BAB 2
LANDASAN TEORI

2.1 Malicious Traffic

Malicious traffic atau lalu lintas berbahaya mengacu pada setiap lalu lintas jaringan yang dimaksudkan untuk menyebabkan kerusakan, mengganggu operasi, mencuri data, atau melakukan tindakan tidak sah dalam jaringan atau sistem komputer. Ini dapat muncul dalam berbagai bentuk dan biasanya diatur oleh penjahat siber atau peretas dengan niat jahat [4]. Berikut adalah beberapa contoh dan karakteristik umum dari lalu lintas berbahaya:

1. *Malware*: Lalu lintas jaringan yang melibatkan distribusi atau komunikasi dengan *malware*, seperti *trojan*, virus, *adware*, *ransomware*, *spyware*, atau *rootkit* [12]. Ini dapat mencakup lalu lintas yang dihasilkan oleh mesin yang terinfeksi dalam jaringan.

2. *Phishing*: Lalu lintas yang terkait dengan upaya untuk menipu pengguna agar memberikan informasi sensitif, seperti kredensial login, melalui *website* atau *email* palsu [13].

3. Serangan DDoS: Serangan *Distributed Denial of Service* melibatkan membanjiri jaringan atau *server* dengan lalu lintas yang berlebihan, menyebabkan layanan seperti *website* dan *email* tidak tersedia bagi pengguna yang sah [14].

4. Lalu Lintas Command and Control (C&C): Komunikasi antara sistem yang ter-*compromised* dan server kontrol *attacker*, sering digunakan untuk mengelola dan mengatur serangan atau mencuri data [15].

5. SQL Injection: Lalu lintas yang melibatkan upaya untuk mengeksploitasi kerentanan aplikasi web. SQL dapat mempenetrasi database-database yang bergantung pada SQL [16].

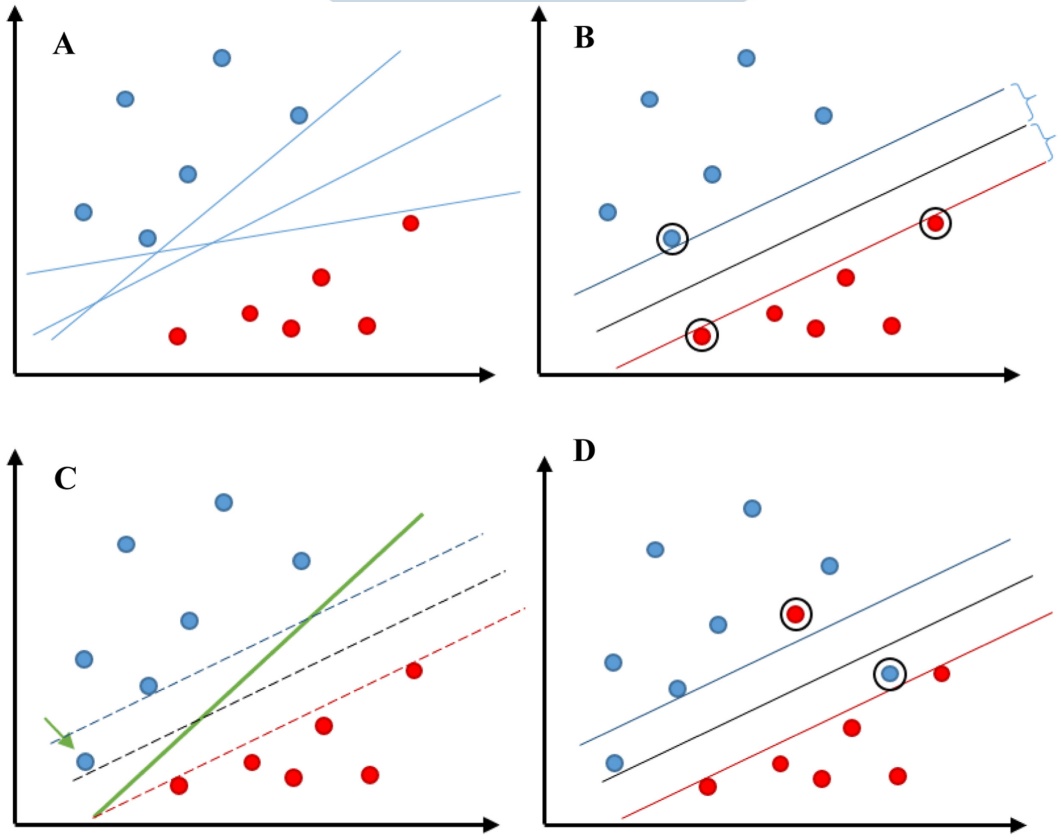
2.2 Support Vector Machine

Support Vector Machine (SVM) adalah model *supervised machine learning* yang digunakan untuk menganalisis *regression* dan *classification*. SVM menjadi

salah satu metode ML yang populer karena kesederhanaan dan fleksibilitasnya untuk mengatasi berbagai masalah klasifikasi [17]. SVM juga digunakan dalam *computer security* contohnya untuk *intrusion detection* [18].

Ada beberapa ketergantungan (*mapping, function*) yang tidak diketahui dan nonlinear y = f(x) antara beberapa vektor input x berdimensi tinggi dan *output* skalar y (atau *output* vektor y seperti pada kasus *multiclass* SVM). Tidak ada informasi tentang fungsi probabilitas bersama yang mendasarinya. Satu-satunya informasi yang tersedia adalah set data *training*. Karena itu, SVM termasuk dalam teknik *supervised learning* [19].

SVM cukup *straightforward* karena SVM mencoba menyelesaikan masalah klasifikasi biner tertentu dengan model yang paling sederhana, memisahkan subjek yang termasuk dalam 2 kelas berbeda dengan batasan klasifikasi. Dalam 2 dimensi, batasan klasifikasi ini akan membentuk garis lurus. Dalam 3 dimensi, batasan klasifikasi ini akan menjadi bidang, sebuah generalisasi garis. Batasan ini akan disebut *hyperplane* untuk dimensi yang lebih tinggi, yang dapat dianggap sebagai bidang dalam lebih dari 3 dimensi.



Gambar 2.1. Ilustrasi data dengan hyperplane berbeda-beda

Dalam Gambar 2.1 [20], figur A menunjukkan masalah klasifikasi yang dapat dipisahkan. Namun, seperti yang diilustrasikan dalam plot, terdapat beberapa *hyperplane* yang berbeda. Solusinya terdapat dalam Gambar 2.1, figur B dan disebut sebagai *classifier margin* maksimum. Untuk meminimalkan risiko salah klasifikasi, batas klasifikasi kita tempatkan sejauh mungkin dari subjek-subjek tetangga yang termasuk dalam kelas yang berbeda. Margin dimaksimalkan antara batas klasifikasi dan data *training* yang memungkinkan adanya wilayah toleransi saat memprediksi label kelas untuk subjek baru.

Titik-titik data yang jauh dari garis klasifikasi tidak memengaruhi posisi batasan data. Titik-titik data yang menentukan batas keputusan adalah 3 titik dengan lingkaran hitam pada Gambar 2.1, figur B. Titik-titik ini disebut sebagai *support vector*. Dengan kata lain, jika kita menghapus semua subjek dari dataset *training* kita selain 3 *support vector* ini, maka lokasi batas keputusan akan tetap tidak berubah. Contoh ini menunjukkan bahwa *support vector* secara signifikan memengaruhi batas keputusan, dan perubahan dalam data *training* akan berdampak besar pada batas keputusan.

Gambar 2.1, figur C menunjukkan subjek tambahan yang ditunjukkan oleh panah yang ditambahkan ke dataset *training*. Subjek ini berada dekat dengan batas keputusan dan merupakan *support vector* yang berpengaruh yang akan memodifikasi masalah margin maksimum, menghasilkan batas klasifikasi yang berbeda, seperti yang ditunjukkan oleh warna hijau. Dan pada Gambar 2.1, figur D terdapat subjek dilingkar hitam yang salah klasifikasi. Ini menunjukkan data yang tidak bisa dipisahkan [20].

2.3 Naive Bayes

Naive Bayes (NB) adalah algoritme *supervised learning* yang memanfaatkan *Bayes's rule* bersama dengan asumsi kuat bahwa atribut-atribut bersifat independen secara kondisional berdasarkan *class*. Walaupun sering kali asumsi independensi ini tidak terpenuhi dalam penggunaannya, Naive Bayes masih sering memberikan tingkat akurasi klasifikasi yang kompetitif. Kombinasi antara efisiensi komputasinya dan berbagai keuntungan lainnya membuat Naive Bayes menjadi pilihan yang populer dalam praktik [21].

Berdasarkan estimasi parameter yang dihitung dari data *training*, klasifikasi dapat dilakukan pada *test data* dengan menghitung *posterior probability* dari setiap *class* berdasarkan bukti dari dokumen uji, dan memilih *class* dengan probabilitas

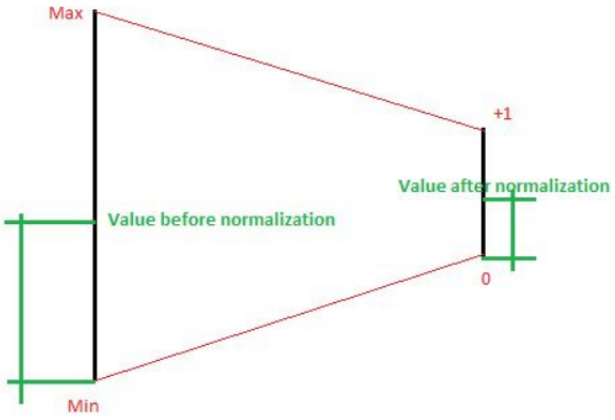
tertinggi [22]. Teorema Bayes menyediakan cara untuk menghitung *posterior probability* dari *P(c/x), P(c), P(x) dan P(x/c)*. Formula tersebut adalah:

$$P(c/x) = \frac{P(c) * P(x/c)}{P(x)}$$

- *P(c/x) : Posterior Probability*
- *P(c) : Class Prior Probability*
- *P(x/c) : Likelihood*
- *P(x) : Predictor Prior Probability*

2.4 Normalisasi

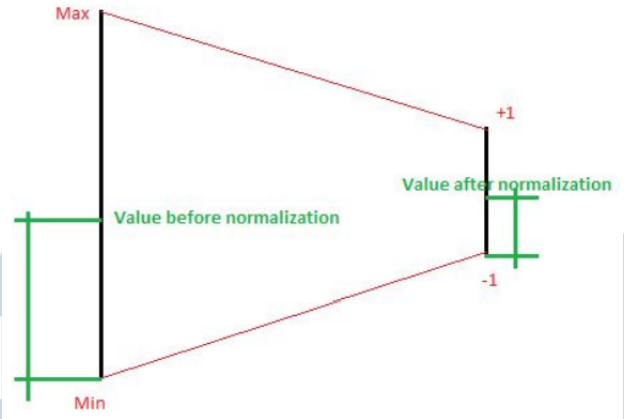
Normalisasi merupakan proses mengubah data ke dalam rentang yang ditentukan, seperti antara 0 dan 1 atau antara -1 dan +1. Normalisasi diperlukan saat adanya perbedaan besar dalam rentang fitur yang berbeda. Metode penskalaan ini berguna ketika set data tidak mengandung *outlier* [23].



Gambar 2.2. Visualisasi normalisasi.

Gambar 2.2 [23] menunjukkan *value* dari sebuah data sebelum dan sesudah normalisasi jika perlu diubah ke rentang 0,1.

Jika normalisasi digunakan, terkadang perlu dilakukan juga denormalisasi. Denormalisasi dilakukan untuk mengoptimasi data *retrieval* dan meningkatkan kinerja.



Gambar 2.3. Visualisasi denormalisasi.

Gambar 2.3 [23] adalah visualisasi data yang telah di-denormalisasi.

2.5 ROC AUC

Kurva ROC (*Receiver Operating Characteristic* adalah grafik yang menunjukkan performa dari model klasifikasi. ROC mengukur dua parameter:

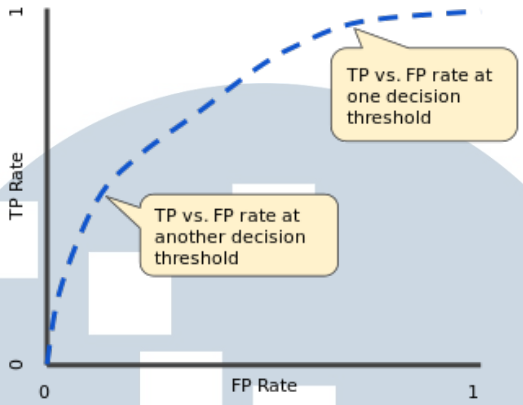
True Positive Rate (TPR)

$$TPR = \frac{TP}{TP + FN} \tag{2.1}$$

False Positive Rate FRP

$$FRP = \frac{FP}{FP + TN} \tag{2.2}$$

Kurva ROC memplot TPR dengan FPR pada ambang yang berbeda. Menurunkan ambang klasifikasi akan meningkatkan TPR dan FPR. Berikut adalah contoh kurva ROC:

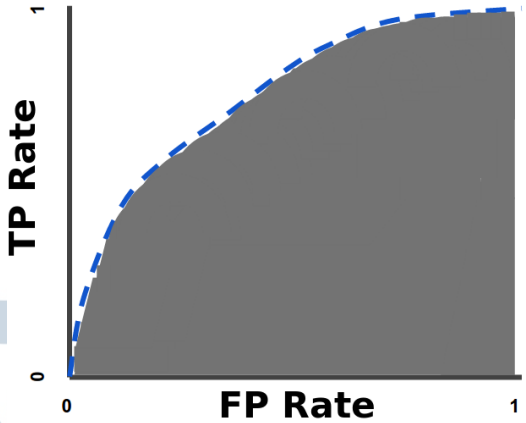


Gambar 2.4. TPR vs FPR

Gambar 2.4 [24], adalah kurva ROC dengan dua ambang klasifikasi yang berbeda.

2.6 AUC

AUC (*Area Under Curve*) mengukur area dua dimensi di bawah kurva ROC. Seperti pada Gambar 2.5 [24].



Gambar 2.5. Area Under ROC Curve

AUC memiliki nilai yang berkisar dari 0 hingga 1. Model yang prediksinya 100% salah memiliki AUC sebesar 0,0. Sedangkan model yang prediksinya 100% benar memiliki AUC sebesar 1,0.