BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Sebelumnya

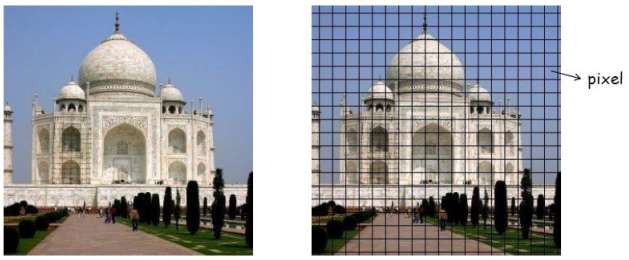
Dari hasil penelitian yang telah dilakukan sebelumnya pada tahun 2017 oleh Fandiansyah dkk., menggunakan metode LDA dan KNN di uji menggunakan 66 citra dari 22 individu didapatkan tingkat akurasi sebesar 98.33% [3]. Dari hasil penelitian yang telah dilakukan sebelumnya pada tahun 2022 oleh Alfi Nurakbar menggunakan metode PCA dan SVM didapatkan tingkat akurasi dan waktu komputasi pada dataset AT&T didapatkan tingkat akurasi sebesar 90%, pada dataset Georgia Tech didapatkan tingkat akurasi sebesar 70%, pada dataset Yale Face didapatkan tingkat akurasi sebesar 98%, didapatkan hasil akurasi 92% untuk dataset Mandiri [2]. Dari hasil penelitian yang telah dilakukan sebelumnya pada tahun 2022 oleh Agus Mulyadi menggunakan metode PCA dan KNN didapatkan tingkat akurasi dan waktu komputasi pada dataset AT&T didapatkan tingkat akurasi sebesar 85%, pada dataset Georgia Tech didapatkan tingkat akurasi sebesar 62%, pada dataset Yale B didapatkan tingkat akurasi sebesar 63%, didapatkan hasil akurasi 74% untuk dataset Mandiri [4].

2.2 *Image Processing* (Pengolahan Citra)

Citra adalah sinyal dwimatra (dua dimensi) yang bersifat kontinu yang dapat diamati oleh sistem visual manusia. Secara matematis, citra adalah fungsi dwimatra yang menyatakan intensitas cahaya pada bidang dwimatra disimbolkan dengan f (x, y), dalam hal ini (x, y) adalah koordinat pada bidang dwimatra dan f (x, y) adalah intensitas cahaya (*brightness*) pada titik (x, y) [5]. Citra digital adalah representasi citra melalui pencuplikan (sampling) secara ruang dan waktu. Pencuplikan (sampling) adalah proses untuk menentukan warna pada piksel tertentu pada citra dari sebuah gambar yang kontinu. Citra digital direpresentasikan sebagai matriks berukuran M $\times$ N. setiap elemen matriks menyatakan sebuah pixel (*picture element*). M $\times$ N menyatakan resolusi citra [6].

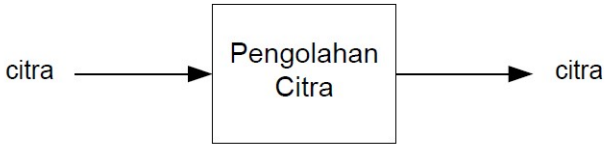
$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

Berikut adalah contoh dari suatu citra dengan resolusi 1200×1500 yang berarti memiliki 1.800.000 pixel. Pada Gambar 2.1 merupakan contoh dari suatu gambar dengan resolusi 1200×1500.



Gambar 2.1. Citra dengan resolusi 1200×1500

Pengolahan citra atau *Image Processing* adalah metode suatu sistem pemrosesan untuk mengubah citra menjadi bentuk digital dan melakukan beberapa operasi pada gambar untuk menemukan gambar yang disempurnakan dan mengekstrak informasi berguna darinya. Tujuan pengolahan citra adalah memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan [7].



Gambar 2.2. Teknik Pengolahan Citra

Meskipun sebuah citra kaya informasi, namun seringkali citra mengalami penurunan mutu (degradasi), misalnya mengandung derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang. Agar citra yang mengalami gangguan mudah diinterp-

retasi, maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik.

2.3 Fitur Ekstraksi

Fitur merupakan karakteristik dari suatu objek. Fitur dibedakan menjadi dua yaitu fitur alami merupakan bagian dari gambar, misalnya kecerahan dan tepi objek. Sedangkan fitur buatan merupakan fitur yang diperoleh dengan operasi tertentu pada gambar [8]. Fitur ekstraksi merupakan suatu pengambilan ciri (*feature*) dari suatu bentuk yang nantinya nilai yang didapatkan akan dianalisis untuk proses selanjutnya. Fitur ekstraksi bertujuan untuk mencari daerah fitur yang signifikan pada gambar tergantung pada karakteristik intrinsik dan aplikasinya. Daerah tersebut dapat didefinisikan dalam lingkungan global atau lokal dan dibedakan oleh bentuk, tekstur, ukuran, intensitas, sifat statistik, dan sebagainya. *Feature Extraction* dilakukan dengan cara menghitung jumlah titik (pixel) yang ditemui dalam setiap pengecekan, dimana pengecekan dilakukan di berbagai arah pada koordinat *cartesius* dari citra digital yang dianalisis, yaitu vertikal, horizontal, diagonal kanan, dan diagonal kiri. Koordinat *cartesius* adalah sistem koordinat berupa susunan garis dan titik dalam dua dimensi [9].

Ekstraksi ciri citra merupakan tahapan mengekstrak ciri dari objek di dalam citra yang ingin dikenali atau dibedakan dengan objek lainnya. Ciri yang telah diekstrak kemudian digunakan sebagai parameter atau nilai masukan untuk membedakan antara objek satu dengan lainnya pada tahapan klasifikasi [10].

Berikut ini adalah beberapa contoh algoritma metode fitur ekstraksi:

1. *Principal Component Analysis* (PCA)
   Algoritma ini menemukan proyeksi yang dapat merepresentasikan data dalam ruang dimensi yang lebih rendah, dengan mempertahankan sebanyak mungkin variasi dalam data. Dalam hal ini, PCA dapat mengurangi dimensi data dengan memilih sejumlah komponen utama yang paling penting untuk merepresentasikan data. [11].

2. *Linear Discriminant Analysis* (LDA)
   Algoritma ini menemukan proyeksi yang dapat memisahkan antara kelas data secara efektif dengan memaksimalkan varian antar kelas dan meminimalkan varian dalam kelas data. [12].

3. *Histogram of Oriented Gradients* (HOG)
   Algoritma ini merepresentasikan citra dalam bentuk histogram orientasi gra-

dien yang dapat menggambarkan pola tekstur pada citra dengan lebih baik. [13].

4. *Local Binary Patterns* (LBP)
   Algoritma ini mengekstrak fitur dari citra digital dalam bentuk pola tekstur yang terdapat pada gambar. [14].

5. *Deep Convolutional Neural Network* (DCNN)
   Algoritma ini mengekstrak fitur dari citra dengan mengekstrak fitur dari citra digital menggunakan jaringan saraf tiruan (*neural network*) dengan arsitektur *convolutional*. [15].

2.4 Metode Klasifikasi

Klasifikasi adalah pengaturan sistematis dalam pembagian atau pengelompokkan hal berdasarkan kesamaan sifatnya [16]. Metode klasifikasi adalah salah satu algoritma pada data *mining* yang mengelompokan suatu data ke dalam kriteria atau kategori tertentu dengan membaca data sebelumnya yang sudah ada. Tujuan utama dari klasifikasi adalah untuk membantu praktisi data dalam menentukan kelas atau kategori dari data baru berdasarkan karakteristik data yang telah ada sebelumnya [17].

Berikut ini adalah beberapa contoh algoritma metode klasifikasi:

1. *Naïve Bayes* (NB)
   Algoritma ini melakukan klasifikasi pada data dengan menggunakan teorema *Bayes*, dimana algoritma ini memanfaatkan probabilitas kondisional untuk menentukan kelas dari suatu data. [18].

2. *K-Nearest Neighbors* (KNN)
   Algoritma ini melakukan klasifikasi pada suatu data dengan cara mencari tetangga terdekat dari suatu data. [19].

3. *Support Vector Machine* (SVM)
   Algoritma ini melakukan klasifikasi dengan menemukan *hyperplane* terbaik yang dapat memisahkan data dalam ruang fitur menjadi dua kelas yang berbeda secara optimal. [20].

4. *Decision Tree*
   Algoritma ini melakukan klasifikasi dengan membuat sebuah model prediksi berdasarkan serangkaian keputusan yang diambil berdasarkan fitur-fitur dari data yang diberikan. [21].

5. *Random Forest*

   Algoritma ini melakukan klasifikasi dengan menggabungkan beberapa pohon keputusan (*Decision Tree*) yang dibangun pada dataset yang berbeda-beda. Dengan menggabungkan beberapa pohon keputusan, *Random Forest* dapat mengurangi *overfitting* pada model dan meningkatkan kemampuan dalam menggeneralisasi data yang belum pernah dilihat sebelumnya. [22].

2.5 Dataset

Dataset adalah istilah informal yang merujuk pada kumpulan data. Secara umum, dataset berisi lebih dari satu variabel dan menyangkut topik tertentu. Dataset digunakan untuk klasifikasi dengan metode data *mining*. Data *mining* adalah suatu proses pengumpulan informasi penting dari suatu data yang besar, yang sering kali menggunakan metode statistika, matematika, *machine learning*, hingga memanfaatkan teknologi *Artificial Intelligence* (AI) [23]. Pengertian dataset adalah sebuah kumpulan data yang berasal dari informasi-informasi pada masa lalu dan siap untuk dikelola menjadi sebuah informasi baru [24].

2.5.1 Dataset AT&T

Dataset wajah AT&T berisi citra wajah yang diambil April 1992 dan 1994. Ada 10 data wajah yang berbeda dari masing – masing 40 subjek berbeda. Semua citra gambar diambil dengan latar belakang gelap dengan posisi subjek tegak. Dataset tersebut berformat PGM, dengan ukuran setiap citra adalah 92×112 piksel, dengan 256 tingkat keabuan per piksel [25]. Di bawah ini adalah contoh citra dataset dari AT&T.

Gambar 2.3. Dataset AT&T

Tujuan dari penggunaan dataset AT&T yaitu sebagai percobaan pengujian pada sistem pengenalan, sebelum sistem diuji dengan menggunakan dataset yang dibuat secara mandiri. Selain itu, dataset AT&T juga digunakan sebagai pembanding hasil tingkat akurasi pengenalan wajah dengan dataset lainnya.

2.5.2 Dataset Yale Face

Dataset Yale Face berisi 165 gambar *grayscale* dari 15 orang, dengan 11 gambar dari setiap orang dalam kondisi pencahayaan yang berbeda. Gambar diperoleh pada akhir tahun 1990-an dan memiliki resolusi yang relatif rendah (sekitar 320×243 piksel). Gambar dikumpulkan dengan lingkungan yang terkendali dan kondisi standar, di mana latar belakang dan pose wajah konsisten. Ini membuatnya berguna untuk menguji algoritma yang dirancang untuk menangani variasi pencahayaan, pose, dan ekspresi [26]. Di bawah ini adalah contoh citra dataset Yale Face.

Gambar 2.4. Dataset Yale Face

2.5.3 Dataset Georgia Tech

Data wajah Georgia Tech berisi gambar 50 orang yang diambil dalam dua atau tiga sesi antara 01 Juni 1999 dan 15 November 1999 di Pusat Pemrosesan Sinyal dan Gambar di Institut Teknologi Georgia.

Semua orang dalam *database* diwakili oleh 15 gambar JPEG berwarna dengan *background* berantakan yang diambil pada resolusi 640×480 piksel. Ukuran rata – rata wajah dalam gambar ini adalah 150×150 piksel. Gambar menunjukkan wajah depan dan miring dengan ekspresi wajah, kondisi pencahayaan, dan skala yang berbeda. Setiap gambar diberi label secara manual untuk menentukan posisi wajah dalam gambar [27]. Di bawah ini adalah contoh citra dataset dari Georgia Tech.

Gambar 2.5. Dataset Georgia Tech

2.5.4 Dataset Mandiri

Dataset Mandiri merupakan data citra wajah yang diambil secara mandiri. Terdapat 10 data citra wajah yang berbeda dari 70 subjek berbeda. Semua citra wajah diambil dengan posisi tegak dan berbeda ekspresi. Dataset tersebut berformat JPG dengan ukuran setiap citra wajah adalah 92×112 piksel. Dibawah ini adalah contoh citra wajah dari dataset Mandiri.

Gambar 2.6. Dataset Mandiri

2.6 *Linear Discriminant Analysis* (LDA)

LDA adalah metode pengenalan wajah yang dikenal sebagai *Fisher's Linear Discriminant* yang merupakan pengembangan dari algoritma PCA. LDA dipergunakan untuk memaksimalkan penyebaran antar kelas dan meminimalkan penyebaran dalam kelas data citra wajah. Perbedaan antar kelas diwakilkan oleh *scatter between class* (matriks Sb) dan untuk perbedaan dalam kelas diwakilkan oleh *scatter within class* (matriks Sw) [28].

Pada tahapan *processing*, akan diterapkan metode LDA untuk menghasilkan vektor fitur citra dari citra wajah. Ekstraksi fitur dilakukan pada citra yang ada di dalam *database*. Pada tahap awal proses ekstraksi fitur pada penelitian ini digunakan algoritma *Eigenface* sebagai salah satu algoritma berbasis metode PCA untuk memperoleh bobot. Bobot tersebut selanjutnya akan digunakan pada metode LDA untuk membentuk vektor fitur citra wajah.

Berikut ini adalah langkah-langkah proses ekstraksi fitur dengan menggunakan metode LDA:

1. Perhitungan *Eigenface*

   Eigenface merupakan salah satu algoritma pengenalan wajah manusia yang berbasis metode PCA. Untuk menghasilkan *Eigenface*, sekumpulan citra digital dari wajah manusia diambil pada kondisi pencahayaan yang sama kemudian dinormalisasikan dan diproses pada resolusi yang sama (misal m $\times$ n), kemudian citra tersebut diperlakukan sebagai vektor dimensi m $\times$ n dimana komponen nya diambil dari nilai piksel citra [29].

   Berikut langkah-langkah Algoritma *Eigenface* untuk memperoleh bobot dari citra *database*:

   a. Mengubah M jumlah citra di *database* dengan ukuran m = B $\times$ K ke dalam bentuk vektor Γ dengan panjang 1 $\times$ m. B adalah baris dan K adalah kolom dapat direpresentasikan dengan Γ_1, Γ_2, Γ_3,..., Γ_m. Citra wajah dapat dituliskan sebagai Γ = [Γ_1, Γ_2, Γ_3,..., Γ_m], dimana setiap Γ adalah vektor dari dimensi m dan jumlah citra wajah dalam *database* disimbolkan oleh M.

   $$m = \begin{bmatrix} U_{1,1} & \cdots & U_{1,n} \\ \vdots & \ddots & \vdots \\ U_{n,1} & \cdots & U_{n,n} \end{bmatrix} \tag{2.2}$$

   $$\Gamma = \begin{bmatrix} \Gamma_1, & \Gamma_2, & ..., & \Gamma_m \end{bmatrix} \tag{2.3}$$

   b. Simpan vektor dari jumlah citra di *database* ke dalam *list image* (S).

   $$S = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \\ \cdots \\ \Gamma_M \end{bmatrix} = \begin{bmatrix} U_{1,1} & U_{1,2} & \cdots & U_{1,m} \\ U_{2,1} & U_{2,2} & \cdots & U_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ U_{M,1} & U_{M,2} & \cdots & U_{M,m} \end{bmatrix} \tag{2.4}$$

   c. Mencari matriks rata-rata (Ψ) dari semua piksel citra di *database* P_m dengan m merupakan jumlah piksel citra dan M adalah jumlah citra di *database*. Sehingga dihasilkan matriks vektor rata –rata dengan dimensi 1 $\times$ m.

   $$\Psi = \frac{1}{M} \sum_{m=1}^{M} P_m \quad ; \quad M = 1,2,...,M \tag{2.5}$$

   d. Menghitung selisih (Φ) antara citra di *database* Γ_m dengan nilai tengah

(Ψ). Sehingga diperoleh matriks selisih (Φ) berukuran M $\times$ m.

$$\Phi = \Gamma_M - \Psi \tag{2.6}$$

e. Kemudian tentukan matriks kovarian C dengan mengalikan selisih (Φ) dan matriks *transpose* (Φ^T). Matriks C yang diperoleh nanti berukuran M $\times$ M.

$$C = \Phi\Phi^T \tag{2.7}$$

f. Tentukan Nilai *Eigenvalue* (λ) dan *Eigenvector* (v) dari matriks kovarian C. (λ) yang diperoleh berukuran 1 $\times$ M, sedangkan V berukuran M $\times$ M.

$$\begin{aligned} Cv &= \lambda v \\ (C-\lambda I)v &= 0 \\ (\Phi\Phi^T - \lambda I)v &= 0 \end{aligned} \tag{2.8}$$

I adalah matriks identitas, λ adalah *Eigenvalue* dari C dan v adalah *Eigenvector* yang bersesuaian dengan λ.

g. Masukkan nilai *Eigenvector* (v) yang dipilih ke dalam matriks *Eigenface* (Ek) dengan mengalikan v yang berukuran M $\times$ M dengan matriks selisih (Φ) yang berukuran M $\times$ m dan simpan hasilnya dalam matriks yang berukuran M $\times$ m.

$$E_k = \sum_{k=1}^{M} V_k \Phi_k;\ k = 1,2,...,M \tag{2.9}$$

h. Hitung bobot dengan mengalikan matriks selisih (Φ) berukuran M $\times$ m dengan matriks *Eigenface transpose* dari (E_k^T) yang berukuran m $\times$ M.

$$\omega = E_k^T \times \Phi \tag{2.10}$$

i. Bobot yang diperoleh merupakan nilai ciri yang digunakan dalam proses pengenalan selanjutnya. Simpan bobot ini ke dalam matriks Ω berukuran M $\times$ N dengan N $<$ M.

$$\Omega = [\omega_1, \omega_2, ..., \omega_n] \tag{2.11}$$

2. Ekstraksi Fitur LDA

LDA bekerja berdasarkan analisis matriks penyebaran yang bertujuan menemukan suatu proyeksi optimal sehingga dapat memproyeksikan data *input*

pada ruang dengan dimensi yang lebih kecil dimana semua pola dapat dipisahkan semaksimal mungkin [3].

Berikut langkah-langkah pembentukan vektor fitur citra wajah menggunakan metode LDA:

a. Hasil dari bobot *Eigenface* (Ek) dijadikan sebagai *input* yang akan ditransformasikan ke dalam vektor kolom.

b. Menghitung rata – rata dalam kelas (m_i).

$$m_i = \frac{1}{n_i} \sum_{i=1}^{n} X_i \tag{2.12}$$

c. Menghitung rata – rata keseluruhan kelas (m).

$$m = \frac{1}{n_1 + ... + n_k} \tag{2.13}$$

d. Menghitung matriks sebaran antar kelas (*between class scatter matrix*, S_b) dan matriks sebaran dalam kelas (*within class scatter matrix*, S_w).

$$S_b = \sum_{i=1}^{k} n_i (m_i - m)(m_i - m)^T \tag{2.14}$$

$$S_w = \sum_{i=1}^{k} \sum_{j=1}^{n_i} (X_j - m_i)(X_j - m_i)^T \tag{2.15}$$

Dimana k merupakan jumlah seluruh kelas, n_i adalah jumlah sampel tiap kelas, dimana i adalah mewakili jumlah kelas dari seluruh kelas.

e. Memproyeksikan matriks sebaran (S_w dan S_b) ke dalam matriks proyeksi PCA (ω).

$$cov = ((\omega^T S_w \omega)^{-1} (\omega^T S_b \omega)) \tag{2.16}$$

f. Mencari *Eigenvalue* (λ) dan nilai *Eigenvector* (v) dari matriks sebaran.

$$S_b v = \lambda S_w v \tag{2.17}$$

g. Mengurutkan *Eigenvalue* (λ) sesuai dengan urutan nilai yang ada pada nilai eigen dari besar ke kecil. Selanjutnya proyeksi menggunakan k – 1 *Eigenvector* (v) (dimana k adalah jumlah kelas).

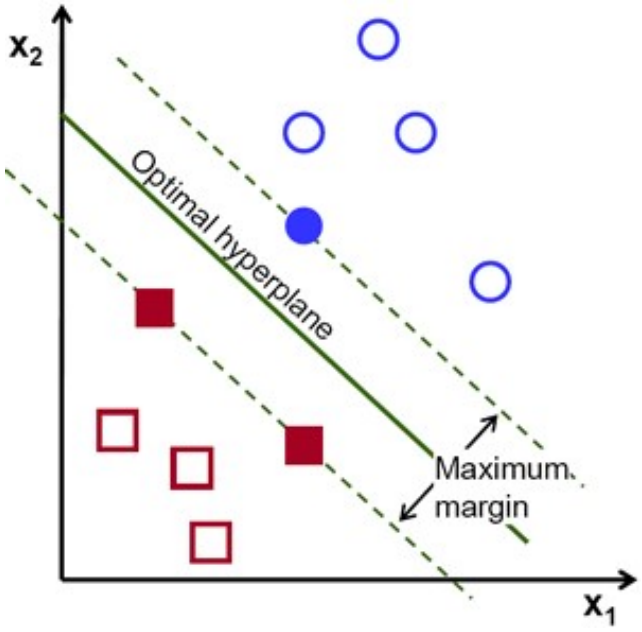
h. Memproyeksikan seluruh citra asal (bukan *centerend image*) ke *fisher* baris vektor dengan menghitung *dot product* dari citra asal v^T ke tiap –

tiap *fisher* baris vektor X^i.

$$M^x = v^T X^i \tag{2.18}$$

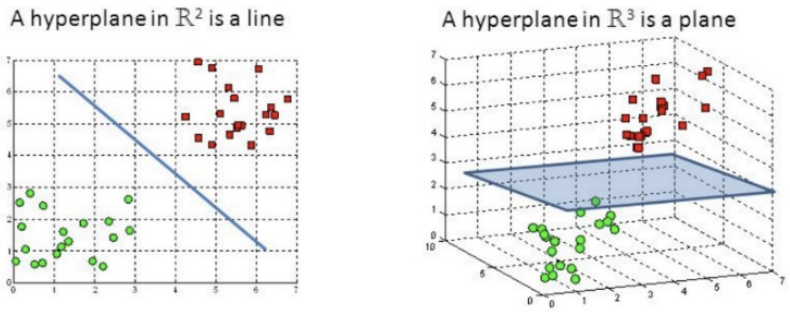
2.7 *Support Vector Machine* (SVM)

Support Vector Machine (SVM) adalah salah satu metode yang digunakan untuk klasifikasi. SVM dapat mengatasi masalah klasifikasi dengan *linear* maupun *non-linear*. SVM bekerja untuk mencari *hyperplane* atau fungsi pemisah untuk memisahkan dua buah kelas atau lebih pada ruang *input*, pada satu dimensi *hyperplane* disebut titik, pada dua dimensi disebut garis, pada tiga dimensi disebut bidang. Berikut ini adalah pemisahan data dengan metode *Support Vector Machine* (SVM) [30].



Gambar 2.7. Pemisahan Data Metode SVM

Dimensi *hyperplane* bergantung pada fitur dataset, apabila terdapat dua fitur, maka *hyperplane* akan berbentuk garis yang lurus, dan apabila terdapat tiga fitur, maka *hyperplane* akan berbentuk bidang dua dimensi. Berikut ini adalah contoh ilustrasi pada *hyperplane*.



Gambar 2.8. Ilustrasi Pada *Hyperplane*

SVM memisahkan data dalam dimensi-p dengan menggunakan bidang berdimensi p-1 (*hyperplane*). *Hyperplane* ini dipilih sedemikian rupa sehingga optimal dalam memaksimalkan *margin* dari data yang dikelompokkan. *Margin* di sini mengacu pada jarak minimum ke *hyperplane*.

Proses pembelajaran dalam SVM bertujuan untuk mendapatkan hipotesis terbaik berupa *hyperplane* pemisah. Tujuannya tidak hanya meminimalkan *empirical risk* yaitu rata-rata *error* pada data pelatihan, tetapi juga mempunyai generalisasi yang baik. Generalisasi ini mengacu pada kemampuan hipotesis untuk dengan benar mengklasifikasikan data yang tidak ada dalam data pelatihan. SVM bekerja berdasarkan prinsip *Structural Risk Minimization* (SRM) untuk memastikan generalisasi yang baik. Berikut merupakan tahapan dari algoritma SVM:

1. Fitur-fitur dari data diubah menjadi representasi dalam ruang dimensi yang lebih tinggi melalui penggunaan fungsi kernel seperti *linear, polynomial, radial basis function, dan sigmoid.*

2. Mencari *hyperplane* terbaik yang dapat memisahkan data yang telah diubah menjadi dimensi yang lebih tinggi.

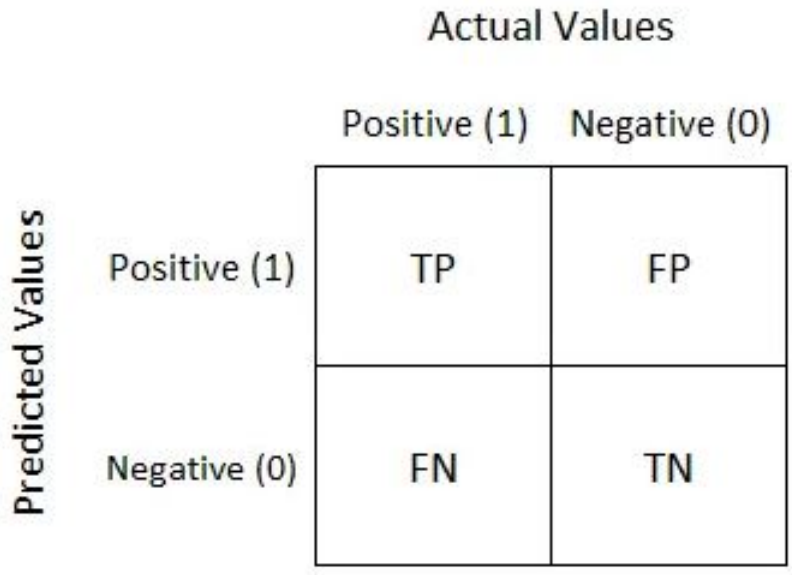
3. *Hyperplane* terbaik ditemukan dengan memperbesar jarak antara *hyperplane* dan titik data terdekat (*Support Vector*).

2.8 Python

Python adalah bahasa pemrograman yang digunakan dalam membuat aplikasi perintah komputer dan menganalisis data. Python merupakan bahasa pemrograman yang tidak ada batasan dalam penyalinannya atau mendistribusikannya (*freeware*). Python digunakan untuk membuat program dan menyelesaikan permasalahan-permasalahan sebagai *general purpose language*. Python termasuk bahasa pemrograman tingkat tinggi yang mudah untuk dipelajari [31].

2.9 *Confusion Matrix*

Pada prinsipnya, *confusion matrix* memberikan informasi mengenai perbandingan antara hasil klasifikasi yang dihasilkan oleh suatu sistem atau model dengan hasil klasifikasi yang sebenarnya. *confusion matrix*, yang diwujudkan dalam bentuk tabel, mencerminkan performa model klasifikasi pada sejumlah data uji dengan nilai sebenarnya yang sudah diketahui [32]. Gambar 2.9 merupakan gambar *confusion matrix* dengan 4 kombinasi *predicted values* dan *actual values* yang berbeda:



Gambar 2.9. *Confusion Matrix*

Berikut adalah empat istilah yang mewakili hasil dari proses klasifikasi dalam *confusion matrix*:

1. *True Positive* (TP), yakni situasi di mana data positif diprediksi dengan benar.
2. *False Positive* (FP), merujuk kepada kasus ketika data negatif diprediksi sebagai data positif.
3. *True Negative* (TN), menggambarkan ketika data negatif diprediksi secara tepat.
4. *False Negative* (FN), mencakup situasi di mana data positif tetapi diprediksi sebagai data negatif.

Berdasarkan istilah-istilah yang disebutkan, dapat dirumuskan persamaan *confusion matrix* guna menghitung akurasi, presisi, dan recall seperti berikut:

$$\text{akurasi} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.19}$$

$$\text{presisi} = \frac{TP}{TP+FP} \tag{2.20}$$

$$\text{recall} = \frac{TP}{TP+FN} \tag{2.21}$$