BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Perkembangan teknologi merupakan suatu keniscayaan yang tidak dapat kita hindari. Membawa berbagai dampak, baik dampak positif maupun negatif. Namun, merebaknya perkembangan teknologi yang kian *massive* justru sesuatu yang patut kita syukuri sebagai suatu anugerah, sebab hal itu menandakan kemajuan peradaban umat manusia dalam menciptakan berbagai alat yang bertujuan untuk memudahkan pekerjaan kita sehari-hari. Tak dapat dipungkiri, perkembangan teknologi yang cukup pesat dewasa ini, menawarkan berbagai kemudahan, membuat sesuatu yang mustahil menjadi mungkin, bahkan hal-hal yang tidak pernah kita bayangkan sekali pun dapat terlaksana berkat kemajuan tersebut.

Tren teknologi saat ini ditengarai lebih mengarah kepada kecerdasan buatan atau yang populer dengan istilah *artificial intelligence* (AI). Istilah dan riset mengenai kecerdasan buatan dikenal sejak tahun 1956 dan semakin populer dewasa ini berkat algoritma canggih, penyimpanan komputasi, dan peningkatan daya. Teknologi ini bahkan diklaim mampu menggantikan beberapa pekerjaan atau profesi di masa depan. Betapa tidak, AI memungkinkan mesin untuk mengerjakan tugas layaknya manusia, sebab konsep utama dari AI sendiri ialah *teaching machine like human* (mengajari mesin untuk bisa berperilaku seperti manusia) (Dwivedi dkk., 2021).

Cabang dari AI yang saat ini banyak dipelajari yakni pembelajaran mendalam atau *deep learning.* Terobosan *deep learning* membuat AI makin populer karena implementasinya banyak digunakan hampir di semua lini industri (Raschka dkk., 2020). *Deep learning* mampu melatih komputer untuk mengerjakan pekerjaan manusia, seperti mengenali suara, mengidentifikasi gambar, bahkan membuat prediksi. Adalah *face recognition* yang merupakan salah satu sistem implementasi deep learning yang dapat mengenal wajah secara spesifik dari gambar

digital atau *frame* video. Tidak hanya wajah yang dapat dikenali, namun juga usia, keadaan emosional, fitur-fitur wajah, dan sebagainya (França dkk., 2021).

Sistem *face recognition* ini dibangun berdasarkan pemodelan (*pre-trained model*). Ketika membuat sistem pengenalan seperti ini, konsep *machine learning* mengharuskan kita membuat pemodelan terlebih dahulu sebelum menjadi sistem yang utuh (Zulfiqar dkk., 2019). Dalam kasus *face recognition*, maka pemodelan yang dibuat adalah pemodelan wajah. Sederhananya, mengajari komputer agar mengenali gambar dan karakteristik wajah untuk kemudian diintegrasikan dengan sistem *database* yang memuat berbagai data pemilik wajah. Wajah yang terdeteksi kemudian muncul berdasarkan pada sebuah ukuran dan orientasi dalam posisi tertentu.

Implementasi sistem *face recognition* dapat digunakan dalam ragam aplikasi, misalnya untuk sistem absensi, me-*monitoring* sistem pengunjung di pusat perbelanjaan dan di tempat-tempat keramaian, hingga mengenali statistik perilaku konsumen bagi industri retail. Sistem ini dapat menggantikan kartu dan sidik jari untuk aplikasi absensi dan *gate system* karena dinilai lebih canggih dan praktis dalam segi penerapannya (Van Natta dkk., 2020).

Terdapat beberapa penelitian yang menjelaskan tentang metode pengenalan wajah untuk melakukan absensi, salah satunya menggunakan metode Viola – Jones yang menjelaskan bahwa metode tersebut dapat mengolah dan mengenali wajah dengan cukup baik (Viola dan Jones, 2018).

Penelitian ini menghasilkan tiga kontribusi utama. Yang pertama gambar representasi atau gambar baru atau di sebut dengan "Gambar Integral", yang dapat terfungsi sebagai dektektor untuk pengolahan dengan sangat cepat. Yang kedua *classifier* sederhana dan efisien yang di bangun oleh AdaBoost untuk memilih sejumlah fitur visual kritis yang kecil dari satu set fitur potensial yang besar. Kontribusi yang ketiga adalah penggabungan pengklasifikasian dengan "*cascade*" yang dapat menghapus latar belakang atau yang bukan wajah dengan komputasi yang sangat cepat dalam menentukan daerah wajah. Sistem deteksi wajah ini menghasilkan kinerja sebanding dengan sistem-sistem sebelumnya dan

diimplementasikan pada desktop konvensional dengan hasil deteksi wajah yaitu 15 *frame* per detik.

Nugroho merancang pendektesian wajah dengan sistem saraf tiruan. Sistem di berikan beberapa contoh atau sempel untuk pembelajaran sistem. Algoritma *quickprop* dan metode *active learning* digunakan untuk mempercepat proses pelatihan sistem. Dari hasil eksperimen dengan menggunakan 23 file citra berisi 149 wajah, sistem pendeteksi wajah ini memberikan hasil detection rate 71,14% dan *false positive* (Nugroho, 2004).

Penelitian yang di lakukan oleh Prayogi membahas tentang pengolahan dektesi wajah pada gambar yang bergerak atau video yang tertangkap oleh kamera. Kemudian di proses untuk mendektesi yang mana wajah yang mana latar belakang. Sehingga menghasilkan citra wajah dengan warna kulit yang di tampilkan sedangkan yang tidak dihitamkan. Dan akan di simpan dalam 5 kelas yang di sebut *euclidean distance* (Prayogi dkk., 2007).

Yusron juga melakukan penelitian dengan citra objek bergerak. Pengujian sistem ini menghasilkan keakuratan 65% dalam proses rata – rata empat detik. Itensitas cahaya sangat berpengaruh dalam proses metode ini. Akibatnya, nilai ambang pada suatu kondisi pencahayaan dengan kondisi pencahayaan yang lain bisa jadi berbeda (Rijal dan Ariefianto, 2008).

Rompas dkk melakukan penelitian absensi berbasis pengenalan wajah *multiple person* yang bertujuan untuk menghilangkan beberapa kekurangan dari absensi manual dengan cara melakukan absen secara otomatis dengan pengenanlan wajah (Rompas dkk., 2021). Penelitian ini menggunakan *Haarcascade*, Metode CNN dan *Support Vector Machine* untuk dapat mendeteksi wajah dan mengenali wajah yang telah ter*input* oleh *user*.

Penelitian yang dilakukan oleh (Putranto dkk., 2017) melakukan peneliti ini menggunakan *Naive Bayes* untuk mengklasifikasikan hasil ekstraksi ciri *eigenface* untuk memprediksi wajah. Normalisasi *z-score* ditambahkan untuk mempertajam akurasi. Untuk melihat performansi metode yang diusulkan, 200 dataset dibagi menjadi data training dan testing dengan menggunakan *cross validation* (k=10). Penelitian lain yang dilakukan oleh (Sutarti dkk., 2019) melakukan penelitian untuk

membandingan Akurasi PCA dan 2PCA dengan Klasifikasi K *Nearest Neighbor* Pada Citra Wajah.

Berdasarkan penelitian yang telah dilakukan sebelumnya, maka penelitian ini melanjutkan dan mengembangkan dari peneltian yang telah dilakukan yaitu merancang suatu sistem deteksi wajah pada suatu gambar dengan metode *eigenface* dan svm.

2.2 Dasar Teori

Sistem berasal dari bahasa Latin *(systēma*) dan bahasa Yunani (*sustēma*) adalah suatu kesatuan yang terdiri komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi atau energi untuk mencapai suatu tujuan. Istilah ini sering dipergunakan untuk menggambarkan suatu set entitas yang berinteraksi, dimana suatu model matematika seringkali bisa dibuat.

Sedangkan kehadiran menurut KBBI perihal hadir, yaitu adanya (seseorang, sekumpulan orang) pada suatu tempat. Berdasarkan definisi sistem dan kehadiran, dapat disimpulkan bahwa Sistem kehadiran adalah sistem yang digunakan dalam pengambilan data guna mengetahui jumlah kehadiran pada suatu acara. Setiap kegiatan yang membutuhkan informasi mengenai peserta tentu dilakukan konfirmasi jika telah hadir. Hal ini juga terjadi pada proses kegiatan pembelajaran santri. Sistem kehadiran diperlukan sebagai salah satu bahan evaluasi kepada guru dan wali santri terhadap pembelajaran dan pembuatan tolak ukur ke depan guna pembelajaran yang lebih baik.

2.2.1 *Face Recognition*

Face recognition atau pengenalan wajah adalah salah satu teknologi biometrik yang telah banyak diaplikasikan dalam sistem keamanan selain pengenalan retina mata, pengenalan sidik jari dan iris mata (Bah dan Ming, 2020), (Harikrishnan dkk., 2019). *Face recognition* juga bagian dari aplikasi komputer vision yang mampu mengidentifikasi atau memverifikasi seseorang dari gambar digital atau frame video. Salah satu cara untuk melakukan ini adalah dengan membandingkan fitur wajah dari gambar dengan wajah yang tersimpan dalam *database* (Saputra dkk., 2013). *Face recognition* umumnya melibatkan dua tahap.

Tahap pertama adalah *face detection*, dimana foto yang dicari untuk menemukan wajah apapun. Tahap kedua adalah *face recognition,* dimana wajah dideteksi dan diproses, dan dibandingkan dengan database wajah yang dikenal untuk menentukan siapa orang tersebut (Annubaha dkk., 2022).

Prinsip dasar pengenalan wajah adalah mengutip informasi unik wajah, kemudian di-*encode* dan dibandingkan dengan hasil *decode* yang sebelumnya dilakukan. Pada metode *eigenface*, *decoding* dilakukan dengan menghitung *eigen vector* kemudian direpresentasikan dalam sebuah matriks. *Eigen vector* juga dinyatakan sebagai karakteristik wajah oleh karena itu metode ini disebut dengan *eigenface*. Setiap wajah direpresentasikan dalam kombinasi linear *eigenface*(Annubaha dkk., 2022).

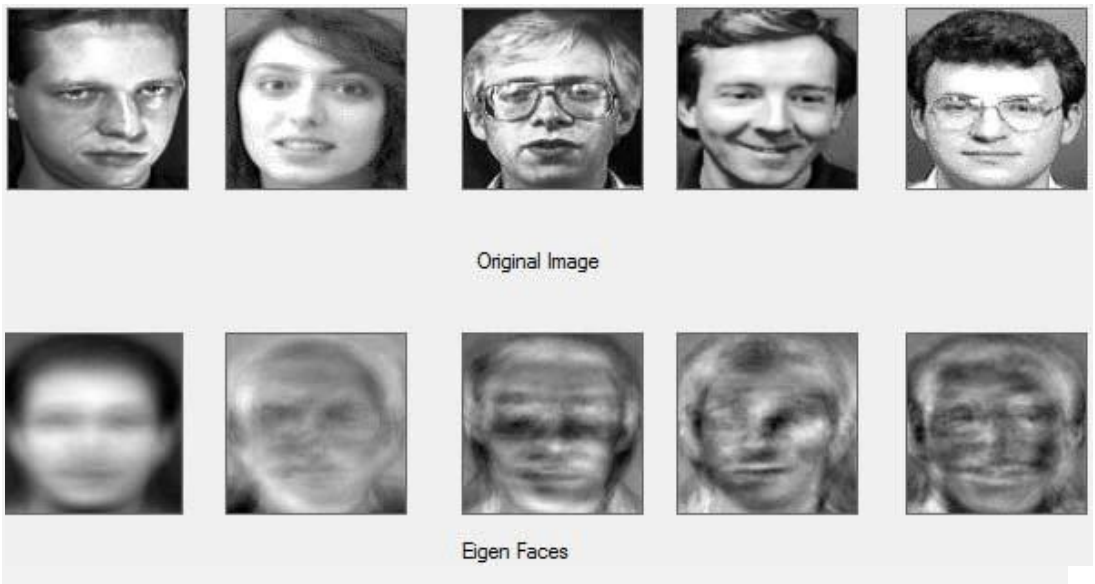
2.2.2 *Eigenface*

Eigenface merupakan sebuah nama pada sekumpulan *eigenvectors* yang digunakan pada *computer vision* untuk kebutuhan *face recognition. Eigenvectors* diturunkan dari *covariance matrix* dari *probability distribution* pada *high-dimensional vector space* gambar wajah. Eigenfaces sendiri membentuk basis set semua gambar yang digunakan untuk membangun *covariance matrix* (Kshirsagar dkk., 2011). Ini akan mereduksi dimensi gambar wajah menjadi kumpulan gambar yang lebih kecil yang dapat mewakili gambar aslinya pada training dataset (Slavkovic dan Jevtic, 2012). Kata *eigenface* sebenarnya berasal dari bahasa Jerman *"eigenwert"* dimana *"eigen"* artinya karakteristik dan *"wert"* artinya nilai. *Eigenface* adalah salah satu algoritma pengenalan pola wajah yang berdasarkan pada *Principle Component Analysis (PCA)* yang dikembangkan di MIT (Alam, dkk, 2015). *Eigenface* adalah sekumpulan *standardize face ingredient* yang diambil dari analisis statistik dari banyak gambar wajah. Teknik ini telah digunakan pada pengenalan tulisan tangan, pembacaan bibir, pengenalan suara dan pencitraan medis (Al Fatta, 2007).

Metode *eigenface* adalah bagaimana cara meguraikan informasi yang relevan dari sebuah citra wajah, kemudian mengubahnya ke dalam satu set kode yang paling efisien dan membandingkan kode wajah tersebut dengan *database*

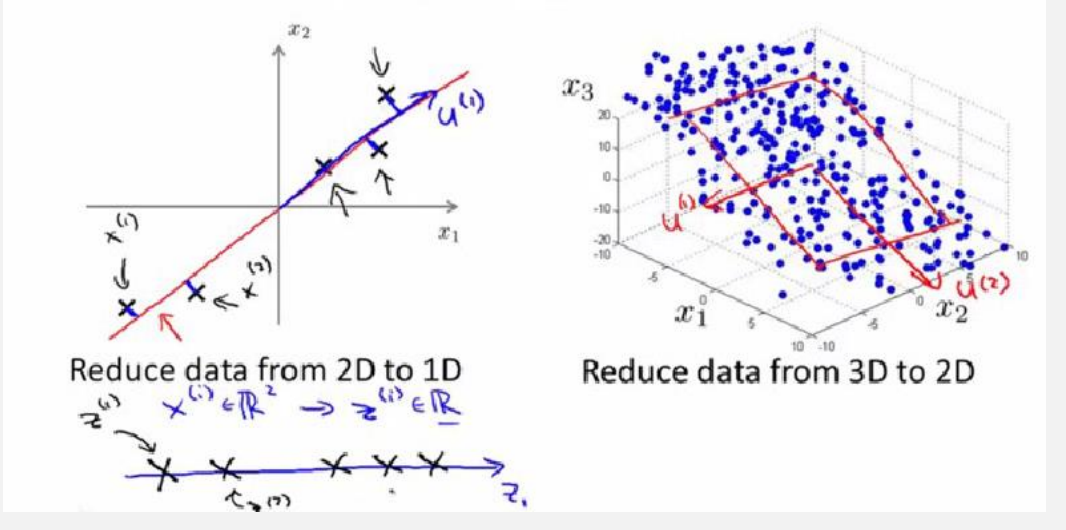
berisi beragam wajah yang telah dikodekan secara serupa. *Eigenfaces* PCA digunakan untuk mereduksi dimensi sekumpulan atau ruang gambar sehingga basis atau sistem koordinat yang baru dapat menggambarkan model yang khas dari kumpulan tersebut dengan lebih baik. Model yang diinginkan merupakan sekumpulanwajah yang dilatihkan. Fitur yang baru tersebut dibentuk melalui kombinasi linear. Komponen fitur ruang karakter ini tidak saling berkolerasi dan memaksimalkan perbedaan yang ada pada variabel aslinya. Secara garis besar langkah-langkah metode PCA adalah sebagai berikut:

- ✓ Cari matrik *u*
- ✓ Cari matrik *covariance*: $C = U^T \times U$
- ✓ Cari *eigen values* (*ë*) dan *eigen vector* (*V*) dari matrik C
- ✓ Cari matrik *eigenface* Matrik *eigenface* dapat digunakan untuk pengenalan citra.



Gambar 2. 1 *Original face* dan *Eigenface*

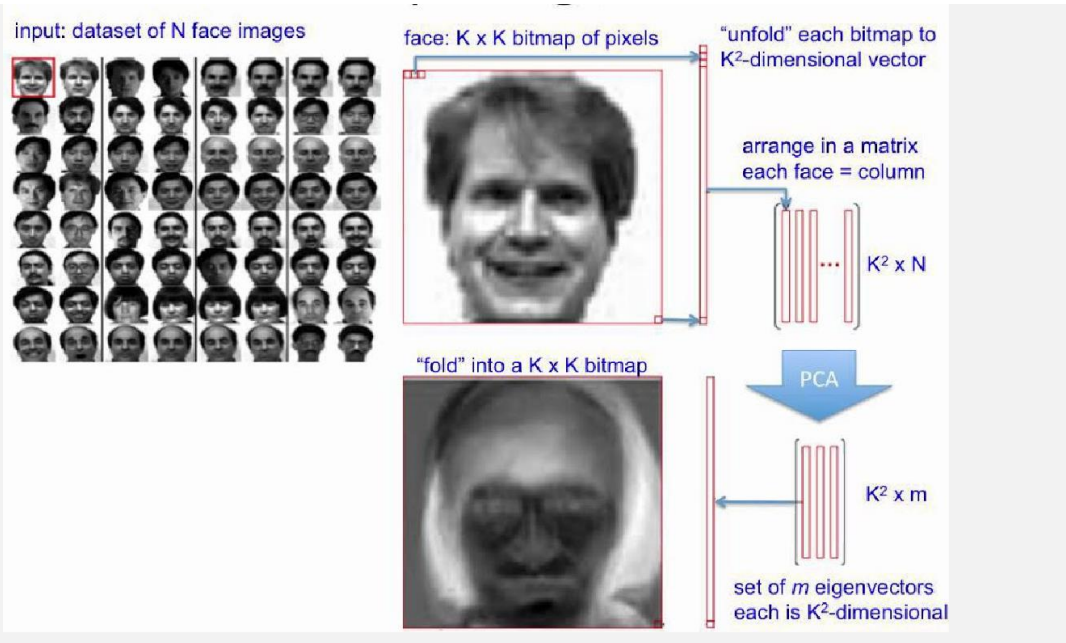
Pada praktiknya untuk mendapatkan *Eigenface* digunakan teknik seperti PCA *(Principal Component Analysis)* (Saputra dkk., 2013). PCA merupakan teknik *dimentionality reduction* yang dapat menghasilkan *output* data dengan dimensi rendah dari *input* data berdimensi tinggi seperti gambar wajah.



Gambar 2. 2 Ilustrasi PCA dalam mereduksi dimensi *vector* data

PCA dihitung dalam dua langkah sebagai berikut.

1. Menghitung *covariance matrix* dari input gambar (*original image*)
2. Melakukan *eigenvalue decomposition* pada *covariance matrix* yang didapatkan



Gambar 2. 3 Ilustrasi *Eigenface* menggunakan PCA

Sebelum di terapkan pada PCA, untuk dataset sejumlah N *sample* gambar wajah dengan dimesi K x K yang kemudian dilakukan froses *unfolding* menjadi *vector* 1D dengan ukuran K^2 x N. Tiap kolom pada matrix merupakan gambar wajah

1D. Setelah itu, dihasilkan *eigenvector* sebanyak *m* kolom (dimensi lebih rendah), dengan ukuran K^2 x m. *Eigenvector* tersebut digunakan untuk proses *folding* menjadi *Eigenface Image* dengan dimensi 1 x K x K.

2.2.3 Cara Kerja *Eigenface* dengan PCA/ *Karhunen-Loeve* (KL)

Pengenalan wajah merupakan suatu pengenalan pola *(pattern recognition)* yang khusus untuk kasus wajah. Ini dapat dideskripsikan sebagai pengklasifikasian suatu wajah apakah dikenali *(known)* atau tidak dikenali *(unknown)*, dimana setelah dibandingkan kemudian disimpan secara tersendiri. Beberapa pendekatan untuk pengenalan obyek didasarkan secara langsung pada citra-citra tanpa penggunaan model 3D dan Jaringan Syaraf Tiruan.

Secara umum sistem dibagi menjadi dua jenis, yaitu *sistem featurebased* dan *image based* (Artotomo dkk., 2019). Pada sistem yang pertama digunakan fitur yang diekstraksi dari komponen citra wajah (mata, hidung, mulut, dll.) yang kemudian hubungan antara fitur-fitur tersebut dimodelkan secara geometris. Sedangkan sistem kedua menggunakan informasi mentah dari piksel citra (kamera) yang kemudian direpresentasikan dalam metode tertentu misalnya *principal component analysis* (PCA), transformasi *wavelet* yang kemudian digunakan untuk klasifikasi identitas citra (Silaban & Sunandar, 2018). Pendekatan *image based* kebanyakan digunakan untuk pengenalan wajah. Pada metode ini, model wajah dipelajari melalui proses *training* dengan menggunakan satu set data pelatihan yang berisi contoh-contoh wajah. Kemudian hasil *training* ini digunakan untuk pengenalan wajah. Yang termasuk dalam metode ini adalah *eigenface*.

Proyeksi ruang *eigen (eigenspace)* dikenal juga sebagai *Karhunen-Loeve* (KL) atau juga dinamakan dengan *Principal Component Analysis* (PCA). Algoritma *eigenface* memanfaatkan *Principal Component Analysis* (PCA) untuk mereduksi dimensinya guna menemukan vektor- vektor yang mempunyai nilai terbaik untuk distribusi citra wajah didalam ruang citra masukan.

Ide utama PCA adalah menemukan vektor dengan nilai terbaik untuk distribusi citra wajah dalam seluruh ruang citra. Vektor-vektor ini mendefinisikan subruang citra wajah atau biasa disebut dengan nama ruang wajah (Turk & Pentland, 1991). Vektor ini mendefinisikan subruang dari citra-citra wajah dan subruang tersebut dinamakan ruang wajah. Semua wajah-wajah dalam himpunan pelatihan diproyeksikan ke dalam ruang wajah untuk menemukan suatu himpunan bobot-bobot yang mendeskripsikan kontribusi dari tiap vektor dalam ruang wajah. Untuk identifikasi suatu citra uji, membutuhkan proyeksi suatu citra ke dalam ruang wajah untuk menentukan korespondensi kumpulan bobot-bobot. Dengan membandingkan kumpulan bobot-bobot wajah dalam training set, Pengujian citra dapat diidentifikasi. Prosedur kunci dalam PCA didasarkan pada tranformasi *Karhumen-Loeve*.

Bentuk umum dari *Principal Component Analysis* (PCA) dapat dilihat berikut ini:

$$C = \sum_{k=1}^{K} (x_k - \Psi)(x_k - \Psi)^T \quad (2.1)$$

$$C = \Phi . \Phi^T \quad (2.2)$$

Keterangan :

C = matriks kovarian

x = *image* (x_1, x_2, …, x_k)

Ψ = rata-rata *image* yang dihasilkan dari merata-rata x (x_1, x_2, …, x_k)

Φ = selisih antara *image* (x) dengan nilai tengah (Ψ).

Dimana C merupakan matriks kovarian, x merupakan *image* (x_1, x_2,… , x_k) dan Ψ adalah rata-rata *image* yang dihasilkan dari merata-rata x (x_1, x_2, … , x_k). Dengan dekomposisi *eigen*, matriks kovarian ini dapat didekomposisi menjadi:

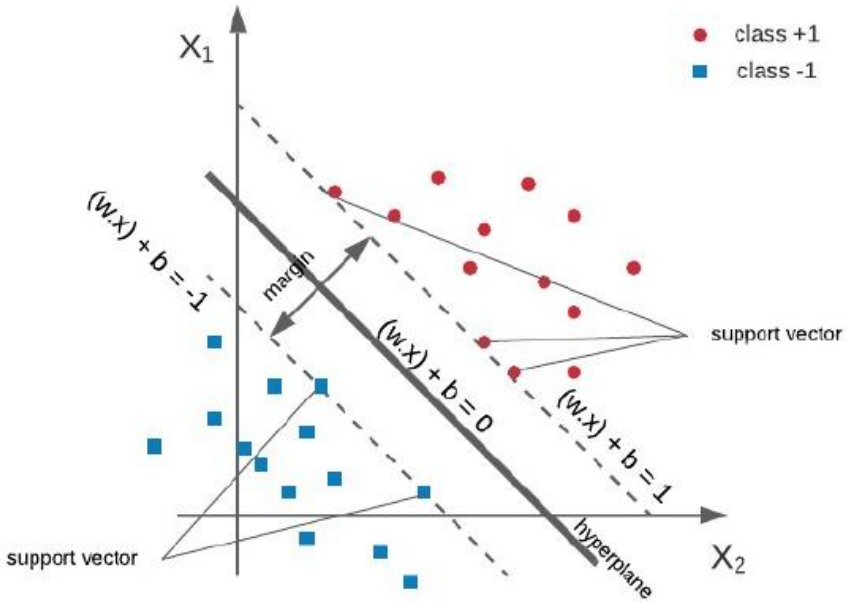
Dimana Φ adalah selisih antara *image* (x) dengan nilai tengah (Ψ).Pilih sejumlah kolom dari matriks Φ yang berasosiasi dengan *eigenvalue* terbesar. Pemilihan sejumlah m kolom dari matriks Φ ini menghasilkan matriks transformasi atau matriks proyeksi Φ_m. Berikutnya sebuah *imagex* (berdimensi

n) dapat diekstraksi kedalam *feature* baru *y* (berdimensi *m* < *n*) dengan memproyeksikan *x* searah dengan Φ_m sebagai berikut:

Dengan kata lain metode PCA memproyeksikan ruang asal kedalam ruang baru yang berdimensi lebih rendah, yang mana sebanyak mungkin kandungan informasi asal tetap dipertahankan untuk tidak terlalu banyak hilang setelah dibawa ke dimensi *feature* yang lebih kecil. Disini terlihat reduksi *feature* yang signifikan dari *n* buah menjadi *m* buah yang tentunya akan sangat meringankan komputasi dalam proses pengenalan berikutnya.

2.2.4 Algoritma *Support Vector Machine (SVM)*

Algoritme *Support Vector Machine* (SVM) ditemukan oleh Vladimir N. Vapnik dan *soft margin* diusulakn oleh Corinna Cortes 2 dan Vapnik pada tahun 1993 (Ghosh dkk., 2019). Konsep dasar SVM merupakan kombinasi dari teori-teori komputasi yang telah ada puluhan tahun sebelumnya seperti *margin hyperplane*. *Support Vector Machine* membangun *hyperplane* yang dapat digunakan untuk klasifikasi, regresi, atau tugas lainnya (Chandra dan Bedi, 2021).



Gambar 2. 4 *Support Vector Machine* dengan *hyperplane*

Konsep SVM secara sederhana adalah mencari *hyperplane* terbaik yang berfungsi sebagai pemisah antara dua *class*. Gambar 2.4 memperhatikan beberapa data objek yang merupakan anggota dari dua *class*; *class* $+1$ dan *class* -1. *Class* $+1$ objek berbentuk bulat dan warna merah, *class* -1 objek berbentuk kotak dan berwarna biru. *Margin* adalah jarak antara *hyperplane* dengan objek terdekat dari masing-masing *class*. Objek data paling dekat disebut *support vector*, dan garis solid sebagai *hyperplane* terbaik, yaitu terletak tepat pada tengah-tengah kedua *class* (F.Y dkk., 2017).

Data yang dinotasikan sebagai $x_i \in \Re^d$ sedangkan label masing-masing dinotasikan $y_i \in \{-1, +1\}$ untuk $i = 1, 2, \ldots l$, dan l adalah banyaknya data. Kedua *class* yaitu: *class* -1 dan *class* $+1$ dapat terpisah secara sempurna oleh *hyperplane* berdimensi d, yang didefinisikan pada persamaan 2.3. Objek data x_i yang termasuk *class* -1 dan objek data x_i yang termasuk *class* $+1$ dapat dirumuskan sebagai objek data yang memenuhi persamaan 2.4

$$w.x + b = 0 \qquad \ldots\ldots\ldots \qquad (2.3)$$

Keterangan :
w = vektor bobot
φ(x) = fungsi yang memetakan x dalam suatu dimensi
b = bias

$$class(x_i) = \begin{cases} C_{-1}, & w.x + b \leq -1 \\ C_{+1}, & w.x + b \geq +1 \end{cases} \qquad \ldots\ldots\ldots \qquad (2.4)$$

Keterangan :
C = konstanta c
w = vektor bobot
φ(x) = fungsi yang memetakan x dalam suatu dimensi
b = bias

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara *hyperplane* dan titik terdekatnya, yaitu $\frac{2}{\|w\|^2}$ mencari *hyperplane* dapat digunakan metode *Quadratic Pragramming* (QP) *problem*, yaittu mencari titik menimal pada persamaan 2.5, dengan memperhatika *constraint* pada persamaan 2.6. Solusi untuk

optimasi diselesaikan dengan berbagai teknik komputasi, diantaranya fungsi *Lagrange Multiplier* pada persamaan 2.7.

$$\min_{\vec{w}} \tau(w) = \frac{1}{2}\|w\|^2 \quad \ldots\ldots\ldots \quad (2.5)$$

Keterangan :
$\min_{\vec{w}}$ = Minimum vektor satuan W
w = vektor bobot

$$y_i(x_i . w + b) - 1 \geq 0, \forall_i \quad \ldots\ldots\ldots \quad (2.6)$$

Keterangan :
y_i = kelas data ke-i
x_i = data ke-i
w = vektor bobot
b = bias
$\forall_i$ = semua i

$$L(\vec{w}, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{l} \alpha_i(y_i((\vec{x}_i . \vec{w} + b) - 1)) \quad \ldots\ldots\ldots \quad (2.7)$$
$$(i = 1, 2, \ldots l)$$

Keterangan :
L = *loss function*
$\vec{w}$ = vektor satuan w
y_i = kelas data ke-i
$\vec{x}_i$ = vektor satuan data ke-i
w = vektor bobot
b = bias

Nilai optimal dapat dihitung dengan memaksimalkan L terhadap α_i dan meminimalkan L terhadap w dan b. berdasarkan asumsi bahwa kedua belas *class* dapat terpisaj secara sempurna oleh *hyperplane*. Akan tetapi, pada umumnya kedua buah *class* tidak dapat terpisah secara sempurna sehingga menyebabkan proses optimasi tidak dapat diselesaikan, karena tidak ada w dan b yang memenuhi pertidaksamaan 2.6. Untuk itu pertidaksamaan 2.6 dimodifikasi dengan memasukkan *slack variable* $\xi_i(\xi_i \geq 0)$, menjadi persamaan 2.8. Demikian juga dengan persamaan 2.5, sehingga diperoleh persamaan 2.9. C adalah parameter

penalty yang mengendalikan *tradeoff* antara maksimal margin dan minimalisasi *classification error*.

$$y_i(x_i.w + b) \geq 1 - \xi_i, \forall_i \quad \text{……..} \quad (2.8)$$

Keterangan :
y_i = kelas data ke-i
x_i = data ke-i
w = vektor bobot
b = bias
ξ_i = ksi-i
$\forall_i$ = semua i

$$\min_{\vec{w}} \tau(w) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \quad \text{……..} \quad (2.9)$$

Keterangan :
$\min_{\vec{w}}$ = Minimum vektor satuan W
w = vektor bobot
C = parameter c
ξ_i = ksi-i

2.2.5 *Non-Linear Classification* (Klasifikasi non-linear)

Pada umumnya masalah yang terjadi jarang bersifat linear *separable* dan kebanyakan bersifat non-linear. Untuk mengatasi masalah non-linear, SVM dimodifikasi dengan memasukkan fungsi kernel. Pemilihan fungsi kernel yang tepat adalah hal yang sangat penting. Karena fungi ini akan menentukan *feature space* dimana fungsi *classifier* akan di cari. Menurut Karatzoglou dan Smola ada beberapa fungsi kernel yang sering digunakan dalam literatur SVM, ditunjukkan pada Tabel 2.1 (Karatzoglou dkk., 2004).

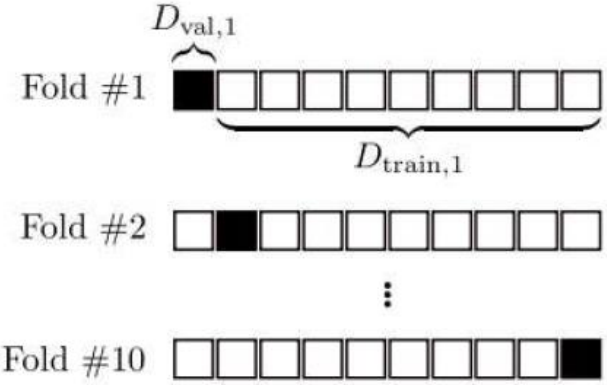
Tabel 2.1. Fungsi kernel

Fungsi kernel	Formula
Linear	$K(x_i, x_j) = {x_i}^T x_j$
Sigmoid	$K(x_i, x_j) = \tanh(\gamma {x_i}^T x_j + r)$
RBF	$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2, \gamma > 0)$
Polynomial	$K(x_i, x_j) = (\gamma {x_i}^T x_j + r)^d, \gamma > 0$

2.2.6 *K-Folds cross validation*

Cross Validation (CV) adalah metode statistik yang paling banyak digunakan untuk mengestimasi kesalahan prediksi dari model dan menyesuaikan parameter model dimana data dipisahkan menjadi dua *dataset* yaitu *dataset training* dan *dataset validation*, kemudian model dilatih oleh *dataset training* dan evaluasi kinerja diuji dengan *dataset validation*. Proses ini berulang hingga setiap K-Fold berfungsi sebagai alat validasi. CV seperti metode *subsampling* acak yang diulang. Tetapi pengambilan sampel dilakukan sedemikian rupa sehingga tidak ada dua set *validation* yang tumpang tindih.

Gambar 2.5 mengilustrasikan proses ini untuk k = 10, yaitu validasi silang 10 kali lipat. Di *fold* pertama, *dataset* pertama berfungsi sebagai *dataset validation* $(D_{val,1})$ dan sembilan *dataset* lainnya berfungsi sebagai *dataset training* $(D_{train,1})$. Di *fold* kedua, *dataset* kedua sebagai *dataset validation* dan *dataset* sisanya adalah *dataset training* dan seterusnya.



Gambar 2. 5 *Cross validation* 10-*fold, dataset* secara acak dibagi menjadi sepuluh *dataset* terpisah. Masing-masing berisi kira-kira 10% data. Model dilatih dengan *dataset training* dan kemudian diterapkan ke *dataset validation*.

2.2.7 Evaluasi Kinerja Model

2.2.7.1 *Confusion matrix*

Confusion matrix digunakan untuk menghitung performa dari algoritma klasifikasi. Nilai ditampilkan di setiap baris untuk menunjukkan jumlah data yang terkandung di dalam kelas. *Confusion Matrix* merupakan pengukuran performa buat

permasalahan klasifikasi *machine learning* dimana luaran bisa berbentuk 2 kelas ataupun lebih. Pada dasarnya *confusion matrix* membagikan data perbandingan hasil klasifikasi yang dicoba oleh sistem dengan hasil klasifikasi sesungguhnya.

Untuk mengukur klasifikasi biner menggunakan empat parameter untuk memprediksi yaitu: *True Positif* (TP), *True Negatif* (TN), *False Positif* (FP) dan *False Negatif* (FN). Performa yang dihitung adalah akurasi, *recall*, *precision* dan F1-*score*. Untuk klasifikasi biner menggunakan rumus standar, untuk klasifikasi *multiclass* menghitung performa akurasi, *recall*, *precision* dan F1-*score* menggunakan perhitungan yang diusulkan (Sokolova & Lapalme, 2009) masing-masing seperti pada persamaan 2.10, 2.11, 2.12, 2.13.

$$Recall = \frac{\sum_{i=1}^{l} TP_i}{(TP_i + FN_i)} \quad \ldots\ldots\ldots \quad (2.10)$$

$$Precision = \frac{\sum_{i=1}^{l} TP_i}{(TP_i + FP_i)} \quad \ldots\ldots\ldots \quad (2.11)$$

$$F1 - score = \frac{Precision \times Recall}{Precision + Recall} \quad \ldots\ldots\ldots \quad (2.12)$$

$$Average\ Accuracy = \frac{\sum_{i=1}^{l} \frac{TP_i + TN_i}{TP_i + FN_i + FP_i + TN_i}}{l} \quad \ldots\ldots\ldots \quad (2.13)$$

Keterangan :

TP = *True Positif* (TP), jumlah dokumen dari kelas 1 yang benar diklasifikasikan sebagai kelas 1.

TN = *True Negatif* (TN), jumlah dokumen dari kelas 0 yang benar diklasifikasikan sebagai kelas 0.

FP = *False Positif* (FP), jumlah dokumen dari kelas 0 yang salah diklasifikasikan sebagai kelas 1.

FN = *False Negatif* (FN) jumlah dokumen dari kelas 1 yang salah diklasifikasikan sebagai kelas 0.

Recall = rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif

precision = rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif

f1-score = perbandingan rata-rata presisi dan recall yang dibobotkan

Average Accuracy = rasio prediksi benar (positif dan negatif) dengan keseluruhan data