BAB II
TINJAUAN PUSTAKA

2.1. Tinjauan Pustaka

Banyak penelitian terdahulu yang menggunakan algoritma *Random Forest* dalam berbagai konteks, terutama risiko kredit. Penerapan *Random Forest* pada klasifikasi risiko kredit menggunakan arsitektur yang standar (menggunakan parameter *default*) sehingga mendapatkan akurasi rendah. Namun ada beberapa penelitian yang menggunakan *feature engineering* dan *hyperparameter tuning* pada klasifikasi risiko kredit menunjukkan hasil yang lebih baik. Akan tetapi penelitian tersebut belum diterapkan pada data lain, seperti data risiko kredit atau "credit risk data" yang diunggah oleh Lao Tse [34]. Penelitian terkait *Random Forest* pada domain lain menunjukkan bahwa penerapan *feature engineering* dan *hyperparameter tuning* dapat meningkatkan keakuratan model *Random Forest*. Dalam beberapa penelitian, teknik-teknik tersebut berhasil mengoptimalkan performa model dengan memilih fitur-fitur yang paling informatif dan mengatur parameter-parameter yang tepat, sehingga meningkatkan prediksi dan keandalan model secara signifikan. Dengan memanfaatkan penelitian-penelitian sebelumnya, dapat diperoleh wawasan yang berharga dalam mengimplementasikan algoritma *Random Forest* dengan akurasi yang lebih tinggi. Berikut ini beberapa penelitian terdahulu yang menjadi rujukan penelitian ini.

Penelitian [10] tentang bagaimana menggunakan algoritma *Random Forest* untuk memprediksi kelayakan kredit secara dini. Keberagaman dan banyaknya fitur pada data serta adanya ketidak seimbangan kelas membuat algoritma *Random Forest* diterapkan pada penelitian ini. Penelitian ini bertujuan untuk mengetahui penerapan dan hasil terbaik dari algoritma *Random Forest* untuk klasifikasi risiko kredit pada data *German Credit Risk*. Data tersebut memiliki 1000 baris dengan 21 fitur. Pada penelitian ini mendapatkan akurasi model 83%.

Penelitian [13] yang bertujuan untuk membandingkan akurasi model *Random Forest*, *Deep Neural Network*, dan Adaboost. Penelitian ini menerapkan *hyperparameter tuning* pada model karena dianggap dapat membantu

meningkatkan akurasi, stabilitas, serta menangani masalah *underfitting* maupun *overfitting* pada data yang terdapat banyak fitur. Penelitian ini berhasil mendapatkan akurasi tertinggi pada *Random Forest* sebesar 90.63%. Namun pada penelitian ini kurang menjelajahi parameter lain pada *Random Forest* untuk dilakukan *hyperparameter tuning*.

Dilanjutkan dengan penelitian [9] yang melihat bahwa penelitian sebelum – sebelumnya yang tidak menerapkan *hyperparameter tuning* pada model *Random Forest*. Sehingga penelitian ini bertujuan untuk menerapkan *hyperparameter tuning* dan penyetelan strategi karena beberapa parameter di *Random Forest* dipercaya dapat mempengaruhi kinerjanya, parameter yang kurang tepat akan mempengaruhi hasil akurasi model. Penelitian ini memaparkan bahwa dengan melakukan *hyperparameter tuning* pada parameter *mtry*, *sample size*, *replacement*, *node size*, *number of trees*, dan *splitting rule* maka nilai *mtry* memiliki pengaruh besar terhadap keakuratan model. Sedangkan nilai *sample size* dan *node size* berpengaruh kecil terhadap performanya.

Didukung penelitian [14] yang menerapkan *hyperparameter tuning* model *Random Forest* pada data perusahaan di China dengan jumlah 3111 dan tidak seimbang (*imbalanced data*). Data yang berdimensi tinggi dan *imbalanced* membuat algoritma *Random Forest* diterapkan pada penelitian ini, namun jumlah parameter yang tidak tepat dianggap krusial pada hasil evaluasi model. Sehingga penelitian ini menerapkan *hyperparameter tuning* pada beberapa parameternya. Penerapan *hyperparameter tuning* ini berhasil meningkatkan akurasi model menjadi 99.67% dan nilai AUC yang didapat yaitu 0.9998, artinya model dapat membedakan kelas 0 dan 1 dengan baik.

Penelitian [11] tentang bagaimana algoritma optimasi pada *Random Forest* dapat meningkatkan akurasi model dan mampu mendeteksi kegagalan jantung pasien. Penelitian ini bertujuan untuk membandingkan 3 algoritma optimasi pada *Random Forest* yaitu *Grid Search*, *Random Search*, dan *Bayesian Search* untuk menghasilkan akurasi terbaik karena pada data yang mempunyai banyak fitur, akurasi model *Random Forest* yang didapat masih rendah. Penelitian ini menunjukkan hasil bahwa optimasi terbaik menggunakan *Random Search* dengan

akurasi tertinggi sebesar 85.63%.

Didukung penelitian [15] tentang bagaimana penerapan *hybrid* dari *Random Forest* dan *Grid Search* dalam menyetel *hyperparameter tuning* pada model. Pencarian akurasi menggunakan data *training* dapat menghasilkan *overfitting*, sehingga perlu dilakukan penyesuaian *hyperparameter*. Penelitian ini bertujuan melakukan pendekatan *hybrid* algoritma *Random Forest* dan *Grid Search* untuk menghasilkan akurasi model yang optimal. *Hyperparameter tuning* dilakukan pada jumlah *n_estimators* dan *max_features* sehingga berhasil meningkatkan akurasi model baseline menjadi 90.02%.

Didukung penelitian [16] yang memaparkan tentang bagaimana penerapan *feature engineering* dan *hyperparameter tuning* untuk meningkatkan akurasi model *Random Forest* dan menghasilkan model terbaik. Data yang berdimensi tinggi mempunyai kecenderungan *overfitting*, sehingga perlu dilakukan *feature selection.* Selain itu, akurasi model *baseline* yang masih rendah membuat penelitian ini menerapkan *hyperparameter tuning*. Penelitian ini melakukan *hyperparameter tuning* pada nilai *n_estimators, min_sample leaf, max_features*, dan *max_depth* pada data berjumlah 12520 baris. Hasil akurasi model meningkat menjadi 71.88%.

Penelitian [17] tentang bagaimana penerapan hasil *hyperparameter tuning* menggunakan *Grid Search* untuk mendapatkan akurasi terbaik. Model yang terlalu kompleks membuat model tersebut *overfitting*, sehingga perlu dilakukan *hyperparameter tuning*. Penelitian ini bertujuan untuk mendapatkan akurasi model *Random Forest* yang paling tepat dan optimal. *Hyperparameter tuning* dilakukan pada parameter *max_depth, max_number_trees, weight parameter* dan *feature evaluation*. Penelitian ini menghasilkan akurasi yang sangat baik yaitu 97%.

Penelitian [18] memaparkan kondisi data yang tidak seimbang (*imbalanced*) dapat mempengaruhi proses *training* dan menyebabkan bias yang tidak menguntungkan terhadap kelas mayoritas. Algoritma yang digunakan yaitu *Random Forest* juga sensitif terhadap parameter – parameter yang digunakan. Untuk itu diperlukan penyeimbangan data ke kelas mayoritas atau *oversampling* dan *hyperparameter tuning* model *Random Forest* untuk mendapatkan hasil akurasi yang lebih baik. Hasil penelitian ini yaitu adanya peningkatan nilai akurasi dari 94%

menjadi 96% setelah menerapkan *oversampling* dan *hyperparameter tuning* menggunakan *Grid Search CV*.

Penelitian [19] yang menggunakan teknik *Random Oversampling* untuk mengatasi *imbalanced* data model *Random Forest* untuk klasifikasi risiko kehamilan. Penelitian ini mendapatkan hasil bahwa model yang dibuat setelah menerapkan *Random Over Sampling* mengalami kenaikan 1.09% pada akurasi model *Random Forest* menjadi 81.86%. Ringkasan 10 penelitian sebelumnya yang dijadikan referensi dalam penelitian ini terdapat pada Tabel 2.1.

Tabel 2.1 Penelitian Terdahulu

Literature Review		**Masalah Penelitian/ Rumusan Masalah**	**Tujuan**	**Metode**	**Data**	**Hasil / Temuan/ Kesimpulan**
Penulis, Tahun	**Judul**					
Budi Prasojo, Emy Haryatmi, (2020)	Analisa Prediksi Kelayakan Pemberian Kredit Pinjaman dengan Metode *Random Forest*	Keberagaman dan banyaknya fitur data pemberian kredit dan adanya kelas yang tidak seimbang (*imbalanced*), sehingga *Random Forest* tepat digunakan pada data yang ada.	Mengetahui penerapan dan hasil akurasi terbaik menggunakan algoritma *Random Forest* pada data *German Credit*.	Klasifikasi *Random Forest*	*German Credit* Data (1000 baris dengan 21 fitur) dan *imbalanced class*	Akurasi sebesar 83%.
Joseph Sanjaya, Erick Renata, Vincent Elbert Budiman, Francis Anderson, Mewati Ayub, (2020)	Prediksi Kelalaian Pinjaman Bank Menggunakan *Random Forest* dan *Adaptive Boosting*	Adanya fitur data yang beragam dan data yang tidak seimbang (*imbalanced*) sehingga penelitian ini menggunakan *Random Forest*, DNN, dan *Adaboost*.	Membandingkan akurasi dan performa model *Random Forest*, DNN, dan *Adaboost* pada data dengan fitur yang banyak.	*Random Forest*, *Adaboost*, dan DNN dengan melakukan *Hyperparameter Tuning*.	Data pinjaman (*loan*) yang diperoleh dari salah satu bank di Indonesia dengan 20 fitur.	Akurasi tertinggi sebesar 90.63% dengan model *Random Forest* dan dilakukan *undersampling*.

Literature Review		Masalah Penelitian/ Rumusan Masalah	Tujuan	Metode	Data	Hasil / Temuan/ Kesimpulan
Penulis, Tahun	**Judul**					
Philipp Probst, Marvin N. Wringht, Anne-Laure Boulesteix, (2018)	*Hyperparameters and Tuning Strategies for Random Forest*	Algoritma *Random Forest* memiliki beberapa parameter yang dapat mempengaruhi kinerjanya. Parameter yang kurang tepat dapat mempengaruhi model, sehingga perlu dilakukan *hyperparameter tuning*	Melakukan *hyperparameter tuning* dan penyetelan strategi untuk mendapatkan akurasi lebih baik pada *Random Forest* dengan melihat parameter berharga pada data yang fiturnya banyak.	*Random Forest* dengan melakukan tuning parameter pada *mtry, sample size, replacement, node size, number of trees,* dan *slitting rule*.	Data dari Open ML berjudul *monks-problem*.	*Mtry* dianggap berpengaruh sangat besar pada *tuning parameter* penelitian ini. Namun *sample size* dan node *size* berpengaruh kecil kepada performa.
Mohammad S. Uddin, Guotai Chi, Mazin A.M. Al Janabi, Tabassun Habib, (2020)	*Leveraging Random Forest in Micro-Enterprises Credit Risk Modelling for Accuracy and Interpretability*	Data yang digunakan memiliki ukuran besar, berdimensi tinggi dan *imbalanced* sehingga digunakan *Random Forest*. Namun jumlah pohon dianggap sangat krusial terhadap kebaikan model, sehingga dilakukan *hyperparameter tuning*.	Penerapan *Random Forest* pada pembuatan model klasifikasi risiko kredit pada data yang besar, berdimensi tinggi dan *imbalanced*.	*Random Forest* dengan menerapkan *hyperparameter tuning* pada *node* dan jumlah pohon.	Data perusahaan mikro di China dengan jumlah data 3111 dengan kondisi tidak seimbang.	Hasil klasifikasi algoritma *Random Forest* mendapatkan nilai AUC 0.9998 dan akurasi model 99.67%.

Literature Review		**Masalah Penelitian/ Rumusan Masalah**	**Tujuan**	**Metode**	**Data**	**Hasil / Temuan/ Kesimpulan**
Penulis, Tahun	**Judul**					
Unang Sunarya, Tita Haryanti, (2022)	Perbandingan Kinerja Algoritma Optimasi pada Metode *Random Forest* untuk Deteksi Kegagalan Jantung	Penerapan *Random Forest* memiliki akurasi yang rendah pada data yang mempunyai banyak fitur (kasus deteksi kegagalan jantung), sehingga diperlukan 3 teknik optimasi untuk dibandingkan hasil akurasi terbaiknya.	Membandingkan 3 algoritma optimasi pada algoritma *Random Forest* untuk mendapatkan akurasi terbaik.	*Grid Search, Random Search,* dan *Bayesian Search.*	Data publik berjudul '*Heart Failure Clinical Records Data*' yang terdiri dari 12 kolom fitur dan 299 baris.	Hasil dari *Random Forest* menggunakan optimasi *Random Search* mendapatkan akurasi tertinggi sebesar 85.63%.
Siji George C G, B. Sumathi, (2020)	*Grid Search Tuning of Hyperparameters in Random Forest Classifier for Customer Feedback Sentiment Prediction*	Mencari akurasi model dari data *training* dapat menghasilkan nilai *overfitting*, sehingga perlu dilakukan penyesuaian *hyperparameter*. *Grid Search* digunakan untuk mengidentifikasi parameter optimal dari klasifikasi sehingga model dapat secara akurat memprediksi data tidak berlabel.	Melakukan pendekatan *hybrid* algoritma *Random Forest* dan *Grid Search* untuk menghasilkan model yang optimal	*Random Forest* dengan menerapkan *tuning parameter* pada *n_estimators* dan *max_features* menggunakan *Grid Search*.	Data *sentiment analysis* dari UCI *database* berjumlah 1500 *reviews* pelanggan (*balanced data*)	Akurasi naik 5.5% menjadi 90.02% setelah menggunakan *proposed method*.

Literature Review		**Masalah Penelitian/ Rumusan Masalah**	**Tujuan**	**Metode**	**Data**	**Hasil / Temuan/ Kesimpulan**
Penulis, Tahun	**Judul**					
Phusanisa Charoen-Ung, Pradit Mittrapiyanuruk, (2019)	*Sugarcane Yield Grade Prediction Using Random Forest with Forward Feature Selection and Hyperparameter Tuning*	Data yang berdimensi tinggi mempunyai kecenderungan *overfitting*, sehingga perlu dilakukan *feature selection.* Di sisi lain, akurasi model *baseline* yang masih rendah membuat diterapkannya *hyperparameter tuning*.	Melakukan *feature selection* dan *hyperparameter tuning* pada model *Random Forest* untuk mendapatkan akurasi yang terbaik	*Random Forest* dengan tuning parameter *n_estimators*, *min_sample_leaf*, *max_features*, dan *max_depth*	Data berjumlah 12520 baris	Akurasi model yang dibuat mendapatkan hasil tertinggi sebesar 71.88%.
Wensu Zhao, Jie Hou, Qili Ran, (2022)	*Analysis of Corporate Credit Risk Based on Random Forest and TOPSIS Models*	Kekompleksan model dapat menimbulkan *overfitting*, sehingga penelitian ini menerapkan *hyperparameter tuning* untuk mendapatkan parameter terbaik dan optimal untuk pembangunan model pada data yang *imbalanced*.	Mendapatkan akurasi model *Random Forest* yang paling tepat dan optimal pada *data imbalanced* dengan *GridSearchCV*.	Menggunakan model *Random Forest* dengan *GridSearchCV* pada *max_depth*, *max number of trees, weight parameter,* dan *feature evaluation.*	Data milik perusahaan di China dengan jumlah 123 baris dengan 8 fitur.	Akurasi model yang didapat 97%

Literature Review		Masalah Penelitian/ Rumusan Masalah	Tujuan	Metode	Data	Hasil / Temuan/ Kesimpulan
Penulis, Tahun	**Judul**					
Anna Baita, Inggar Adi prasetyo, Nuri Cahyono (2023)	*Hyperparameter Tuning on Random Forest for Diagnose Covid-19*	Adanya ketidakseimbangan data yang mempengaruhi proses pelatihan dan menyebabkan bias yang tidak menguntungkan terhadap kelas mayoritas. *Random Forest* sendiri merupakan metode yang sensitif terhadap nilai parameternya. Sehingga diperlukan *balancing data* dan *hyperparameter tuning*.	Membandingkan penggunaan *oversampling* dan *hyperparameter tuning* pada model *Random Forest* khsusunya pada kasus Diagnosa penyakit Covid-19.	*Oversampling* dan *hyperparameter tuning* model *Random Forest.*	Data penyakit Covis-19 dengan 20 fitur.	Penggunaan *oversampling* dan *hyperparametre tuning* khususnya *Grid Search* dapat meningkatkan akurasi model menjadi 96%.
Riska Aryanti, Titik Misriati, Rahmat Hidayat (2023)	Klasifikasi Risiko Kesehatan Ibu Hamil Menggunakan *Random Oversampling* Untuk Mengatasi Keseimbangan Data	Masalah ketidakseimbangan data yang terjadi pada klasifikasi risiko kesehatan ibu hamil yang menyebabkan model klasifikasi yang dibuat cenderung akan memprediksi kelas mayoritas saja.	Menerapkan *Random Oversampling* untuk mengatasi *imbalanced* pada data untuk mendapatkan nilai akurasi yang lebih baik.	Menggunakan teknik *Random Oversampling* untuk mengatasi *imbalanced data* model *Random Forest* untuk klasifikasi risiko kehamilan.	Data berasal dari UCI *Machine learning* yang terdiri dari 1014 baris dengan 6 fitur.	Kenaikan 1.09% pada akurasi model RF pada data hasil *Random Oversampling*

2.2. Landasan Teori

2.2.1. *Credit Risk* / Risiko Kredit

Surat Edaran Bank Indonesia No.13/24/DPNP/2011 menyatakan bahwa risiko kredit adalah risiko akibat kegagalan debitur dan atau pihak lain dalam memenuhi kewajiban kepada Bank. Risiko kredit terjadi karena ketidakmampuan nasabah atau peminjam dalam melunasi kredit di waktu yang ditentukan pada perjanjian awal [20]. Adanya keterlambatan pembayaran kredit pada suatu lembaga keuangan membuat nilai *Non-Performing Loan* (NPL) meningkat. Dari pernyataan tersebut dapat dikatakan bahwa semakin tinggi nilai *Non-Performing Loan* maka kualitas kredit di lembaga keuangan tersebut bisa disebut buruk.

Bank Indonesia menentukan batas maksimal nilai NPL adalah 5%. Jika nilai NPL di bawah 5% artinya lembaga keuangan dinyatakan sehat dalam peminjaman kreditnya. Namun jika NPL di atas 5% artinya lembaga keuangan tersebut mengalami kerugian atau likuiditas [21]. Perhitungan nilai *Non-Performing Loan* sesuai dengan persamaan 2.1.

$$NPL = \frac{Kredit\ Bermasalah}{Total\ Kredit} \qquad (2.1)$$

Keterangan:

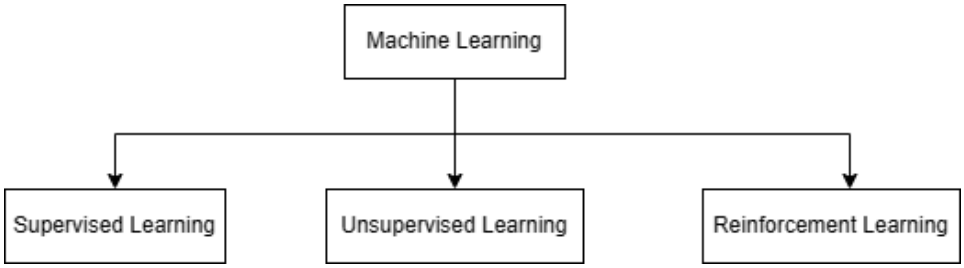
1. NPL (*Non-Performing Loan*) mencakup kredit yang mengalami tunggakan pembayaran atau tidak dapat dipenuhi oleh peminjam sesuai perjanjian.
2. Kredit bermasalah merupakan jumlah total kredit yang dianggap bermasalah atau tidak lancar.
3. Total kredit merupakan total keseluruhan kredit yang diberikan oleh lembaga keuangan kepada peminjamnya.

Untuk mengajukan kredit sendiri terdapat beberapa persyaratan yang harus diinputkan oleh kreditur kepada lembaga keuangan seperti identitas peminjam, penghasilan, jumlah pinjaman, dan lain sebagainya. Dari data tersebut risiko kredit bisa diklasifikasikan dan diprediksi.

2.2.2. *Machine Learning*

Machine learning atau biasa disebut dengan pembelajaran mesin adalah salah satu bagian dari *artificial intelligence* (AI) yang berfokus pada pengembangan

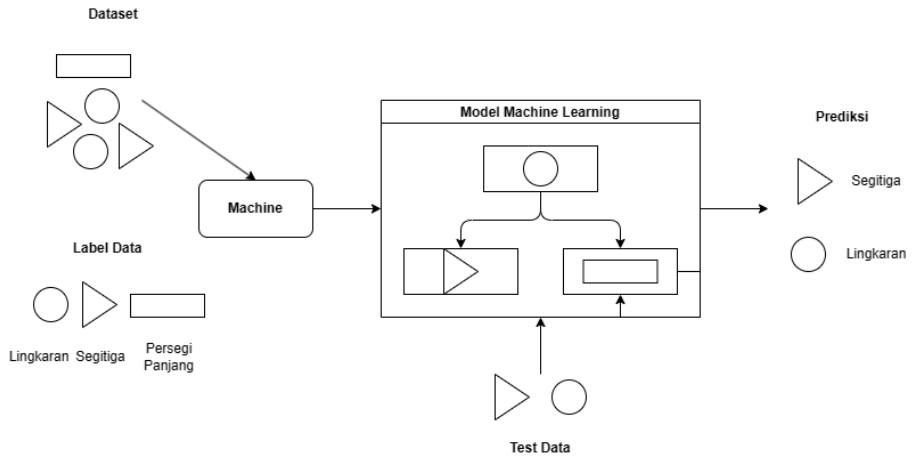
sebuah sistem untuk mampu belajar sendiri layaknya manusia tanpa harus diprogram berulang kali [22]. Menurut Mitchel [23], bisa dikatakan *machine learning* jika satu program komputer telah melakukan pembelajaran dari sebuah pengalaman (*experience*) terhadap tugas (*task*) dan mengukur hasil peningkatan kinerja (*performance measure*). Fokus utama dari *machine learning* adalah membangun aplikasi komputer yang dapat mempelajari data secara mandiri dan mampu membuat model yang siap digunakan untuk memecahkan sebuah kasus atau masalah [24]. *Machine learning* terdiri dari tiga sub bidang yang terlihat pada Gambar 2.1, yaitu *supervised learning*, *unsupervised learning*, dan *reinforcement learning*.



Gambar 2.1 Sub Pembagian *Machine learning*

Supervised learning merupakan sub bagian dari *machine learning* yang digunakan untuk mengolah kumpulan data berlabel [25]. Contoh penerapan dari *supervised learning* adalah klasifikasi dan regresi.

Klasifikasi merupakan proses pengelompokan objek yang mempunyai kesamaan karakteristik atau ciri kedalam beberapa kelas [3]. Proses klasifikasi dengan menerapkan *machine learning* dimulai dari inputan data berlabel yang akan diberikan pembelajaran menggunakan mesin dan mesin akan mendeteksi data – data baru (data uji / *testing*) ke dalam label atau kelas tertentu. Contoh pengaplikasian klasifikasi menggunakan *machine learning* terdapat pada Gambar 2.2.



Gambar 2.2 Alur Klasifikasi Menggunakan *Machine learning*

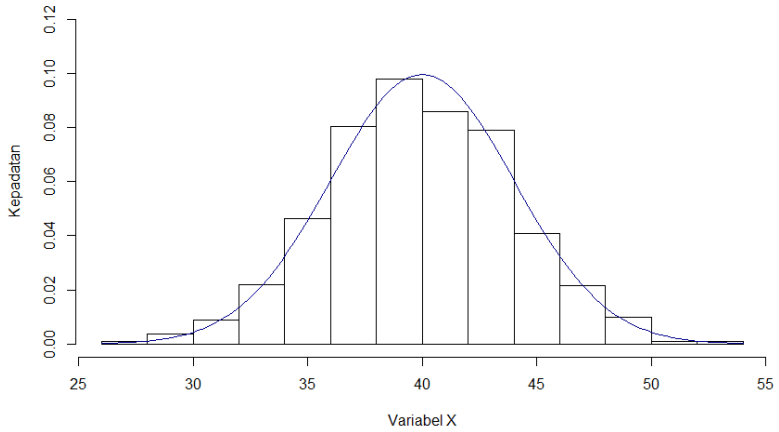
Unsupervised learning merupakan sub bagian pada *machine learning* yang digunakan untuk mengolah data tidak berlabel [24]. Contoh penerapannya pada kasus *clustering*. Sub bagian yang terakhir yaitu *reinforcement learning* yang pada prosesnya tanpa menggunakan inputan data, pada pembelajaran ini terdapat *agent* yang dapat mengamati lingkungan, memilih dan melakukan tindakan dan mendapatkan hasil untuk proses pembelajaran dan tindakan selanjutnya [24].

2.2.3. *Exploratory Data Analysis*

Exploratory Data Analysis (EDA) adalah salah satu tahapan pada analisis penelitian domain *Data Science*. *Exploratory data analysis* merupakan pendekatan analisis terhadap suatu data yang berguna untuk membuat dan mengetahui *summary* data. Tujuan utama *exploratory data analysis* adalah untuk memeriksa distribusi, *outliers*, dan tren data yang mengarah ke pengujian spesifik hipotesis yang dibuat [26]. untuk mengungkap wawasan yang dapat membantu memahami data dengan lebih baik. *Exploratory data analysis* berperan penting dalam mengidentifikasi pertanyaan yang relevan, merumuskan hipotesis, dan mempersiapkan data sebelum diterapkan pada model analisis yang lebih lanjut.

2.2.3.1. Distribusi Data

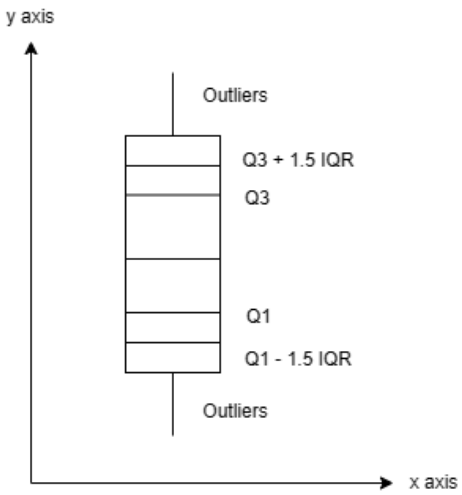
Terdapat bermacam – macam distribusi data pada statistika, seperti distribusi data normal dan tidak normal. Data dikatakan berdistribusi normal jika datanya simetris di sekitar nilai tengahnya. Gambar 2.3 memperlihatkan bahwa data berdistribusi normal.



Gambar 2.3 Contoh Data Berdistribusi Normal

2.2.3.2. *Outliers Data*

Sebuah data yang memiliki sifat dan karakteristik yang berbeda dari data pada umumnya dan kemunculan kejadian yang relatif sedikit dikatakan *outliers* [27]. *Outliers* bisa jauh lebih besar atau jauh lebih kecil daripada nilai-nilai lainnya, sehingga dapat menyebabkan distorsi pada analisis dan kesimpulan yang diambil dari data. Untuk melihat nilai *outliers*, biasanya menggunakan grafik boxplot.



Gambar 2.4 Data *Outliers*

Gambar 2.4 menunjukkan data yang terdeteksi *outliers* yaitu jika terletak di atas nilai *upper bound* dan di bawah nilai *lower bound.* Perhitungan nilai *upper bound* dan *lower bound* terdapat pada persamaan 2.2 dan 2.3.

$$Lower\ Bound = \text{Q1} - (1.5 \text{ x IQR}) \qquad (2.2)$$

$$Upper\ Bound = \text{Q3} + (1.5 \text{ x IQR}) \qquad (2.3)$$

Keterangan:

Q1 = Percentile ke-25

Q3 = Percentile ke-75

IQR = Q3 – Q1

2.2.3.2. Korelasi Fitur

Korelasi fitur adalah suatu ukuran statistik yang mengukur sejauh mana dua variabel atau fitur dalam dataset berhubungan satu sama lain. Terdapat beberapa jenis korelasi seperti korelasi *pearson, spearman, kendall*, dan lainnya. Korelasi *pearson* digunakan untuk digunakan untuk mengukur sejauh mana hubungan linier antara dua variabel kontinu. Perhitungan korelasi *pearson* pada persamaan 2.4.

$$r = \frac{n\Sigma xy - (\Sigma x)(\Sigma y)}{\sqrt{\{n\Sigma x^2 - (\Sigma x)^2\}\{n\Sigma y^2 - (\Sigma y)^2\}}} \qquad (2.4)$$

Keterangan:

N = banyaknya pasangan x dan y

Σx = Total jumlah dari variabel x

Σy = Total Jumlah dari Variabel y

Σx2= Kuadrat dari total jumlah variabel y

Σy2= Kuadrat dari total jumlah variabel y

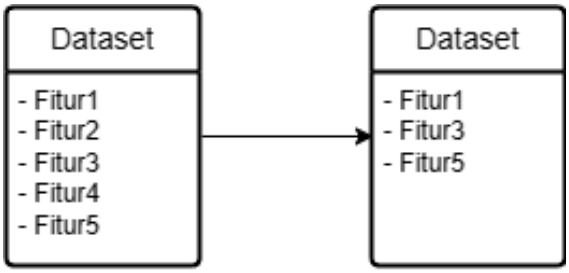
Σxy= Hasil perkalian dari total jumlah variabel x dan variabel y.

2.2.4. *Feature Engineering*

Feature atau fitur merupakan representasi numerik dari data mentah [20]. *Feature engineering* adalah proses mengubah, memodifikasi, atau membuat fitur baru dari data mentah (*raw data*) untuk meningkatkan kualitas dalam pemodelan. Tujuan utamanya adalah untuk meningkatkan performa dan akurasi model dengan membersihkan data dan memilih fitur data yang mempunyai pengaruh besar. *Feature engineering* merupakan langkah penting dalam membangun model *machine learning* karena fitur yang tepat dapat meringankan kesulitan pemodelan, sehingga menghasilkan akurasi yang lebih baik dan berkualitas [28]. Terdapat beberapa teknik yang termasuk *feature engineering* untuk data tabular, seperti *feature selection, standardization,* dan *oversampling*.

2.2.4.1. *Feature Selection*

Feature selection merupakan teknik menyeleksi fitur yang akan digunakan dalam membuat model *machine learning*. Contoh penerapan *feature selection* terdapat pada Gambar 2.5.

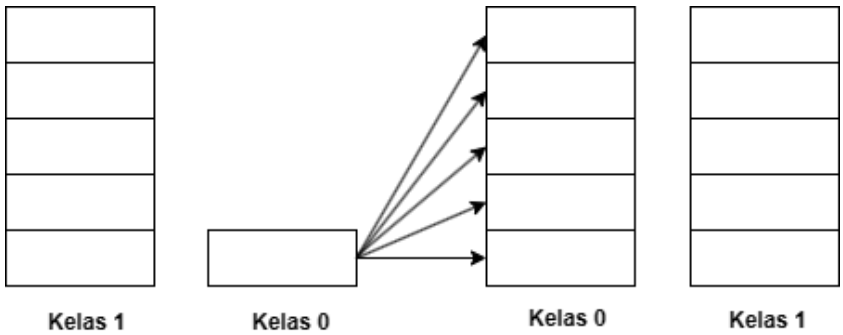


Gambar 2.5 *Feature Selection*

Untuk mendapatkan fitur – fitur yang relevan pada model *Random Forest*, dilihat dari nilai *feature importance*. *Feature importance* yaitu proses untuk melihat sejauh mana fitur-fitur berkontribusi terhadap pemodelan. *Random Forest* merupakan salah satu algoritma yang dapat dicari nilai *feature importance*nya. Semakin tinggi nilai *feature importance*, semakin besar kontribusinya fitur terhadap model. Dalam prediksi risiko kredit sendiri, mengabaikan fitur yang tidak relevan dapat meningkatkan akurasi model, mengurangi durasi waktu *training* atau pelatihan, dan menurunkan biaya komputasi saat menjalankan beberapa model *machine learning*.

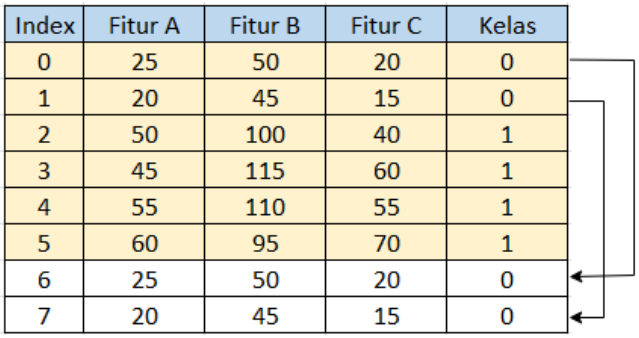
2.2.4.2. *Oversampling*

Terdapat beberapa teknik *sampling* seperti *undersampling* dan *oversampling*. *Oversampling* merupakan teknik menyeimbangkan data dengan memperbanyak data pada kelas minoritas [31]. Contoh *oversampling* terlihat pada Gambar 2.6.



Gambar 2.6 *Oversampling*

Terdapat beberapa teknik *oversampling* seperti *Random Oversampling*, SMOTE, BorderLine SMOTE, KMeans SMOTE, SVM SMOTE, *Adasyn*, dan SMOTE-NC. SMOTE adalah singkatan dari *Synthetic Minority Over-sampling Technique*. SMOTE merupakan teknik oversampling yang menghasilkan data sintetik dengan menghasilkan *instance* baru di antara garis yang menghubungkan dua fitur [32]. *Random oversampling* merupakan teknik *sampling* dengan menambahkan jumlah data secara acak pada kelas minoritas [33]. Contoh penerapan *random oversampling* terdapat pada Gambar 2.7.



Index	Fitur A	Fitur B	Fitur C	Kelas
0	25	50	20	0
1	20	45	15	0
2	50	100	40	1
3	45	115	60	1
4	55	110	55	1
5	60	95	70	1
6	25	50	20	0
7	20	45	15	0

Gambar 2.7 Contoh *Random Ovesampling*

Gambar 2.7 menunjukkan bahwa index ke 0 dan 1 diduplikat secara random ke index 6 dan 7 sehingga jumlah kelas minoritas (0) menjadi sama dengan kelas mayoritas (1).

2.2.4.3. Standardisasi

Standardisasi atau *standardization* merupakan metode yang dilakukan untuk membuat beberapa variabel data memiliki rentang nilai yang sama, tidak ada yang terlalu besar maupun terlalu kecil [29]. Salah satu teknik standardisasi yaitu *Robust Scaler*, merupakan salah satu teknik optimasi untuk mentransformasikan suatu nilai pada data dengan menggunakan *median* dan *quartiles* [30]. Standardisasi menggunakan *Robust Scaler* dijelaskan pada persamaan

$$x' = \frac{\mathrm{x} - \mathrm{median\ (x)}}{\mathrm{Q3} - \mathrm{Q1}} \qquad (2.5)$$

Notasi x’ merupakan hasil standardisasi menggunakan *Robust Scaler*, Q3 – Q1 merupakan hasil *Interquartile Range*. Menggunakan persamaan 2.5, maka contoh perhitungan *Robust Scaler* terdapat pada Tabel 2.2.

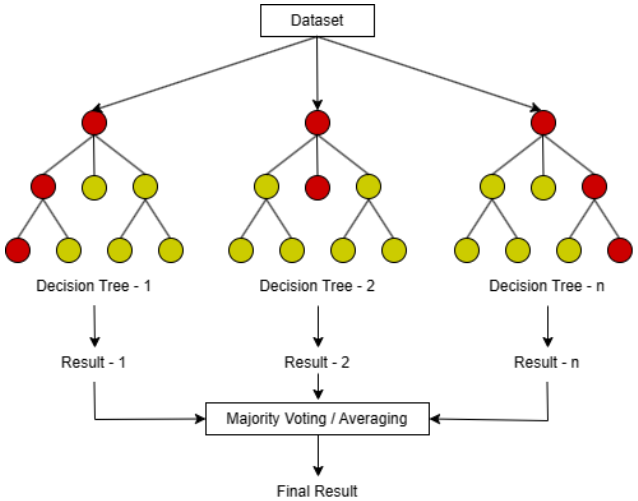
Tabel 2.2 Contoh Hasil Standardisasi Menggunakan *Robust Scaler*

Sebelum standardisasi		Setelah standardisasi	
Fitur A	Fitur B	Fitur A	Fitur B
12000	25	-1.14285714	-1.38461538
15000	50	-0.28571429	0.15384615
20000	75	1.14285714	1.69230769
17000	45	0.28571429	-0.15384615

Dari Tabel 2.2 menunjukkan bahwa sebelum dilakukan standardisasi, nilai pada kolom Fitur A (12000) dan Fitur B (25) memiliki rentang skala nilai yang berbeda. Setelah menjalankan proses standardisasi, Fitur A dan Fitur B akan memiliki rentang skala data yang serupa.

2.2.5. *Random Forest*

Random Forest adalah salah satu algoritma *machine learning* yang bekerja dengan cara mengkombinasikan beberapa algoritma *Decision Tree* [15]. *Random Forest* termasuk algoritma *ensemble* yang artinya algoritma ini membangun model di atas banyak model sehingga mendorong model tersebut mendapatkan hasil lebih baik [24]. Dengan menggunakan banyak *Decision Tree,* algoritma ini dapat memperoleh prediksi yang akurat dan meminimalkan *overfitting*. Selain kuat terhadap *overfitting*, *Random Forest* juga memiliki kinerja yang lebih baik jika dibandingkan dengan beberapa *machine learning* lain seperti *Support Vector Machine* (SVM) dan *Discriminant Analysis*. Gambar 2.8 menjelaskan alur kerja algoritma *Random Forest*.



Gambar 2.8 Cara Kerja Algoritma *Random Forest*

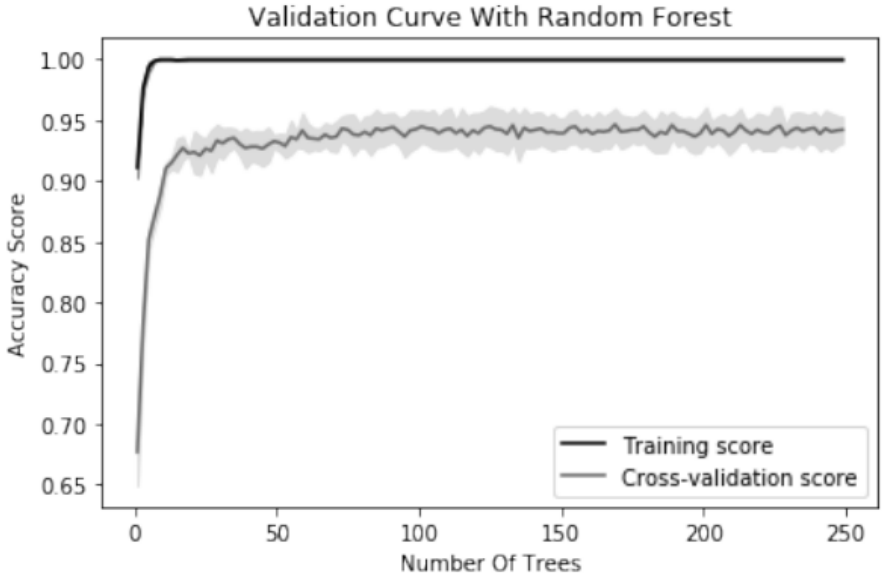
Cara kerja algoritma *Random Forest* dimulai dari pemilihan acak sampel data yang digunakan, kemudian membuat *Decision Tree* untuk setiap sampel data. Dilanjutkan dengan *voting* setiap hasil prediksi menggunakan nilai modus (pada model klasifikasi), sehingga hasil akhirnya atau hasil prediksi didapatkan dari hasil *voting* terbaik.

2.2.6. Hyperparameter Tuning

2.2.6.1. Parameter *Random Forest*

Hyperparameter Tuning adalah tugas menemukan parameter paling optimal untuk algoritma *machine learning* yang akan diterapkan pada data. Pada *machine learning*, optimasi menggunakan *hyperparameter tuning* sangat diperlukan untuk mendapatkan ukuran kinerja (*Accuracy* atau AUC) yang terbaik. Terdapat beberapa parameter yang bisa diubah dan diterapkan pada algoritma *Random Forest*, seperti jumlah pohon (*n_estimators*), kriteria (*criterion*), maksimal kedalaman pohon (*max_depth*), jumlah fitur yang dipertimbangkan (*max features*), jumlah sampel minimum untuk pembentukan *node* baru (*min_samples_split*), dan jumlah sampel minimum pada daun (*min_samples_leaf*).

N_estimators atau jumlah pohon pada *Random Forest* dapat ditentukan untuk mendapatkan akurasi model paling optimal.



Gambar 2.9 Percobaan *Jumlah n_estimators*

Gambar 2.9 menunjukkan percobaan *tuning parameter* pada *n_estimators*

atau jumlah pohon mulai dari 0 sampai 250 yang menghasilkan nilai akurasi berbeda – beda. Nilai akurasi mengalami kenaikan dan penurunan di setiap jumlah pohon yang berbeda sehingga jumlah pohon keputusan pada model ini harus diperhatikan.

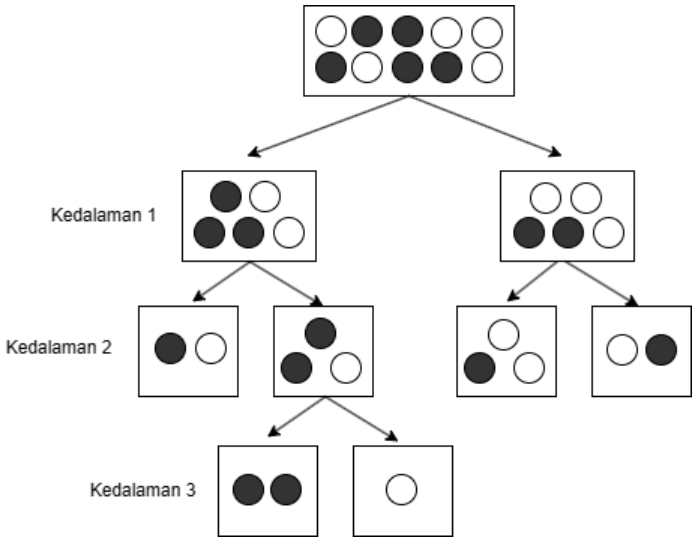
Criterion atau kriteria pembuatan model *Random Forest* terbagi menjadi dua yaitu *Gini* dan *Entropy*. Perhitungan menggunakan kriteria *gini* lebih cepat jika dibandingkan dengan *entropy* dikarenakan tidak menggunakan perhitungan logaritma. Rumus perhitungan *gini* dan *entropy* sesuai persamaan 2.2 dan 2.3

$$Gini = 1 - \sum_{i=1}^{n} (pi)2 \qquad (2.2)$$

$$Entropy = \sum_{i=1}^{c} - pi\ Log2\ pi \qquad (2.3)$$

Notasi p menunjukkan probabilitas *entropi* dan *pi* merupakan probabilitas suatu objek dikategorikan ke kelas tertentu.

Max_depth atau kedalaman pohon dapat ditentukan berdasarkan data yang akan digunakan. Gambar 2.10 menunjukkan kedalaman pohon sebuah model *Random Forest* yang ditentukan.



Gambar 2.10 Kedalaman pohon

Gambar 2.10 menggambarkan kedalaman pohon yang diatur yaitu 3 sehingga terdapat 3 kedalaman pohon yang terbentuk. Jumlah kedalaman pohon atau *max_depth* dapat diatur sesuai kebutuhan.

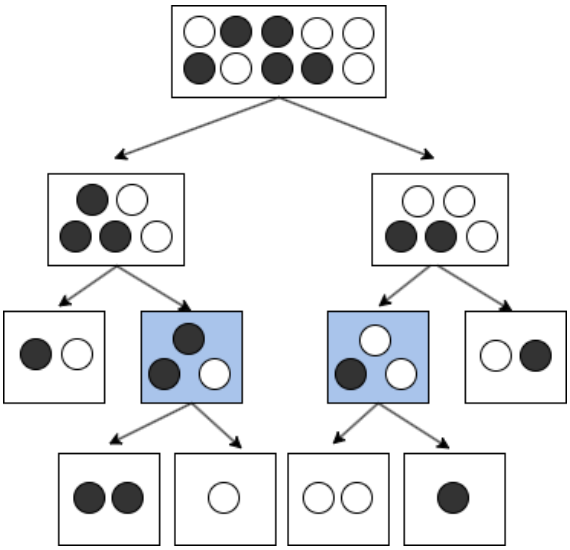
Selain itu terdapat *features* pada *Random Forest* yang bisa ditentukan jumlah maksimalnya atau bisa disebut *max_features*. Jumlah *max_features* yang digunakan pada umumnya yaitu berupa nilai *sqrt*, *log2*, dan *none* yang berupa nilai integer atau *float*. Secara *default* yang digunakan yaitu *sqrt* yang terdapat pada persamaan 2.4, 2.5, dan 2.6.

$$max_{features}(sqrt) = sqrt(n_{features}) \quad (2.4)$$

$$max_{features}(log2) = log2(n_{features}) \quad (2.5)$$

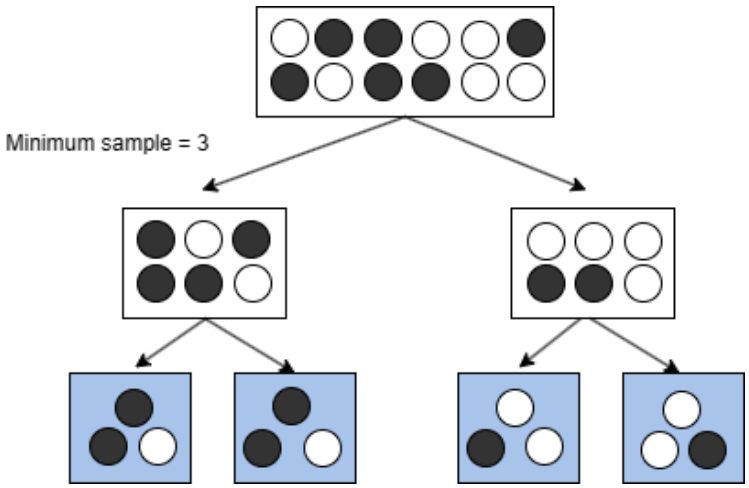
$$max_{features}(none) = n_{features} \quad (2.6)$$

nfeatures pada persamaan 2.4, 2.5, dan 2.6 merupakan jumlah fitur pada data. Parameter yang bisa ditentukan selanjutnya yaitu *min_sample_split* atau jumlah minimal *split* pada *node* yang ditunjukkan pada Gambar 2.11.



Gambar 2. 11 Contoh *Min_Samples_Split*

Gambar 2.11 menunjukkan bahwa jumlah *min_sample_split* yang ditetapkan yaitu 3 (kotak berwarna biru). Jika jumlah sampel di bawah 3 maka tidak dapat dilakukan *split*. Parameter lain yang bisa ditentukan pada model *Random Forest* yaitu *min_samples_leaf* atau jumlah minimal *node* pada daun.



Gambar 2.12 Contoh *Min_Samples_Leaf*

Pada Gambar 2.12 menunjukkan bahwa pada jumlah sampel pada daun telah menenuhi kondisi minimal yaitu 3 sampel (kotak berwarna biru). Dapat dikatakan bahwa di suatu daun tidak boleh terdapat sampel di bawah 3.

2.2.6.2. Grid Search

Untuk memudahkan dalam penggunaan *hyperparameter tuning*, terdapat algoritma optimasi seperti *Grid Search CV*, *Randomized Search CV*, dan *Bayesian CV* [11]. Algoritma *Grid Search* tepat untuk diterapkan saat ingin melakukan pencarian parameter secara eksplisit dan memeriksa semua kombinasi secara sistematis. Dibandingkan *Random Search, Grid Search* dianggap lebih sistematis karena mencoba semua kombinasi, sehingga cocok untuk ruang pencarian yang kecil dan terdefinisi dengan baik.

2.2.6.3. Random Search

Random Search tepat digunakan jika ingin mencari solusi dengan cepat karena algoritma ini secara acak mengambil parameter yang akan digunakan. Algoritma *Random Search* mempunyai kelebihan pada proses *training* dikarenakan secara acak memilih sekumpulan kombinasi parameter untuk dievaluasi. Namun kekurangan dari *Random Search* adalah kurang menjelajahi semua parameter dikarenakan hanyak diambil beberapa parameter secara acak saja.

2.2.7. Matriks Konfusi

Terdapat beberapa cara pengukuran keakuratan model *machine learning*.

Beberapa metrik pengukuran yang digunakan seperti *accuracy*, *confusion matrix*, dan *classification report. Accuracy* merupakan sebuah matrik evaluasi yang digunakan untuk mengukur keakuratan model dalam melakukan prediksi atau klasifikasi yang secara perhitungan ditunjukkan persamaan (2.7).

$$Accuracy = \frac{TP\ + TN}{TP + TN + FP + FN} \qquad (2.7)$$

Keterangan:

a. *True Positive* (TP) merupakan jumlah observasi yang benar – benar termasuk ke dalam kelas positif dan diprediksi benar oleh model.
b. *True Negative* (TN) merupakan jumlah observasi yang benar – benar termasuk ke dalam kelas negatif dan diprediksi dengan benar oleh model.
c. *False Positive* (FP) merupakan jumlah observasi yang diprediksi sebagai bagian dari kelas positif oleh model, tetapi sebenarnya observasi tersebut termasuk dalam kelas negatif.
d. *False Negative* (FN) merupakan jumlah observasi yang diprediksi sebagai bagian dari kelas negatif oleh model, tetapi sebenarnya observasi tersebut termasuk dalam kelas positif.

Untuk membuktikan nilai akurasi model yang baik, dapat dilihat dari nilai *confusion matrix* seperti Gambar 2.13.

		Predicted Class	
		Normal	Attack
Actual Class	Normal	True Negative (TN)	False Positive (FP)
	Attack	False Negative (FN)	True Positive (TP)

Gambar 2.13 *Confusion Matrix*

Model dikatakan baik jika nilai TN dan TP > FN dan FP. Menggunakan tabel *confusion matrix* ini, akan terlihat jumlah hasil prediksi benar dan salah sehingga dapat membuktikan secara akurat kebaikan model.

2.2.8. Classification Report.

Classification Report merupakan salah satu metrik evaluasi kinerja model pada *machine learning* berbasis klasifikasi. Pada hasil *classification report* terdapat empat nilai yaitu *precision, recall, f1- score,* dan *support*.

a. *Precision,* mengukur sejauh mana model memberikan hasil yang benar secara proporsional terhadap hasil yang dinyatakan sebagai positif, ditunjukkan persamaan (2.8).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \qquad (2.8)$$

b. *Recall*, mengukur sejauh mana model dapat mengidentifikasi secara benar semua kasus positif yang ada, seperti pada persamaan 2.9.

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad (2.9)$$

c. *F1 – Score,* memberikan pengukuran yang seimbang antara precision dan recall. *F1-score* berguna ketika ingin memperhatikan precision dan recall secara bersamaan, serta dan mencari nilai rata-rata yang optimal dari keduanya. Perhitungan *F1-Score* terdapat pada persamaan 2.10.

$$F1 - Score = \frac{Precision \times Recall}{Precision + Recall} \qquad (2.10)$$

d. *Support* merupakan keterangan informasi tentang banyaknya data yang ada untuk setiap kelas dalam data yang dievaluasi.

2.2.9. Kurva AUC (*Area Under the Curve*)

Kurva AUC (*Area Under the Curve*) adalah alat evaluasi untuk mengukur kinerja model klasifikasi. Kurva AUC menggambarkan hubungan antara tingkat *true positive* (*sensitivity*) dan tingkat *false positive* (1-*specificity*) pada berbagai *threshold*. Nilai AUC berkisar antara 0 hingga 1, nilai 1 menunjukkan performa sempurna dan nilai 0,5 menunjukkan performa model setara dengan perbandingan acak. Semakin tinggi nilai AUC, semakin baik model dalam membedakan antara kelas positif dan negatif. Analisis kurva AUC untuk memperoleh wawasan tentang kinerja model dan memilih *threshold* yang tepat untuk pengambilan keputusan berdasarkan *trade-off* antara *true positive* dan *false positive.*