BAB 2

LANDASAN TEORI

2.1 Computer Vision

Visi komputer merupakan gabungan dari pengolahan citra dan pengenalan pola, di mana hasil akhir dari visi komputer adalah pemahaman suatu citra. Pengembangan visi komputer bergantung dari teknologi sistem komputer, baik itu sebagai peningkatan citra atau pengenalan citra. Visi komputer berkerja dengan mengunakan algoritma dan sensor optik untuk mengsimulasikan penglihatan manusia agar secara otomatis mendapatkan informasi penting dari suatu objek, yang kemudian digabungkan dengan analisis citra. Di mana tahapan analisis citra adalah; Informasi citra, di mana citra suatu objek disimpan kedalam komputer. Pengolahan citra, di mana kualitas suatu citra ditingkatkan untuk meningkatkan detil suatu citra. Segmentasi citra, di mana suatu citra objek disegmentasi dan dipisahkan dari latar belakang. Perhitungan citra, di mana beberapa fitur yang penting dihitung. Intepretasi citra, di mana citra yang sudah diekstraksi akan diintepretasi. (Wiley & Lucas, 2018)

2.1.1. Pattern Recognition

Pengenalan pola sebagai salah satu cabang visi komputer, berfokus pada proses indentifikasi suatu objek melalui transformasi citra agar mendapatkan kualitas dan intepretasi citra lebih baik. Proses ini bertujuan untuk mengekstraksi informasi untuk membuat keputusan berdasarkan citra yang di dapatkan, atau dengan kata lain bertujuan membuat suatu mesin yang dapat melihat. (Wiley & Lucas, 2018)

2.1.2. Image Processing

Pengolahan citra pada dasarnya meningkatkan intepretasi atau persepsi informasi di suatu citra untuk pengamat manusia dan memberikan input yang ideal kepada metode pengolahan citra lainnya. Secara prinsip, tujuan dari peningkatan citra adalah untuk mengubah atribut dari suatu citra agar lebih ideal untuk tugas

tertentu dan pengamat spesifik. Dalam proses ini satu atau lebih atribut akan dimodifikasi, penentuan atribut yang akan dimodifkikasi dan bagaimana memodifikasinya bergantung dengan tugas yang diberikan.(Qi et al., 2021).

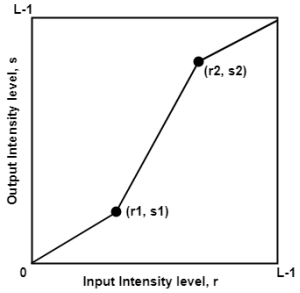
Teknik peningkatan citra secara besar dibagi menjadi dua kategori, yaitu; *Spatial Domain Methods* dan *Frequency Domain Methods*. Metode *Spatial Domain*, secara langsung memanipulasi pixel secara langsung untuk mendapatkan hasil yang diinginkan, sedangkan *Frequency Domain methods* mengubah citra terlebih dahulu kedalam domain frekuensi.(Qi et al., 2021)

2.3.1.1 Point Processing Operation

Merupakan transformasi tingkat *Spatial Domain* tersederhana, dikarenakan dilakukan langsung pada pixel sebuah citra itu sendiri (Qi et al., 2021). Point processing operation terdapat beberapa metode metode transformasi; *Negative Image*, *Thesholding Transformation*, *Intensity Transformation*, *Logarithmic Transformation*, *Powers-law Transformation*, *Piecewise Linear Transformation Function*, dan *Grey Level Slicing*.(Qi et al., 2021)

2.3.1.2 Contrast Stretching

Contrast Stretching merupakan teknik yang digunakan untuk mendapatkan citra baru dengan kontras yang lebih baik daripada kontras dari citra asalnya (Wakhidah, 2011). Dalam mengubah nilai kontras pixel dari pixel aslinya dapat menggunakan fungsi matematis yang dibuat oleh pengguna secara sewenang-wenang (Qi et al., 2021).



Gambar 2. 1 Grafik Intensitas Contrast
Sumber: Qi et al (2021)

Min-max stretching:

$$(r1, s1) = (r_{min}, 0)\ dan\ (r2, s2) = (r_{max}, L-1) \quad (2.\ 1)$$

$$Xnew = \frac{Xinput - Xmin}{Xmax - Xmin} \times 255 \quad (2.\ 2)$$

2.2 Principal Component Analysis

Principal Component Analysis (PCA), adalah formula matematika yang digunakan untuk mereduksi suatu dimensi data. Sehingga teknik PCA memungkinkan identifikasi suatu standart dalam data dan ekspresinya dalam suatu bentuk di mana kemiripian dan perbedaannya teteap tersimpan. Setelah suatu pola ditemukan, pola tersebut dapat dikompres. Seperti dengan mereduksi dimensi tanpa kehilangan informasi yang signifikan (Santo, 2012).

Permisalkan $x_1, x_2 \ldots x_n$ merupakan data dari vektor N^2x 1, dan $\overline{x}$ adalah rata-rata.

$$x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN^2} \end{bmatrix} \overline{x} = \frac{1}{n}\sum_{i=1}^{i=n} \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{iN^2} \end{bmatrix} \quad (2.\ 3)$$

Setiap N x N citra dapat di represntasikan sebagai vektor N^2x 1, di mana berisikan nilai pixel citra.

Permisalkan X sebagai N^2 x n matriks dengan kolom $x_1 - \overline{x}\ x_2 - \overline{x} \ldots x_n - \overline{x}$

$$x = [x_1 - \overline{x}\ x_2 - \overline{x} \ldots x_n - \overline{x}] \quad (2.\ 4)$$

Mengurangi mean equivalen dengan menerjemahkan koordinat menuju lokasi mean.

Permisalkan $Q = XX^T$ menjadi $N^2 \times N^2$matriks

$$Q = XX^T = [x_1 - \overline{x}\ x_2 - \overline{x} \ldots x_n - \overline{x}] \begin{bmatrix} (x_1 - \overline{x})^T \\ (x_2 - \overline{x})^T \\ \vdots \\ (x_n - \overline{x})^T \end{bmatrix} \quad (2.\ 5)$$

Q adalah kuadrat, Q adalah simetris, Q adalah matriks convariance, Q ukuran dapat menjadi besar

Setiap x_j dapat ditulis sebagai

$$x_j = \overline{x} + \sum_{i=1}^{i=n} g_{ji} e_i \qquad (2.6)$$

Di mana e_i adalah n eigenvectors dari Q dengan eigenvalues tidak nol.

Eigenvectors e_1 e_2 ... e_n merentang eigenspaces

e_1 e_2 ... e_n adalah vektor orthonormal N^2x1

Skalar g_{ji} adalah koordinat dari x_j dalam ruang

$$g_{ji} e_i = (x_j - \overline{x}).\, e_i \qquad (2.7)$$

2.3 Support Vector Machine

Support Vector Machine (SVM) adalah metode machine learning yang menggunakan teori pembelajaran statistika dan diklasifikasikan sebagai salah satu pendekatan komputasi yang di buat oleh Vladimir Vapnik, Bernhard Boser dan Isabelle Guyon. SVM didasari dengan prinsip *structural risk minimization* (SRM), oleh karena itu SVM bisa mendapatkan aturan penentuan dan meraih error yang kecil untuk set test mandiri sehingga bisa menyelesaikan permasalahan *learning* dengan efisien. SVM diimplementasikan untuk menyelesaikan berbagai masalah seperti nonlinear, *local minimum* dan *high dimension*. SVM dapat menjamin akurasi yang lebih tinggi untuk prediksi jangka panjang dibandingkan pendekatan komputasi lainnya, dalam penerapan di banyak aplikasi. SVM di dasari konsep *decision planes* yang mendefinisikan suatu batasan keputusan. SVM membuat *hyperplane* dengan mengunggakan model linear untuk implementasi batasan keputusan kelas nonlinear melalui pemetaan nonlinear *input vector* menjadi sebuah ruang fitur berdimensi tinggi (Mat Deris et al., 2011).

Permisalkan terdapat 2 atribut input A_1 dan A_2, himpunan $W = \{w_1, w_2, \cdots, w_d\}$; W merupakan bobot atau *weight*, d adalah jumlah atribut, dan tupel *traning*

$X = (x_1, x_2)$ di mana x_1 dan x_2 adalah nilai-nilai atribut A_1 dan A_2,maka fungsi *hyperplane* dapat dinotasikan sebagai berikut

$$f(x) = W.X + b \qquad (2.\ 8)$$

Di mana $W, X \in \mathbb{R}^d$ (d adalah jumlah atribut) dan b adalah bias yang berupa skalar.

Hyperplane yang terletak diantara dua set objek dari kelas positif ($y_1 = +1$) dan kelas negatif ($y_2 = -1$) dapat ditulis sebagai berikut.

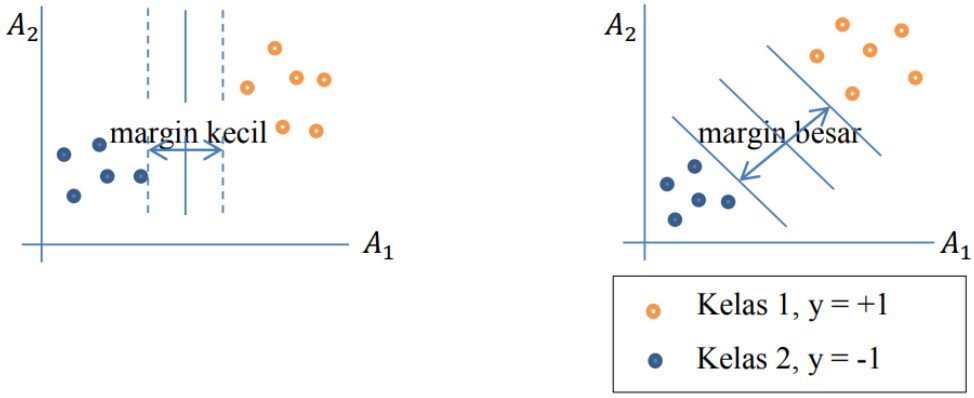
$$H_1: x_i w + b \geq 1 \text{ untuk } y_1 = +1 \qquad (2.\ 9)$$

$$H_2: x_i w + b \geq -1 \text{ untuk } y_1 = -1 \qquad (2.\ 10)$$

Penggabungan dari persamaan (2.9) dan (2.10) menghasilkan pertidaksamaan:

$$y_i(x_i w + b) \geq 1 \text{ untuk } \forall i = 1,2,\cdots,n \qquad (2.\ 11)$$

Margin antara dua kelas dapat dihitung dengan mencari jarak antara kedua *hyperplane* H_1atau H_2. Setiap tupel pelatihan yang jatuh pada *hyperplane* H_1atau H_2 yang memenuhi persamaan (2.9) disebut *support vector*. Jarak terdekat suatu titik di bidang terhadap pusat dapat dihitung dengan meminimalkan $x^T x$ dengan memperhatikan kendala $x_i w + b \geq 1$ (Perdana & Furqon, 2018).



Gambar 2. 2 Margin Kecil dan Margin Besar
Sumber: Perdana & Furqon (2018)

Ada tiga keuntungan SVM. Pertama, hanya perlu menentukan dua parameter yaitu batas atas dan kernel. Kedua, unik, optimal dan global untuk menyelesaikan sebuah permasalahan linear yang terbatasi secara quadratic. Ketiga, generalisasi performa yang baik dikarenakan implementasi prinsip SRM(Mat Deris et al., 2011).

2.3.1. Support Vector Classification

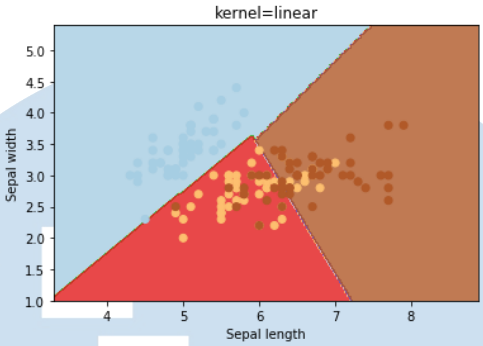
Support Vector Classifier (SVC) mencoba mencari *hyperplane* terbaik untuk memisahkan kelas kelas berbeda dengan cara memaksimalkan jarak antara titik sample dan *hyperplane* (Fraj, 2018). Klasifikasi adalah proses menemukan model (atau fungsi) yang menggambarkan dan membedakan kelas data atau konsep. Model yang diturunkan berdasarkan analisis dari set data pelatihan (data yang label kelasnya diketahui) yang kemudian digunakan untuk memprediksi label kelas yang tidak diketahui. Banyak metode klasifikasi telah diperkenalkan peneliti seperti machine learning, pattern recognition, dan statistik (Perdana & Furqon, 2018). Dalam klasifikasi terdapat beberapa parameter yang dapat dimodifikasi.

2.3.1.1. Kernel

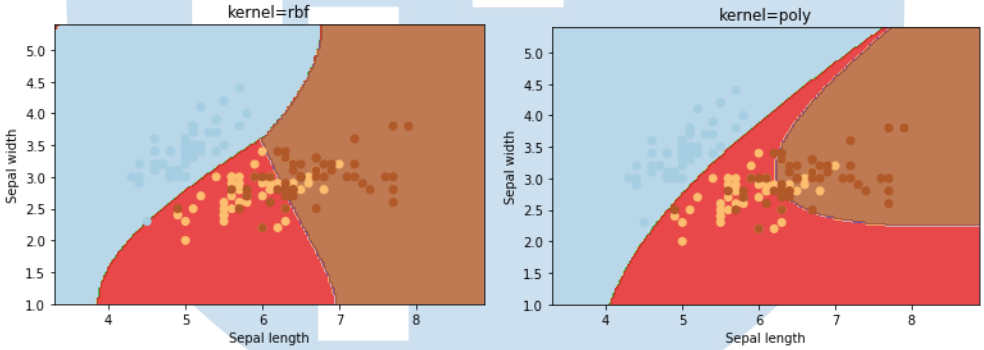
Secara umum, kasus klasifikasi di dunia nyata adalah kasus yang tidak linier sehingga digunakan metode kernel untuk mengatasi masalah tersebut. Dengan menggunakan metode kernel suatu data x di *input space* dipetakan ke *feature space* F dengan dimensi yang lebih tinggi melalui map $\varphi(\varphi: x \rightarrow \varphi(x))$. Oleh karena itu, x di *input space* menjadi $\varphi(x)$ di *feature space*. Sering kali fungsi $\varphi(x)$ tidak tersedia atau tidak dapat dhitung. Tetapi dot product dari dua vektor dapat dihitung, baik dalam input space maupun di feature space (Perdana & Furqon, 2018).

$$K(X_i \cdot X_j) = \varphi(X_i) \cdot \varphi(X_j) \qquad (2.\ 12)$$

Kernel digunakan untuk menentukan tipe *hyperplane* yang akan digunakan untuk mengklasifikasi data. Menggunakan *kernel* linear maka akan memnggunakan *hyperplane* linear ditunjukan pada gambar 2.3 dan menggunakan *kernel* rbf dan poly akan menggunakan *hyperplane* nonlinear ditunjukan pada gambar 2.4 (Fraj, 2018).



Gambar 2. 3 Kernel Linear
Sumber: Fraj (2018)



Gambar 2. 4 Kernel Nonlinear
Sumber: Fraj (2018)

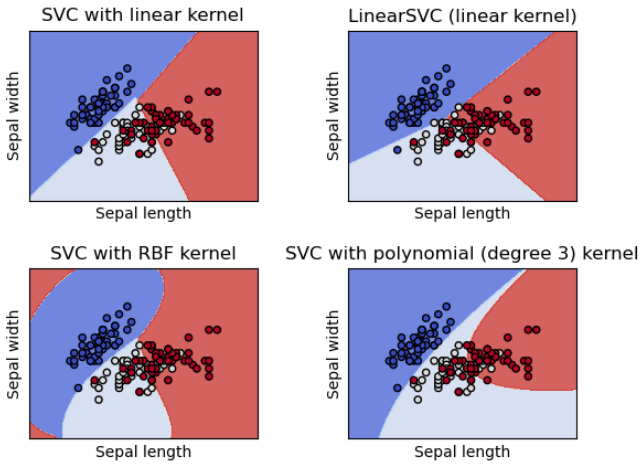
Fungsi *kernel* yang umum digunakan adalah sebagai berikut:

Kernel Derajat Polinomial h:

$$K(X_i \cdot {X_j}^T) = (X_i \cdot {X_j}^T + 1)^h \quad (2.\ 13)$$

Kernel Radial Basis Function:

$$K(X_i \cdot {X_j}^T) = \frac{e - \|x_i - x_j\|^2}{2\sigma^2} \quad (2.\ 14)$$



Gambar 2. 5 Perbandingan Kernel SVC
Sumber: Fraj (2018)

Radial Basis Function (RBF) *kernel* adalah bentuk *kenrel* yang paling tergeneralisasi dan salah satu *kernel* yang paling banyak digunakan dikarenakan kemiripannya dengan *Gaussian distribution*. Fungsi *kernel* RBF untuk dua point X1 dan X2 menghitung seberapa mirip atau dekat kedua titik tersebut. Fungsi kernel ini dapat dituliskan dengan rumus berikut.

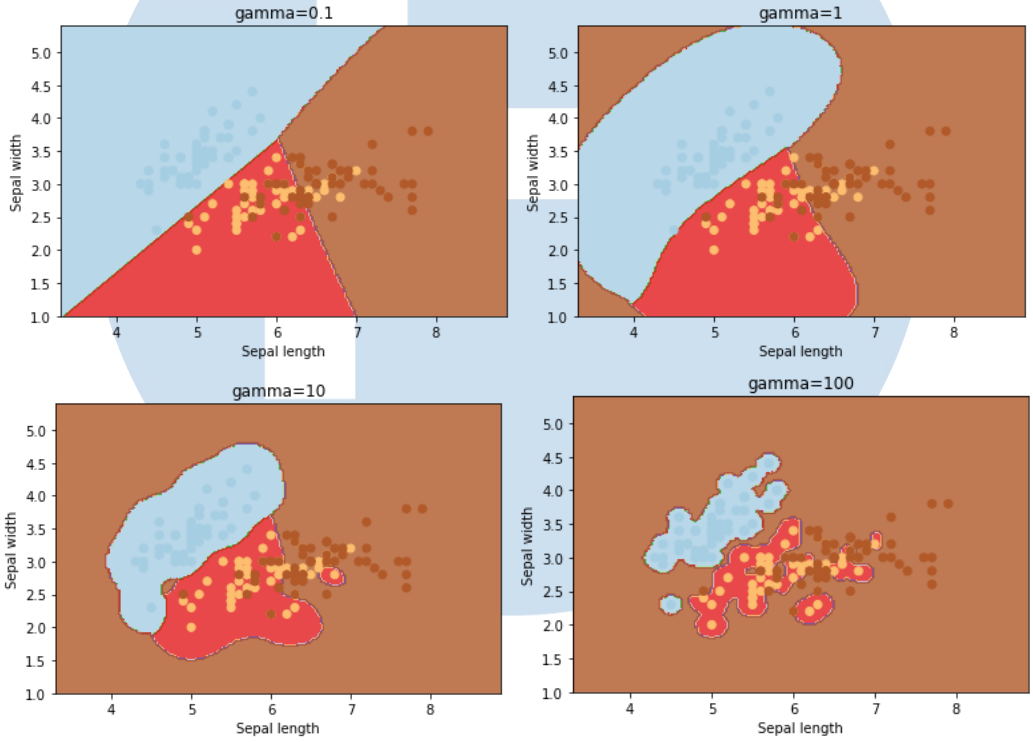
$$K(X1, X2) = \exp\left(-\frac{\|X1 - X2\|^2}{2\sigma^2}\right) \quad (2.\ 15)$$

Di mana, 'σ' adalah *variance* dan *hyperparameter*, $\|X1 - X2\|$ adalah jarak *euclidean* antara titik X1 dan X2.

Kernel RBF banyak digunakan karena kemiripannya dengan K-Nearest Neighborhood Algorithm. Kernel RBF memiliki kelebihan dibandingkan K-NN dan mengatasi permasalahan *space complexity*, karena *kernel* RBF hanya perlu menyimpan *support vectors* pada saat *training* dan tidak seluruh dataset (Sushant Sreenivasa, 2020).

2.3.1.2 Gamma

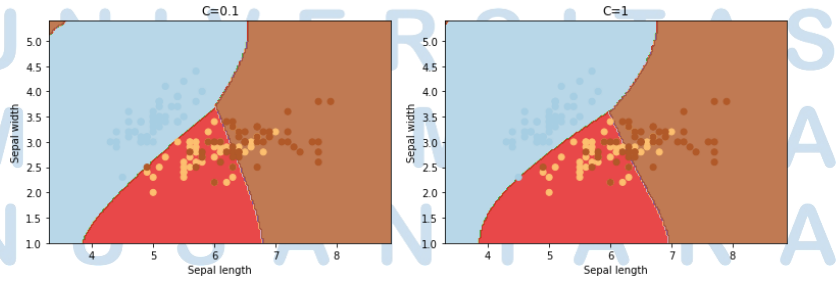
Gamma, *gamma* merupakan parameter untuk *hyperplane* nonlinear. Nilai pada *gamma* berpengaruh pada bagaimana model berusaha untuk mencoba menyesuaikan dengan dataset *training* (Fraj, 2018).

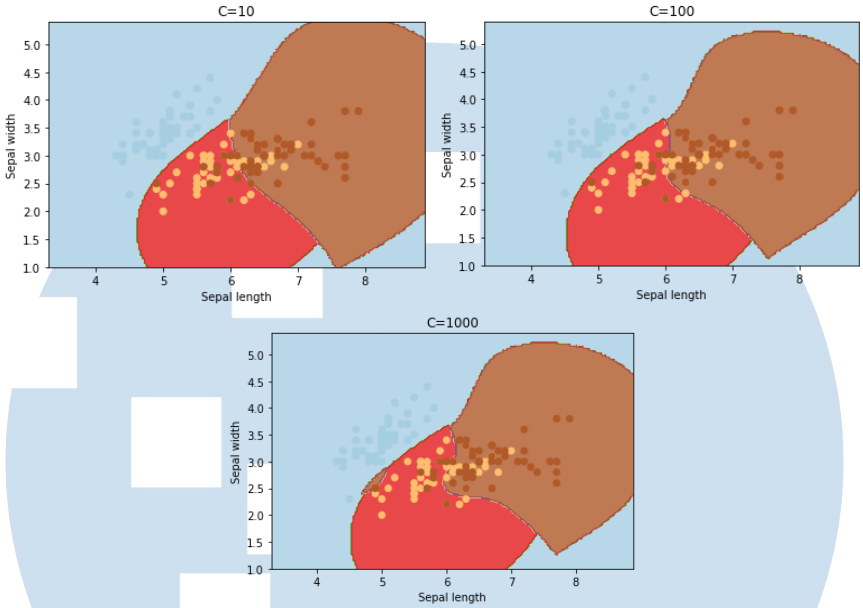


Gambar 2. 6 Perbandingan Gamma
Sumber: Fraj (2018)

2.3.1.3 C

C merupakan parameter pinalti untuk error. C mengatur pertukaran antara seberapa halus pembatasan keputusan dan mengklasifikasi point training dengan benar.

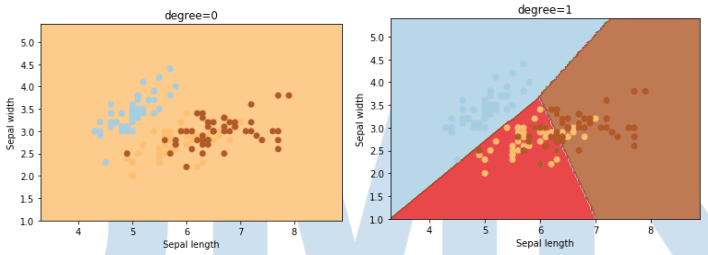




Gambar 2. 7 Perbandingan C
Sumber: Fraj (2018)

2.3.1.4 Degree

Degree adalah parameter yang digunakan untuk kernel poly. *Degree* pada dasarnya merupakan *degree* dari *polynomial* yang digunakan untuk mencari *hyperplane* untuk membagi data.



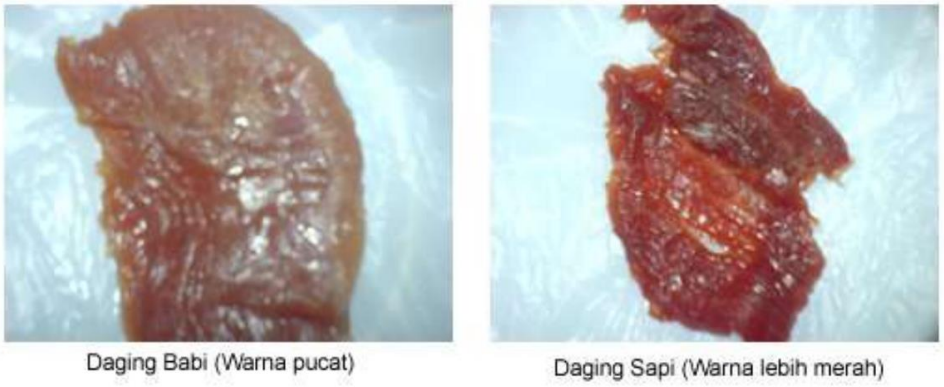
Gambar 2. 8 Perbandingan Degree
Sumber: Fraj (2018)

2.4 Perbedaan daging sapi dan daging babi

Terdapa beberapa perbedaan mendasar antara daging sapi dan babi, secara kasat mata terdapat secara kasat mata ada lima aspek yang terlihat berbeda antara daging babi dan sapi yaitu warna, serat daging, tipe lemak, aroma dan tekstur (Asyaukani, 2019).

1. Warna

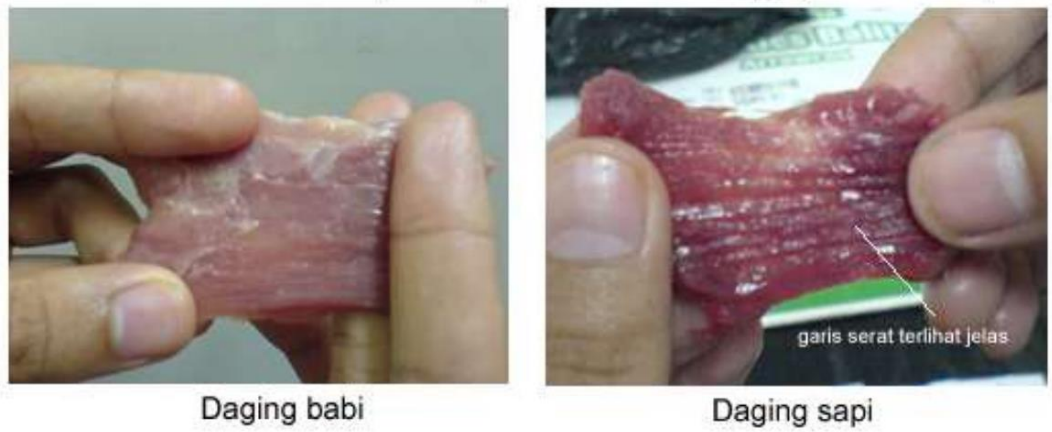
Daging babi memiliki warna yang lebih pucat dari daging sapi, warna daging babi mendekati warna daging ayam.



Gambar 2. 9 Perbandingan Warna Daging
Sumber: Asyaukani (2019)

2. Serat Daging

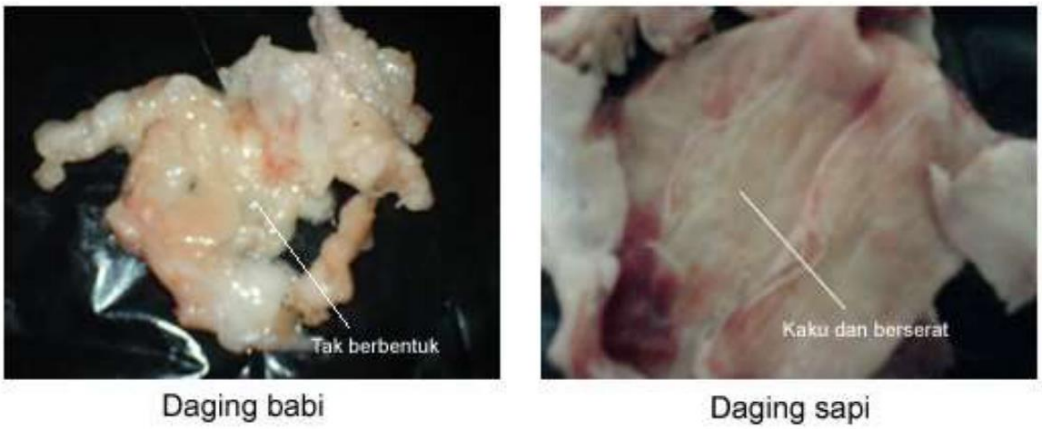
Terlihat perbedaan serat daging yang jelas antara kedua daging. Serat-serat daging sapi tampak padat dan garis-garis serat terlihat jelas. Sedangkan pada daging babi, serat-seratnya terlihat samar dan sangat renggang. Perbedaan ini semakin jelas ketika kedua daging direnggangkan bersama.



Gambar 2. 10 Perbandingan Serat Daging
Sumber: Asyaukani (2019)

3. Penampakan Lemak

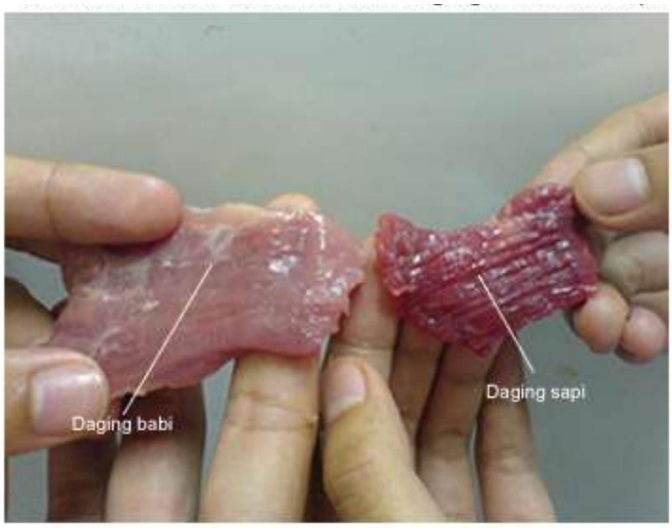
Perbedaan terdapat pada tingkat keelastisannya. Daging babi memiliki tekstur lemak yang lebih elastis sementara lemak sapi lebih kaku dan berbentuk. Selain itu lemak pada babi sangat basah dan sulit dilepas dari dagingnya sementara lemak daging agak kering dan tampak berserat.



Gambar 2. 11 Perbandingan Lemat Daging
Sumber: Asyaukani (2019)

4. Tekstur

Daging sapi memiliki tekstur yang lebih kaku dan padat dibanding dengan daging babi yang lembek dan mudah diregangkan.



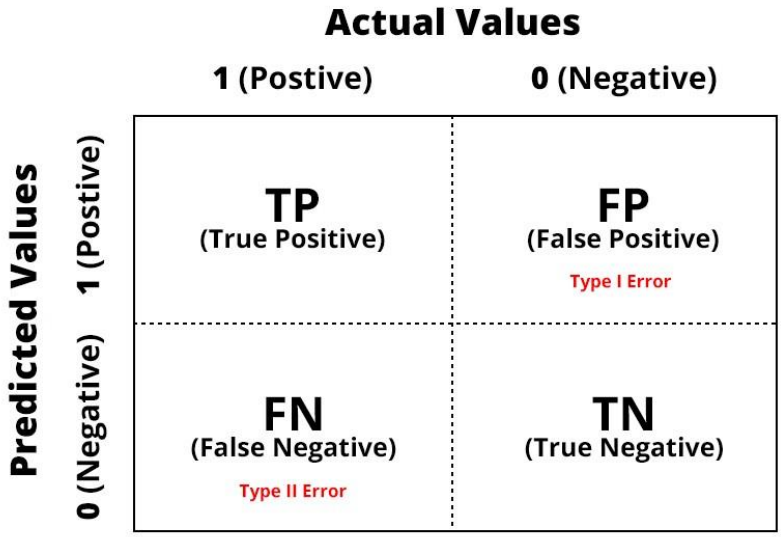
Gambar 2. 12 Perbandingan Tekstur Daging
Sumber: Asyaukani (2019)

5. Aroma

Terdapat sedikit perbedaan antara keduanya. Daging babi memiliki aroma khas tersendiri, sementara aroma daging sapi adalah anyir.

2.5 Confusion Matrix

Confusion Matrix juga sering disebut error matrix. Pada dasarnya *confusion matrix* memberikan informasi perbandingan hasil klasifikasi yang dilakukan oleh sistem (model) dengan hasil klasifikasi sebenarnya. Confusion matrix berbentuk tabel matriks yang menggambarkan kinerja model klasifikasi pada serangkaian data uji yang nilai sebenarnya diketahui. Gambar dibawah ini merupakan confusion matrix dengan empat kombinasi nilai prediksi dan nilai aktual yang berbeda.



Gambar 2. 13 Tabel Confuison Matrix
Sumber: Setyo Nugroho (2019)

Terdapat empat istilah sebagai representasi hasil proses klasifikasi pada confusion matrix. Keempat istilah tersebut adalah *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). *True Positive* (TP), merupakan data positif yang diprediksi benar; *True Negative* (TN), merupakan data negatif yang diprediksi benar; *False Positive* (FP), merupakan data negatif namun diprediksi positif; *False Negative* (FN), merupakan data positif namun diprediksi sebagai data negatif (Setyo Nugroho, 2019).

Akurasi menunjukan seberapa akurat model dapat mengkalsifikasi dengan benar, pada confusion matrix dapat dihitung dengan persamaan

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.\ 16)$$

Presisi menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model, pada confusion matrix dapat dihitung dengan persamaan

$$precision = \frac{TP}{TP + FP} \quad (2.\ 17)$$

Recall menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi, pada confusion matrix dapat dihitung dengan persamaan

$$recall = \frac{TP}{TP + FN} \quad (2.\ 18)$$

Kombinasi hasil untuk recall dan precision dapat disimpulkan kedalam beberapa arti, yaitu (Rocca, 2019).

- *Recall* tinggi dan *precision* tinggi, model secara sempurna menguasai kelas tersebut.
- *Recall* rendah dan *precision* tinggi, model tidak dapat mengklasifikasi kelas dengan baik akan tetapi dapat dipercaya ketika mengklasfikasi positif.
- *Recall* tinggi dan *precision* rendah, kelas dapat deteksi dengan baik akan tetapi model juga termasuk point dari kelas lainnya.
- *Recall* rendah dan *precision* rendah, model tidak menguasai klasifikasi kelas dengan baik.