari pemisah antar kelas maka, pada SVM berusaha untuk menemukan pemisah terbaik pada ruang *input*.

Gambar 2.3 menjelaskan data dua dimensi secara linier dipisahkan karena garis lurus dapat ditarik untuk memisahkan semua kelompok kelas *C1* dan kelompok kelas *C2* sehingga masalah klasifikasi dapat diterjemahkan. Usaha tersebut dilakukan untuk menemukan garis *hyperplane* yang memisahkan antara kedua kelompok tersebut dengan berbagai alternatif garis pemisah. Jadi, jika terdapat bidang yang berdimensi *x* maka dapat diartikan bahwa *hyperplane* merupakan bidang yang berdimensi *x*–1. *Hyperplane* tersebut akan membagi bidang *x* menjadi dua bagian. Jika terdapat bidang satu dimensi, maka *hyperplane*nya adalah bidang 0 dimensi (titik). Jika terdapat bidang dua dimensi, maka *hyperplane*nya adalah bidang satu dimensi (garis) dan jika pada bidang 4 dimensi, maka *hyperplane* berupa bangun ruang dan seterusnya yang sulit untuk divisualisasikan. Pada *support vector machine*, jarak antara titik-titik positif dan negatif terdekat di sekitar *hyperplane* yang disebut *margin* yang tampak pada Gambar 2.4.



Gambar 2.4 *Hyperplane* yang dipisahkan oleh *margin* (Han dkk., 2011)

Gambar 2.4 menjelaskan bahwa secara umum jarak terpendek dari *hyperplane* ke satu sisi dari *margin* adalah sama dengan jarak terpendek dari *hyperplane* ke sisi lain dari *margin*nya dengan sisi dari *margin* yang sejajar dengan *hyperplane*. Adapun *hyperplane* terbaik antara kedua kelas yang dibentuk dapat ditemukan dengan mengukur *margin hyperplane* tersebut dan mencari titik maksimalnya. *Margin* yang dimaksud merupakan jarak antara *hyperplane* dengan data yang terdekat pada tiap kelas. Data terdekat yang termasuk dalam area *margin* dan *hyperplane* ini disebut dengan *support vector*.

Dasar dari SVM adalah pengklasifikasian linear dan pada perkembangannya SVM lebih dikembangkan agar dapat bekerja pada problem *non-linear* dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi. Pada kasus klasifikasi linear, fungsi pemisah dapat diartikan pada persamaan (2.3).

$$g(x) = sgn\big(f(x)\big)$$
$$f(x) = w^T\, x + b \tag{2.3}$$

Masalah klasifikasi pada persamaan (2.3) dapat dijelaskan dengan variabel w dan b untuk menemukan set parameter dan untuk menemukan dua macam obyek sebagai fungsi pemisah terbaik diantara fungsi yang tidak terbatas jumlahnya untuk menemukan dua macam obyek.

Setiap objek dataset dinyatakan oleh (x_i, y_i), dengan i = 1, 2, 3, … , m dan x_i = $[x_{i1}$, x_{i2}, .. , $x_{in}]^T$ merupakan himpunan fitur yang di simpan pada baris ke-i dari $m \times n$ matriks x. Objek atau label kelas dilambangkan oleh i dan y_i dengan mengartikan labelnya sebagai y_i = 1 jika objek i termasuk ke set I_1 atau y_i = −1 jika objek i termasuk ke set I_2. Dalam arti luas, *support vector machine* membangun *hyperplane* yang dipisahkan menggunakan fungsi pemisah pada persamaan (2.4). Dengan vektor w tegak lurus terhadap fungsi pemisah $wx+b$=0.

$$f\,(x) = w.x_i + b \;=\; 0 \tag{2.4}$$

atau

$$f(x) = \sum_{i=1}^{n} \alpha_i y_i K(x, x_i) + b = 0 \quad (2.5)$$

Maksimum *hyperplane margin* dapat ditemukan dengan menghitung sampel terdekat antara dua kelas (Wang dkk., 2013; Chau dkk., 2013). Adapun *x* merupakan data yang menjadi *support vector* dan x_i termasuk data yang akan diklasifikasikan. Data x_i yang termasuk dalam objek kelas -1 sebagai sample negatif dapat dirumuskan dalam persamaan (2.6).

$$w.x_i + b \leq -1 \quad (2.6)$$

Pola pada x_i yang termasuk objek kelas +1 sebagai sample positif dapat dirumuskan dalam persamaan (2.7).

$$w.x_i + b \leq +1 \quad (2.7)$$

Persamaan (2.4) mengklasifikasikan objek x_i sebagai dataset uji yang memaksimalkan pemisahan *margin* antara kedua kelas $\frac{2}{||w||}$ (Maldonado dkk., 2018). *Margin* yang tebesar dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya yaitu $\frac{1}{||w||}$ yang disebut dengan masalah *Quadratic Programming* (QP). *Quadratic programming* merupakan pencarian titik minimal pada persamaan (2.8) dengan memperhatikan pembatas (*constraint*) persamaan (2.9).

$$\tau\ (w) = \frac{1}{2}\|w\|^2 \quad (2.8)$$

$$y_i(w.x_i + b) - 1 \geq 0, i = 1, 2, \ldots, n \quad (2.9)$$

Adapun y_i merupakan label dari kelas {-1, +1} dan merupakan jumlah data yang menjadi *support vector*. Secara spesifik, nilai *margin* dapat diperoleh dengan perhitungan sebagai berikut:

$$(w.x_1 + b = +1) - (w.x_2 + b = -1)$$
$$w(x_1 - x_2) = 2$$
$$\left(\frac{w}{\|w\|}(x_1 - x_2)\right) = \frac{2}{\|w\|} \quad (2.10)$$

Adapun x_1 merupakan vektor masukan kelas +1, x_2 merupakan vektor masukan kelas -1 dan $\|w\|$ merupakan vektor bobot w. Bidang pembatas yang optimal dihitung dengan memaksimalkan jarak antara bidang pemisah dan data terdekat. Selanjutnya masalah ini diformulasikan ke dalam permasalahan *quadratic programming* dengan meminimalkan *invers* persamaan (2.8).

Jumlah fungsi pemisah yang digunakan tidak terbatas jumlahnya merupakan pembuktian bahwa memaksimalkan *margin* antara dua set obyek akan meningkatkan probabilitas pengelompokan secara benar dari data testing, misalnya dari jumlah yang tidak terbatas tersebut diambil dua variabel, yaitu $f_1(x)\ dan\ f_2(x)$. Fungsi f_1 memiliki *margin* yang lebih besar dari pada fungsi f_2. Setelah menemukan kedua fungsi ini, kemudian suatu data baru masuk dengan keluaran -1. Selanjutnya dari hasil tersebut harus dikelompokkan data baru tersebut berada dalam kelas -1 atau +1 menggunakan fungsi pemisah yang sudah ditemukan. Fungsi f_1 tersebut selanjutnya akan mengelompokkan data baru tersebut di kelas -1 yang mengartikan bahwa data benar dalam pengelompokannya. Selanjutnya menggunakan f_2 untuk menempatkannya di kelas +1 yang berarti salah.

2.2.4.1 *Non-Linear Support Vector Machine*

Permasalahan klasifikasi yang sebenarnya jarang yang bersifat *linear separable* dan kebanyakan bersifat *non-linear*. Hasil dari keputusan *non-linear support vector machine* dapat dimodifikasi dengan memasukkan fungsi kernel. Proses pembelajaran SVM bergantung pada *dot product* (perkalian titik) yang merupakan bentuk perkalian antara dua vektor yang akan menghasilkan skalar. Angka yang dihasilkan dari *dot product* tersebut tidak dapat menunjukkan besaran apapun sehingga sangat sulit untuk dimengerti secara mudah. Perhitungan *dot*

product dapat digantikan dengan fungsi kernel $K(x_i, x_j)$ yang hanya perlu menghitung $K(x_i, x_j)$ dan tidak perlu melakukan perhitungan dalam ruang berdimensi tinggi secara langsung yang disebut dengan kernel trik. Perhitungan *dot product* dapat dirumuskan pada persamaan (2.11).

$$K(x_i, x_j) = \Phi(x_i)^T \Phi(x_j) \tag{2.11}$$

Fungsi kernel yang biasa digunakan dalam *dot product* tampak pada Tabel 2.1 dengan fungsi kernel dan pengaturan parameter *default* yang umum digunakan.

Tabel 2.1 Fungsi kernel dan parameter *default* untuk pemilihan model. (Wang dkk., 2018)

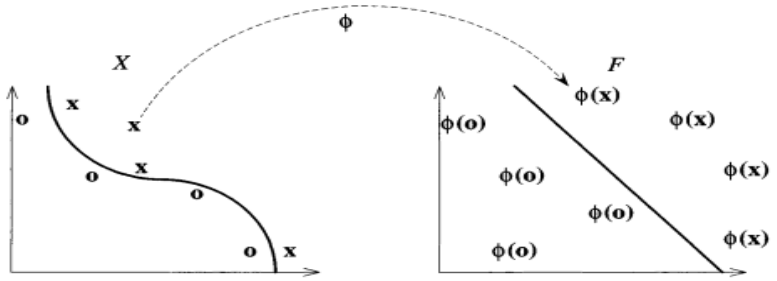
Tipe Kernel	Function	Default Parameter
RBF	$K_r(x_i, x_j) = exp(-2\sigma\|\|x_i - x_j\|\|^2)$	$\sigma = 1$
Polynomial	$K_p(x_i, x_j) = [(x_i, x_j) + 1]^r$	$r = 1$
Linear	$K_l(x_i, x_j) = (x_i, x_j)$	/
Laplacian	$K_{lap}(x_i, x_j) = exp(-\sigma\|\|x_i - x_j\|\|)$	$\sigma = 1$
HT	$K_t(x_i, x_j) = tanh((x_i, x_j) + b)$	$b = 1$
ANOVA RBF	$K_a(x_i, x_j) = (\Sigma_{k=1}^{s} exp(-\sigma\|\|x_i^k - x_j^k)^2))^d$	$\sigma = 1$, $d = 1$

Berbagai kernel dasar *support vector machine* memiliki kinerja masing-masing pada model dasar untuk menunjukkan dampak parameter model pada akurasi model. Penggunaan parameter dalam model *support vector machine* dapat secara signifikan mempengaruhi akurasi klasifikasi. Secara khusus akurasi klasifikasi bervariasi karena pilihan yang berbeda dari fungsi kernel dan struktur *support vector machine* yang memungkinkan *individu*nya memiliki batas ketepatan maksimal. Peningkatan potensial dapat dicapai dengan menggabungkan berbagai model *support vector machine* bersamaan untuk mengurangi kelemahan dari *support vector machine individual,* terutama pada struktur SVM tunggal (Wang dkk., 2018).

Dengan menerapkan x_i ke dalam metode *support vector machine,* maka akan menghasilkan produk yang berada diantara dua titik dalam ruang fitur (Zendehboudi dkk., 2018). Adapun x_i selanjutnya akan mentransfer dari ruang

fitur ke ruang dimensi yang lebih tinggi menggunakan metode kernel yang dapat memisahkan secara linear titik data yang sangat dan saling tumpang tindih di ruang baru yang merupakan fungsi SVM. Untuk klasifikasi yang dihasilkan menggunakan persamaan (2.5) dengan *f(x)* menunjukkan fungsi pemetaan.

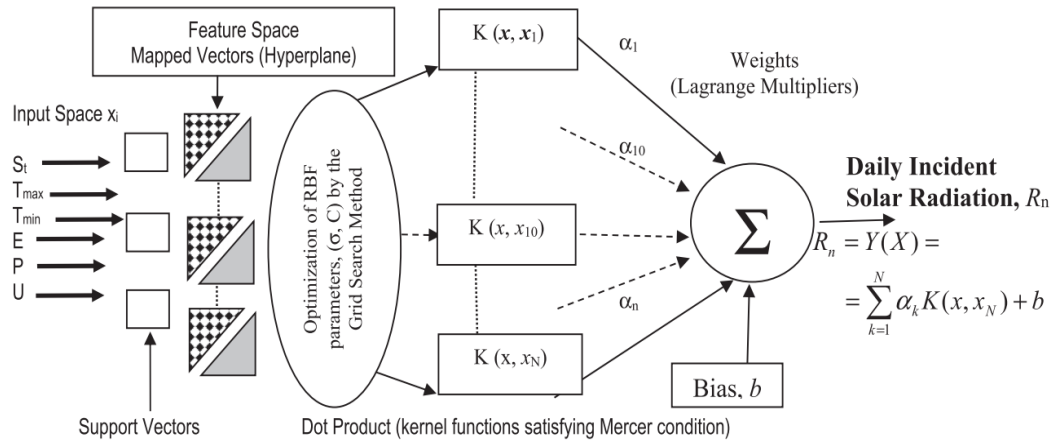
Pada kenyataanya hanya dengan memetakan data ke ruang lain merupakan tugas yang sangat sederhana dan telah memunculkan sejumlah teknik untuk memilih representasi data terbaik (Cristiani dan Taylor, 2000). Kuantitas yang diperkenalkan untuk mendeskripsikan data biasanya disebut fitur, sedangkan kuantitas aslinya terkadang disebut atribut. Tugas memilih representasi yang paling sesuai dikenal sebagai pemilihan fitur. Ruang X disebut sebagai ruang input, $F = \{\Phi(x): x \epsilon X\}$ disebut ruang fitur (*feature space*) yang tampak pada Gambar 2.5.



Gambar 2.5 Pemetaan fitur yang dapat menyederhanakan tugas klasifikasi (Cristiani dan Taylor, 2000)

Gambar 2.5 menunjukkan contoh dari pemetaan *feature* dari ruang masukan dua dimensi ke ruang *feature* dua dimensi yang datanya tidak dapat dipisahkan oleh fungsi linear di ruang masukan, tetapi bisa di ruang *feature*. Misalkan terdapat $x = (p_1^x, p_1^y, p_1^z, p_2^x, p_2^y, p_2^z, m_1, m_2)$, maka cara untuk mengurangi masalah dimensi adalah pemetaan $\Phi : R^8 \mapsto R^3$. Sehingga dapat diselesaikan dengan $x = \left(p_1^x, p_1^y, p_1^z, p_2^x, p_2^y, p_2^z, m_1, m_2\right) \mapsto \Phi(x) = \left(\sqrt{\sum_{i\in\{x,y,z\}}(p_1^i - p_2^i)^2}, m_1, m_2\right)$. Penggunaan utama pada analisis komponen yang menyediakan pemetaan data ke ruang fitur dengan fitur-fitur baru adalah fungsi linear dari atribut asli dan diurutkan berdasarkan jumlah varians yang

menunjukkan data di setiap arah. Pengurangan dimensi kadang-kadang dapat dilakukan dengan hanya menghapus fitur yang sesuai dengan arah di mana data memiliki varian rendah, meskipun tidak ada jaminan bahwa fitur-fitur ini tidak penting untuk melakukan klasifikasi. Skema dari model *support vector machine* yang menghasilkan keluaran model fungsi klasifikasi tampak pada Gambar 2.6.



Gambar 2.6 Skema model *support vector machine* (Deo dkk., 2016)

Gambar 2.6 merupakan gambaran skema dari alur SVM untuk mengklasifikasi pencahayaan matahari harian dengan variabel prediktor (masukan) adalah jam matahari (S_t), suhu maksimum (*Tmax*), suhu minimum (*Tmin*), evaporasi (*E*), pengendapan (*P*) dan kecepatan angin (*U*) untuk prakiraan harian radiasi matahari insiden global (R_n)). Ruang *feature* biasanya menciptakan vektor pendukung untuk proses regresif dan ruang keluaran berisi R_n yang diperkirakan sebagai variabel obyektif. Model SVM bertujuan untuk mengekstrak set pola optimal (*feature* prediktif) yang terdapat dalam *x time-series* untuk mengklasifikasi variabel objektif R_n.

Support vector machine memetakan titik-titik pelatihan dimensi *n* ke ruang fitur dimensi yang lebih tinggi menggunakan fungsi pemetaan (Scholkopf dan Smola, 2002). Titik-titik sampel data yang bersesuaian dengan koefisien bukan nol dengan kesalahan estimasi yang sama dengan atau lebih besar dari ε disebut vektor-vektor pendukung (*support vectors*). Sedangkan, titik-titik data di wilayah

yang tidak sensitif tidak merupakan *support vector* dan tidak sebagai *support vector*. Biasanya, semakin besar variabel *slack (ε)* maka semakin sedikit jumlah *support vector* dan semakin jarang penggambaran solusi, sehingga titik data sample *n* yang dispesifikasikan dan ε-SVM menggunakan perhitungan *2n*X*2n* matriks kernel. Untuk persamaan dengan memasukkan variabel *slack* dapat ditunjukkan pada persamaan (2.12).

$$\frac{1}{2}\|w\|^2 + C(\sum_{i-1}^{n} \varepsilon) \tag{2.12}$$

Misalkan pada penyelesaian klasifikasi produksi listrik menggunakan kernel RBF yang didefinisikan pada persamaan (2.13).

$$K(x_i , x_j) = exp.\left(- \frac{||x_i - x_j||^2}{2\sigma^2}\right) \tag{2.13}$$

dengan σ adalah parameter Gauss yang merupakan lebar dari kernel RBF dan menentukan tempat yang sama dari *support vector* di atas ruang data, $x_i \; dan \; x_j$ merupakan masukan dari setiap i dan j dari dalam dimensi dengan $x_j = x_i$. Satu set kernel alternatif berdasarkan persamaan linear $(x_{i,} x_j)$. Persamaan RBF lebih banyak digunakan karena merupakan fungsi yang paling umum dan menawarkan kinerja yang baik yang terkait dengan dukungan ringkasnya dan persyaratan sederhana untuk memilih hanya satu parameter lebar kernel. Dalam penggunaan komputasional RBF dapat dijalankan dengan efisien karena pelatihannya hanya membutuhkan solusi dari satu set persamaan linear daripada komputasi yang menuntut masalah pemrograman kuadratik. Dalam model SVM yang terpenting adalah untuk mengoptimalkan faktor regularisasi (*C*) dan lebar kernel (σ). Secara khusus, bahwa besarnya σ (lebar kernel) menentukan area perkiraan pengaruh *support vector* dalam ruang optimasi *feature*. Adapun *C* digunakan untuk menyeimbangkan pemasangan dalam tahap pelatihan model dan generalisasi data dalam tahap implementasi. Maka dari itu, sangat penting untuk mencapai kinerja terbaik sehingga perlu mengatur *C* dan parameter dengan benar.

Dengan menggunakan teori dualitas, *support vector machine* mampu membangun klasifikasi *nonlinier* tanpa perlu mendefinisikan fungsi pemetaan

tersebut. Pesamaan (2.14) merupakan contoh-contoh pelatihan yang hanya muncul sebagai produk skalar yang memungkinkan penggunaan fungsi kernel $K(x_i\ x_j) = \varphi(x_i).\varphi(x_j)$ yang merupakan penggunaan fungsi perkalian dalam (*inner product*) pada ruang *feature*. Adapun versi lain yang berbasis kernel untuk metode *support vector machine* yaitu mengikuti *soft-margin*.

$$max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(x_i, x_s) \qquad (2.14)$$

$$\sum_{i=1}^{m} \alpha_i y_i = 0, \ \ 0 \leq \alpha_i \leq C, i = 1, \dots, m$$

dengan α_i merupakan pengali Lagrange yang berkorespondensi dengan x_s dan nilai α_i adalah 0 atau positif. Nilai x_i dan x_s merupakan perkalian dua titik data dalam dataset. Dari persamaan 2.13 diperoleh nilai α_i yang nantinya digunakan untuk menemukan *w* dan terdapat nilai α_i untuk setiap dataset. Data pelatihan yang memiliki nilai $\alpha_i > 0$ adalah *support vector* sedangkan sisanya memiliki $\alpha_i = 0$. Formula pemisahan ini disebut pemrograman kuadratik sehingga nilai maksimum global dari α_i selalu dapat ditemukan. Setelah solusi permasalahan ditemukan, maka kelas dari data pengujian *x* dapat ditentukan berdasarkan nilai fungsi keputusan.

Umumnya, karakteristik metode SVM dapat secara singkat dinyatakan sebagai metode yang sangat tepat dan kuat, mampu memodelkan batas keputusan *non-linear* yang kompleks, memperlihatkan deskripsi ringkas dari model yang dipelajari, serta memiliki potensi implementasi dalam pengenalan pola, regresi dan klasifikasi (Zendehboudi dkk., 2018).

2.2.5 *Sequential Training*

Optimasi parameter merupakan alternatif yang dapat digunakan dengan sederhana untuk mengatasi *hyperplane* yang optimal pada *support vector machine* dengan merumuskan ke dalam *Quadratic Programming* (*QP*) *problem* yang nantinya akan diselesaikan dengan analisa numerik yaitu *Sequential Training* (Vijayakumar, 1999) langkah-langkahnya adalah sebagai berikut:

1. Inisialisasi α_1=0.1 setelah itu hitung matrik *Hessian*. Matrik *hessian* merupakan perkalian antara kernel *gaussian* dengan nilai *Y*. Nilai *Y* ini adalah nilai vektor yang berisi nilai -1 dan 1. α_1 digunakan untuk mencari nilai *support vector*. pada setiap data dari *i* sampai *j*, kemudian hitung menggunakan rumus matrik *hessian* yang ditunjukkan sebagai berikut:

$$D_{ij} = y_i y_j \, K(x_i , x_j)+\lambda^2 \tag{2.15}$$

2. Dilanjutkan pada langkah kedua dengan melakukan rumus berikut:

$$E_i = \sum_{j=i}^{n} \alpha_i D_{ij} \tag{2.16}$$

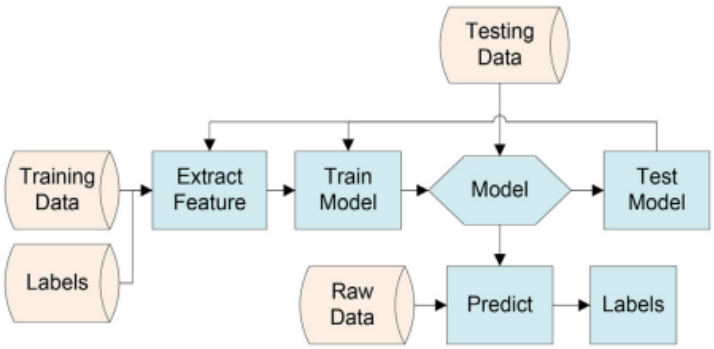
$$\delta\alpha_i = min(max[\gamma(1 - E_i), -\alpha_i], C - \alpha_i) \tag{2.17}$$

$$\alpha_i = \alpha_i + \delta\alpha_i \tag{2.18}$$

Dengan α_i diketahui sebagai alfa ke-i , D_{ij} merupakan matriks *hessian*, E_i merupakan *error rate*, C merupakan konstanta dan $\delta\alpha_i$ merupakan nilai delta alfa ke-i. Setelah itu kembali ke langkah 2 sampai nilai α_i mencapai konvergen. Konvergensi diperoleh dari tingkat perubahan pada nilai α_i yang tidak ada perubahan signifikan.

2.2.6 Klasifikasi

Klasifikasi dilakukan dengan memprediksi nilai dari suatu variabel numerik dari sebuah kelas sehingga klasifikasi hampir mirip dengan prediksi. Klasifikasi sendiri dilakukan untuk prediksi sebuah kelas saja, tetapi dalam istilah prediksi merujuk pada prediksi nilai pada variabel selanjutnya yang biasanya menggunakan istilah estimasi. Teknik pembelajaran mesin biasanya diklasifikasikan menjadi tiga kategori besar yaitu pembelajaran yang diawasi (misalnya: klasifikasi dan regresi), pembelajaran tanpa pengawasan (misalnya: *clusterisasi* atau pengelompokan) dan pemfilteran kolaboratif yang merupakan gabungan dari pembelajaran yang diawasi dan tidak diawasi (Wu dkk., 2017). Gambaran dari proses pembelajaran mesin yang diawasi tampak pada Gambar 2.11.



Gambar 2.11 Diagram proses pembelajaran mesin yang diawasi (Wu dkk., 2017)

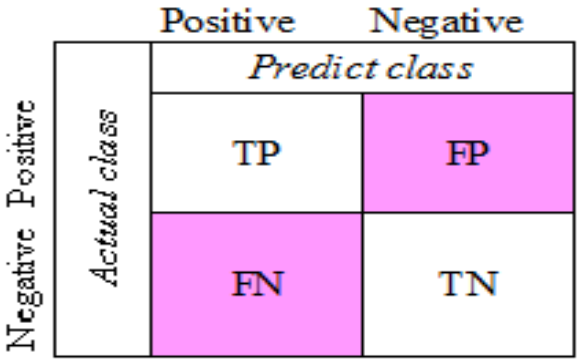
Algoritma pembelajaran mesin membutuhkan dua jenis data yaitu data pelatihan (*testing*) dan pengujian (*training*). Serangkaian fitur atau atribut diekstraksi sebagai masukan untuk algoritma pembelajaran berdasarkan kumpulan data pelatihan dan data berlabel. Perangkat data pelatihan juga digunakan untuk melatih algoritma pembelajaran. Selanjutnya dataset pengujian digunakan untuk mengevaluasi algoritma pembelajaran. Setelah model pembelajaran mesin dievaluasi, hasilnya dapat digunakan untuk mengklasifikasi potensi hasil dari suatu peristiwa.

Dalam IoT, pembelajaran mesin dapat digunakan untuk produksi konsumsi listrik dan klasifikasi produksi sehingga dapat menentukan kapan pemeliharaan harus dilakukan. Pembelajaran mesin pada volume data pelatihan membutuhkan data yang sangat besar, sehingga membutuhkan sejumlah besar memori. Penerapan algoritma tersebut menjadikan pembelajaran mesin di jaringan *cloud* menjadi lebih cepat dalam pemodelan klasifikasi.

2.2.7 *Confusion Matrix*

Pengukuran kinerja klasifikasi merupakan salah satu teknik pengukuran kinerja algoritma klasifikasi yang menggambarkan seberapa baik sistem dalam menjalankan klasifikasi data. Proses *confusion matrix* membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya. Pengukuran kinerja algoritma *support vector machine* dengan mengunakan *confusion matrix* menunjukkan kinerja teknik klasifikasi yang berhubungan

dengan data uji. Pengukuran kinerja pada *confusion matrix* biasanya digunakan pada dua kelas yaitu kelas positif dan kelas negatif yang ditunjukkan pada gambar 2.12



Gambar 2.12 *Confusion Matrix* (Ting, 2017)

Confusion matrix dibentuk oleh empat sel yaitu *true posistive* (*TP*), *false positive* (*FP*), *true negative* (*TN*), dan *false negative* (*FN*) yang merupakan representasi hasil proses klasifikasi. Nilai *true negative* (*TN*) menunjukkan jumlah data negatif yang terdeteksi dengan benar sebagai negatif, *false positive* (*FP*) menunjukkan data negatif yang terdeteksi sebagai data positif, *true positive* (*TP*) menunjukkan data positif yang terdeteksi benar sebagai positif, dan *false negative* (*FN*) menunjukkan data positif yang terdeteksi sebagai data negatif (Ting, 2017). Nilai-nilai sel tersebut dapat menghasilkan nilai akurasi, presisi dan *recall* dan nilai *error*. Persamaan nilai akurasi, presisi, *recall dan error* dapat diperoleh dengan menggunakan persamaan berikut:

1. Nilai akurasi menunjukkan seberapa akurat sistem dapat mengklasifikasikan data secara benar, sehingga membandingkan antara data yang terklasifikasi benar dengan keseluruhan data.

$$\text{Akurasi} = \left(\frac{TP+TN}{TP+TN+FP+FN} \right) \text{X}100\% \qquad (2.19)$$

2. Nilai presisi menunjukkan jumlah data kategori positif yang diklasifikasikan secara benar dibagi dengan total data yang diklasifikasi positif.

$$\text{Presisi} = \left(\frac{TP}{TP+FP}\right) \text{X } 100\% \quad (2.20)$$

3. Nilai *recall* menggambarkan berapa persen data pada kelas positif yang terklasifikasikan dengan benar oleh sistem.

$$\textit{Recall} = \left(\frac{TP}{TP+FN}\right) \text{X } 100\% \quad (2.21)$$

4. Nilai *error* menggambarkan berapa persen data yang tidak terklasifikasi dengan benar.

$$\textit{Error rate} = \left(\frac{FP+FN}{TP+TN+FP+FN}\right) \text{X } 100\% \quad (2.22)$$