BAB II
TINJAUAN PUSTAKA

2.1 *Decision Tree*

Decision Tree (DT) adalah sebuah algoritma dengan skema pohon keputusan. DT adalah sebuah alat pendukung yang menggunakan model pohon keputusan dan kemungkinan-kemungkinan hasilnya. DT sudah sering digunakan untuk sistem klasifikasi dan prediksi. Metode DT ini mengubah suatu data yang sangat besar menjadi sebuah pohon yang dapat merepresentasikan aturan yang dihasilkan. DT sangat berguna untuk mengeksplorasi data, dan menemukan suatu hubungan tersembunyi antara sejumlah variabel dengan variabel target [14][15].

Classification and Regression Trees (CART) merupakan salah satu jenis dari algoritma DT. Algoritma *CART* adalah jenis algoritma pohon keputusan yang dapat digunakan untuk klasifikasi dan regresi. Algoritma CART bekerja dengan cara membagi data menjadi subset-subset yang berdasarkan nilai dari sebuah atribut dalam satu waktu. Tujuan dari pembagian subset-subset ini adalah untuk memaksimalkan homogenitas dari setiap subset-subset hasil dari pembagian data. Proses yang dilakukan ini akan terus diulang sampai kriteria yang diinginkan terpenuhi, seperti sudah mencapai batas kedalaman dari pohon keputusan atau mencapai jumlah minimum dari jumlah sampel yang ada pada setiap *leaf node* [16].

Algoritma CART saat digunakan untuk sistem klasifikasi menentukan setiap kelas dari *leaf node* berdasarkan mayoritas data dari *training data* yang ada pada *node* itu, sedangkan saat Algoritma CART digunakan untuk sistem regresi maka CART membuat kalkulasi nilai *mean* dari target atribut untuk *training data* pada setiap *leaf node*. Algoritma CART ini memiliki beberapa kelebihan. Kelebihan itu adalah mudah dimengerti manusia, kesederhanaan sistemnya, dan kemampuan untuk menangani data atribut dalam jenis atribut kategori dan attribut *continuous*. Algoritma CART rawan terhadap *overfitting* jika pohon keputusanya terlalu dalam dan data yang ada terkontaminasi dengan *noise*. *Overfitting* dapat diselesaikan dengan beberapa cara. Cara itu adalah *pruning*, *regularization*, dan

ensembling. *Pruning* adalah memotong *leaf node* pada pohon keputusan yang tidak meningkatkan performa akurasi pada *validation* set. *Regularization* adalah memberikan sistem *penalty* ke *splitting criterion* untuk membuat pohon keputusan menjadi lebih simpel. *Ensembling* adalah menggabungkan beberapa pohon keputusan, seperti dengan menggunakan teknik *bagging* atau *boosting* untuk meningkatkan akurasi dan ketahanan dari sebuah model.

2.2 *Naive Bayes*

Naive Bayes (NB) adalah sebuah algoritma SL. NB adalah sebuah algoritma yang simpel tetapi sangat efektif. NB akan mengklasifikasikan suatu objek berdasarkan teori *Naive Bayes*. Algoritma NB akan mengasumsikan bahwa suatu atribut pada suatu kelas tidak berhubungan dengan atribut lainya. NB menggunakan teori probabilitas yang berfokus pada nilai parameter dari suatu kelas [16]. Sistem klasifikasi NB bekerja dengan cara mengklasifikasikan data baru sebagai nilai dari sebuah target label A, jika nilai target label B diketahui [17]. Data baru itu kemudian diklasifikasikan ke kelas yang memiliki probabilitas data yang lebih besar. Teori dari NB dapat dituliskan dalam bentuk persamaan seperti Persamaan (2.1) dari teori NB, yaitu:

Teori NB

$$A|B = \frac{P(B|A) * P(A)}{P(B)} \tag{2.1}$$

Keterangan pada Persamaan (2.1) yaitu, A|B = peluang kejadian A jika kejadian B diketahui, P(B|A) = peluang kejadian B jika A diketahui, P(A) = peluang kejadian A saat B tidak diketahui, P(B) = peluang kejadian B saat A tidak diketahui. Hasil dari A|B bisa didapatkan dengan mengalikan P(B|A) dengan P(A) yang dibagi dengan P(B).

2.3 *Support Vector Machine*

Support Vector Machine (SVM) merupakan teknik klasifikasi yang bisa digunakan dalam *machine learning*. Cara kerja SVM adalah dengan memisahkan data menjadi kelas-kelas yang berbeda dengan menggunakan *plane* atau *hyperplane*. SVM bekerja dengan mencari garis yang dapat memisahkan kelas-

kelas yang ada, dengan memaksimalkan margin atau jarak antara garis atau bidang dan titik-titik yang terdekat dari masing-masing kelas [18]. Teori dari SVM dapat dituliskan dalam bentuk persamaan seperti Persamaan (2.2) dari teori SVM, yaitu:

Teori SVM

$$y = sign(w * x + b) \quad (2.2)$$

Keterangan pada Persamaan (2.2) yaitu, y = adalah kelas prediksi yang diketahui, w = adalah vektor bobot, x = adalah vektor fitur dari data yang ingin diprediksikan, b = b adalah bias, sign= *sign* adalah fungsi signum, yaitu fungsi yang akan menghasilkan argumen positif atau 0, dan nilai -1 jika hasilnya negatif. Perhitungan w dikalikan x ditambah b kemudian dikalikan dengan *sign* perlu dilakukan untuk melakukan klasifikasi data baru.

2.4 *K-Nearest Neighbour* (KNN)

KNN adalah sebuah algoritma *supervised learning non-parametic* yang dapat digunakan untuk tugas klasifikasi dan regresi. Sistem klasifikasi KNN ini bekerja dengan cara mengelompokan data baru ke kelas terbanyak pada k tetangga terdekat (k adalah *integer* positif). Jika k= 2, maka data baru akan diklasifikasikan ke kelas dengan dua tetangga terdekatnya [19]. Tahapan dalam KNN dapat dilakukan dengan langkah berikut:

1. Melakukan perhitungan setiap data pada *training* data dan *testing* data pada jarak *Euclidean* masing-masing data.
2. Melakukan pengurutan jarak terdekat dari hasil perhitungan setiap data.
3. Menentukan k tetangga terdekat.
4. Menentukan hasil klasifikasi dari mayoritas k tetangga terdekat.

2.5 Python

Python adalah bahasa pemrograman yang dibuat pertama kalinya oleh Guido van Rossum yang saat itu seorang peneliti di Universitas Amsterdam, Belanda. Python dirilis ke publik untuk yang pertama pada awal dekade 1990. Lisensi Python yang *open source* dipegang oleh Python Software Foundation (PSF), lembaga yang didirikan oleh Guido dan rekan-rekannya di awal 2000-an.

Pembuatan *machine learning* tentunya memerlukan bahasa pemrograman yang mendukung proses *machine learning*. Salah satu Bahasa pemrograman yang mendukung hal ini adalah Python. Python merupakan bahasa pemrograman berorientasi objek yang dapat digunakan dalam banyak pengembangan perangkat lunak terutama dalam bidang *data science*. Python lebih populer digunakan karena didukung oleh beragam *open library* yang berkaitan dengan penelitian di bidang kecerdasan buatan [7].

A. *Artificial Intelligence*

Artificial intelligence atau yang biasa dikenal sebagai AI dan juga kecerdasan buatan adalah jalan untuk membuat mesin agar dapat berpikir dan memiliki kemampuan intelegensi. Mesin fisik AI menggunakan algoritma tertentu agar dapat memahami situasi dan kondisi sebagaimana yang manusia pikirkan [20].

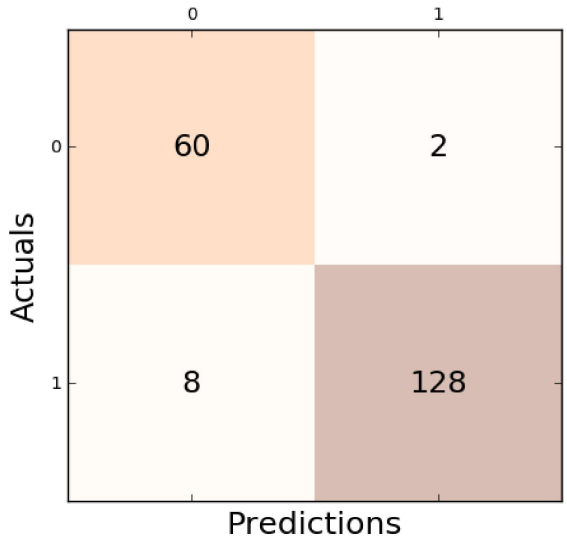
B. *Machine Learning*

Proses pembelajaran mesin atau yang biasa dikenal sebagai *machine learning* merupakan sebuah metode untuk membuat mesin dapat berpikir layaknya manusia. *Machine learning* merupakan cabang dari ilmu kecerdasan buatan atau *artificial intelligence*. Pembelajaran pada *machine learning* ini dimaksudkan agar mesin dapat menyelesaikan masalah. Masalah yang dapat diselesaikan oleh *machine learning* biasanya bersifat spesifik agar dapat memberikan jaminan solusi terbaik dalam penyelesaiannya [21]. *Machine learning* model sendiri merupakan sebuah algoritma yang telah dilakukan siklus pelatihan pada data tertentu, sehingga telah siap untuk dilakukan uji data untuk mengetahui keakuratan dari algoritma yang telah dibangun.

2.6 *Confusion Matrix*

Confusion matrix dalam *machine learning* dikenal juga dengan nama *error matrix*. *Confusion matrix* adalah sebuah susunan tabel yang secara spesifik memvisualisasikan performa dari suatu algoritma, yang biasanya algoritma ini merupakan algoritma *Supervised Learning* (SL). *Confusion matrix* dalam

unsupervised learning disebut dengan nama *matching matrix*. Setiap baris pada *confusion matrix* ini merepresentasikan sebuah data dalam kelas yang sebenarnya, sedangkan kolom pada *confusion matrix* merepresentasikan kelas hasil prediksi dari algoritma yang digunakan (bisa juga sebaliknya yaitu setiap baris merepresentasikan kelas prediksinya dan setiap kolom merepresentasikan kelas yang sebenarnya). Nama dari *confusion matrix* ini diambil karena *confusion matrix* ini memudahkan untuk melihat apakah sistem yang dibuat ini salah atau benar berdasarkan data asli dan data hasil prediksi yang ada [22]. Gambar dari *confusion matrix* dapat dilihat pada Gambar 2.1 di bawah ini.



Gambar 2.1 *Confusion Matrix*

Berdasarkan Gambar 2.1 di atas, dapat dilihat bahwa *confusion matrix* di atas ini menggunakan pengaturan baris pada *confusion matrix* untuk data yang sebenarnya, dan kolom pada *confusion matrix* ini untuk menampilkan data hasil prediksi dari algoritma yang digunakan.

Evaluasi pada algoritma sistem klasifikasi *machine learning* merupakan cara untuk melihat sebaik apa performa suatu sistem model klasifikasi yang sudah dibuat. Tujuan dari evaluasi sistem ini adalah untuk melihat sebaik apa performa suatu model klasifikasi dalam memprediksi kelas data yang benar. Ada beberapa metode evaluasi yang umum digunakan untuk mengevaluasi model *machine learning*, yaitu *accuracy*,

precision, *recall*, *confusion matrix* dan *F1score*. *Accuracy* adalah cara evaluasi model dengan mengukur seberapa tepat suatu model *machine learning* dalam mengklasifikasikan keseluruhan *testing* data dengan benar. persamaan dari *Accuracy* dapat dilihat pada Persamaan (2.3) di bawah ini [23].

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (2.3)$$

Berdasarkan Persamaan (2.3) *true positive* (TP) adalah jumlah *testing* data positif yang berhasil diklasifikasikan dengan benar dan *true negative* (TN) adalah jumlah *testing* data negatif yang berhasil diklasifikasikan dengan benar. *False negative* (FN) adalah jumlah *testing* data positif yang tidak diklasifikasikan dengan benar dan *false positive* (FP) adalah jumlah *testing* data negatif yang tidak diklasifikasikan dengan benar. *Precision* dalam model evaluasi digunakan untuk mengukur seberapa presisi model dalam mengidentifikasikan data yang benar-benar positif dari semua data yang diprediksi positif. *Precision* mengukur seberapa tepat data yang diprediksikan positif dari data yang aslinya benar positif. Persamaan dari *precision* dapat dilihat pada Persamaan (2.4) di bawah ini [23].

$$Precision = \frac{TP}{TP+FP} \quad (2.4)$$

Berdasarkan Persamaan (2.4) *precision* dapat dihitung dengan membagi nilai TP dengan nilai TP yang ditambah dengan nilai FP. Dalam kasus *imbalanced data*, *precision* bisa digunakan untuk melihat performa dari suatu model pada kelas data minoritas dan kelas mayoritas sehingga bisa diketahui bagaimana performa model *machine learning* tersebut saat memprediksi kelas minoritas dan mayoritas. *Recall* merupakan metode evaluasi yang memberikan gambaran seberapa baik suatu model klasifikasi dalam mengklasifikasikan semua kelas data yang diprediksikan positif dari semua kelas data yang benar-benar positif. Persamaan dari *recall* dapat dituliskan dalam Persamaan (2.5) di bawah ini [23].

$$Recall = \frac{TP}{TP+FN} \quad (2.5)$$

Berdasarkan persamaan (2.5) di atas, nilai dari *recall* dapat dicari dengan membagi nilai TP dengan nilai TP ditambah nilai FN. *Recall* bisa juga digunakan untuk mengevaluasi performa suatu model algoritma yang datanya memiliki *imbalanced* data. *F1score* adalah suatu metode evaluasi yang mengukur *harmonic* dari nilai *recall* dan *precision*. *F1score* ini juga memberikan gambaran tentang

kinerja suatu model secara keseluruhan. Persamaan dari *F1score* dapat dilihat pada Persamaan (2.6) di bawah ini [23].

$$F1score = \frac{2 \times precision \times recall}{precision + recall} \quad (2.6)$$

Berdasarkan Persamaan (2.6) di atas, *F1score* dapat dicari dengan cara dua dikalikan dengan *precision* dan *recall*, kemudian dibagi dengan nilai dari *precision* ditambah *recall*. *F1score* dalam prakteknya memberikan sebuah gambaran keseimbangan antara *precision* dan *recall* [23].

Selain metode evaluasi yang dijelaskan di atas, masih ada metode evaluasi lainya yaitu matthews *correlation coefficient* (MCC). MCC merupakan suatu metode evaluasi yang sangat cocok digunakan untuk evaluasi model *machine learning* terutama pada sistem *binary classification*. MCC disebut merupakan metode evaluasi performa model yang paling baik karena MCC ini hanya akan menghasilkan nilai yang tinggi jika model *machine learning* tersebut memiliki performa yang bagus dari keempat nilai pada *confusion matrix* (TP, TN, FN, FP). MCC ini disebutkan sebagai model yang sesuai dan paling baik untuk mengevaluasi performa model yang memiliki *class imbalanced*. Persamaan perhitungan dari MCC dapat dilihat pada Persamaan (2.7) di bawah ini [24].

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)\ (TP + FN)\ (TN + FP)\ (TN + FN)}} \quad (2.7)$$

Berdasarkan Persamaan (2.7) di atas, dapat dilihat bahwa nilai MCC menggunakan semua nilai dari TP, TN, FP, dan FN. Hasil dari MCC ini juga memiliki nilai yang artinya dapat dilihat pada Tabel 2.1 di bawah ini.

Tabel 2.1 Rentang Nilai Skor MCC

Matthews *Correlation Coefficient*	
Nilai skor MCC	Keterangan
-1	Performa prediksi model terburuk
0	Performa prediksi model standar yang menghasilkan nilai yang acak
1	Performa prediksi model yang sempurna

Berdasarkan Tabel 2.1 di atas, dapat dilihat rentang nilai yang menggambarkan performa suatu model *machine learning* dengan menggunakan metode evaluasi MCC. Semakin mendekati 1 maka performa model itu semakin

baik. Sebaliknya semakin dekat ke angka -1 maka performa model itu semakin buruk. Nilai 0 jika didapatkan berarti performa model itu merupakan model yang tidak istimewa atau bisa disebut hanya mendapat nilai 50 dari 100 [24][25].

2.7 Kajian Pustaka

Penelitian ini berlandaskan dari penelitian-penelitian terdahulu, baik dari landasan teori, metode atau teknik penelitian yang digunakan, maupun jenis penelitiannya. Berikut ini lima penelitian yang menjadi landasan dari penelitian yang sedang dilakukan, yaitu:

Penelitian pertama, penelitian ini dilakukan di sebuah universitas di negara Turki. Algoritma yang dibandingkan pada penelitian ini adalah *random forests*, *Neural Network*, *support vector machine*, *logistic regression*, NB, dan KNN. Hasil yang didapatkan menyatakan bahwa *random forest* dan *neural network* memiliki performa terbaik dengan akurasi sebesar 74,6%. [8].

Penelitian kedua, penelitian kedua ini berfokus pada prediksi performa siswa. Penelitian ini melakukan prediksi performa siswa pada SMAN 3 Ambon dengan menggunakan tiga teknik *data mining*, yaitu KNN, DT, dan NB. Hasil penelitian ini menunjukan bahwa DT memiliki akurasi tertinggi dengan nilai sebesar 99,6%. Hasil ini lebih baik dibanding KNN dan NB [10].

Penelitian ketiga, penelitian ketiga melakukan penelitian untuk membandingkan performa DT dan NB. Penelitian ini menggunakan *data mining* untuk untuk membandingkan kecepatan dan akurasi dari sistem klasifikasi untuk menentukan penerima beasiswa. Hasil yang didapat dari penelitian menunjukan bahwa DT memiliki performa lebih baik dengan akurasi sebesar 96,40%. Lebih baik dari NB yang mendapatkan akurasi sebesar 95,11% [11].

Penelitian keempat, mirip dengan penelitian ketiga, penelitian keempat membuat penelitian untuk memprediksi performa mahasiswa pada Universitas Duhok, provinsi Duhok, daerah Kurdistan, Iraq. Berdasarkan penelitian ini, digunakan lima algoritma yaitu DT, SVM, KNN, NB, dan *Random Forest* (RF). Hasil yang didapatkan menunjukan bahwa DT memiliki performa yang paling baik [13].

Penelitian kelima, penelitian ini dilakukan di sekolah dasar negeri 4 Trimulyo sekampung. Penelitian ini membuat prediksi performa siswa dengan menggunakan algoritma DT C4.5. Hasil Penelitian yang didapatkan adalah sebuah sistem prediksi performa siswa dengan akurasi sebesar 94,43% [14].

Berdasarkan beberapa hasil penelitian di atas, dapat diketahui bahwa *machine learning* dapat digunakan untuk membuat prediksi performa siswa. Namun penelitian di atas belum ada yang melakukan prediksi performa siswa pada tingkat SMK, maka dari itu penelitian ini menggunakan *machine learning* untuk memprediksi performa siswa pada tingkat SMK.