BAB 2
DASAR TEORI

2.1 KAJIAN PUSTAKA

Terdapat beberapa penelitian yang dapat digunakan sebagai acuan pada penelitian ini. Salah satunya penelitian yang dilakukan oleh Vrushali C. Waikar, Sheetal Y. Thorat, Ashlesha A. Ghute, Priya P. Rajput dan Mahesh S. Shinde pada tahun 2020 dengan judul penelitian "*Crop Prediction based on Soil Classification using Machine Learning with Classifier Ensembling*" [4]. Penelitian ini bertujuan untuk membangun sistem klasifikasi tanah dan melakukan prediksi tanaman menggunakan *machine learning.* Algoritma yang digunakan pada penelitian ini meliputi *Support Vector Machine, Bagged Tree, Adaboost, Naive Bayes,* dan *Artificial Neural Network*. Algoritma *Artificial Neural Network* menghasilkan akurasi 92%, Algoritma *Support Vector Machine* menghasilkan akurasi 98.5%, Algoritma *Bagged Tree* menghasilkan akurasi 88%, Algoritma *Naive Bayes* menghasilkan akurasi 76% dan Algoritma *AdaBoost* menghasilkan akurasi 66%.

Sk Al Zaminur Rahman, Kaushik Chandra Mitra dan S.M. Mohidul Islam pada tahun 2018 melakukan penelitian dengan judul "*Soil Classification using Machine Learning Methods and Crop Suggestion Based on Soil Series*" [5]. Penelitian ini bertujuan untuk membuat model yang cocok untuk mengklasifikasikan berbagai jenis data tanah bersama dengan saran tanaman yang cocok untuk daerah tertentu di Upazilla Bangladesh. Pada penelitian ini digunakan beberapa algoritma *machine learning* meliputi *K-Nearest Neighbor*, *Bagged Trees* dan *Support Vector Machines* digunakan untuk klasifikasi tanah. Penelitian ini menunjukkan algoritma *Bagged tree* dan K-NN memiliki nilai akurasi yang bagus namun pada klasifikasi tanah algoritma *Support Vector Machines* memperoleh akurasi tertinggi dengan nilai akurasi sebesar 94,95%. Hasil tersebut diikuti oleh algoritma K-NN dengan akurasi 92,93% dan algoritma *Bagged tree* dengan akurasi 90,91%.

Penelitian berikutnya dilakukan oleh Madhuri Shripathi Rao, Arushi Singh, N.V. Subba Reddy dan Dinesh U Acharya pada tahun 2021 dengan judul penelitian "*Crop Prediction Using Machine Learning*" [9]. Penelitian ini dilakukan dengan

tujuan menemukan model terbaik untuk prediksi tanaman, yang dapat membantu petani memutuskan jenis tanaman yang akan ditanam berdasarkan kondisi iklim dan nutrisi yang ada di dalam tanah. Pada penelitian ini digunakan beberapa algoritma *machine learning* meliputi *K -Nearest Neighbor*, *Decision Tree*, dan *Random Forest Classifier*. Penelitian ini menyatakan algoritma *Random Forest* memberikan akurasi tertinggi di antara ketiganya dengan akurasi sebesar 99,32%.

Penelitian dengan judul "*Comparative Analysis of Machine Learning Algorithms in The Study of Crop and Crop yield Prediction*" [10] yang dilakukan oleh S Bharath, Yeshwanth S, Yashas B L dan Vidyaranya R Javalagi pada tahun 2020 bertujuan untuk menunjukkan penggunaan berbagai algoritma *machine learning* untuk memprediksi hasil panen. Prediksi hasil panen dilakukan berdasarkan nilai masukkan berkaitan dengan curah hujan, pH dan suhu. Penelitian ini menunjukkan nilai akurasi dari algoritma *Support Vector Machine* sebesar 92.6%, *Decision Tree Classifier* sebesar 99.87%, K-NN *Classifier* sebesar 99.73% dan *Random Forest Classifier* sebesar 81.07%.

Penelitian selanjutnya dilakukan oleh Shilpa Mangesh Pande, Dr. Prem Kumar Ramesh, Anmol, B.R Aishwarya, Karuna Rohilla dan Kumar Shaurya pada 2021 dengan judul penelitian "*Crop Recommender System Using Machine Learning Approach*" [11]. Penelitian ini bertujuan untuk membuat suatu sistem yang dapat memandu petani untuk memaksimalkan hasil panen serta menyarankan tanaman yang paling menguntungkan untuk wilayah tertentu. Algoritma *machine learning* yang digunakan pada penelitian ini meliputi *Support Vector Machine*, *Artificial Neural Network*, *Random Forest*, *Multivariate Linear Regression* dan *K-Nearest Neighbor*. Dari kelima algoritma yang digunakan, algoritma *Random Forest* menunjukkan hasil terbaik dengan nilai akurasi sebesar 95%.

2.2 DASAR TEORI

2.2.1 Pertanian

Pertanian memainkan peran penting dalam ekonomi global. Tekanan pada sistem pertanian akan meningkat dengan berlanjutnya ekspansi populasi manusia [12]. Indonesia merupakan negara agraris dimana pertanian merupakan tumpuan

utama perekonomian nasional. Dibandingkan dengan negara-negara Asia lainnya, Indonesia merupakan negara agraris terbesar ketiga setelah India dan China. [1]. Indonesia merupakan salah satu negara agraris yang mayoritas penduduknya bermata pencaharian dari pertanian [2].

Sektor pertanian Indonesia merupakan bagian penting dari perekonomian negara yang menempati posisi pertama, disusul oleh sektor perdagangan dan konstruksi [3]. Sektor pertanian merupakan salah satu sektor utama sebagai penghasil pangan. Mengingat Indonesia merupakan negara agraris yang sebagian besar penduduknya adalah petani, maka pertanian menjadi sangat penting karena berkontribusi terhadap pencapaian tujuan pembangunan ekonomi nasional [2].

2.2.2 Nutrisi Tanah

Tanah adalah sistem kehidupan yang sangat beragam dan kompleks. Tanah itu sendiri dapat dipandang sebagai organisme hidup, karena merupakan habitat bagi tumbuhan, hewan, dan mikroorganisme yang semuanya saling terkait. Ketersediaan atau kekurangan nutrisi tanah berpengaruh pada sistem pertanian meliputi hasil produksi, kualitas tanaman dan keuntungan [13].

Dalam pertanian tanah adalah hal utama dan mendasar. Pengujian tanah sangat berpengaruh pada bidang pertanian. Produksi dan kualitas tanaman sangat bergantung pada tanah. Pengujian tanah sangat penting karena memberikan informasi tentang semua nutrisi yang ada di tanah seperti Ca (kalsium), K (kalium), dan N (Nitrogen) [4].

Unsur hara pada tanah sangat diperlukan tumbuhan sebagai sumber nutrisi dalam menunjang pertumbuhannya. Tumbuhan memerlukan berbagai kombinasi yang tepat dari berbagai nutrisi tanah untuk dapat tumbuh, berkembang dan bereproduksi. Nutrisi yang terlalu sedikit maupun terlalu banyak pada tumbuhan dapat menyebabkan masalah pada pertumbuhannya. Tanah merupakan lapisan permukaan bumi yang berfungsi sebagai tempat tumbuh dan berkembangnya tanaman serta menyuplai kebutuhan air dan udara. Secara kimiawi tanah berfungsi sebagai penyuplai hara atau nutrisi senyawa organic maupun anorganik sederhana serta unsur esensial seperti N, P, K, Ca, Mg, S, Cu, Zn, Fe, Mn, B, Cl. Tanah adalah media sekaligus pemasok unsur hara untuk pertumbuhan tanaman. Unsur hara

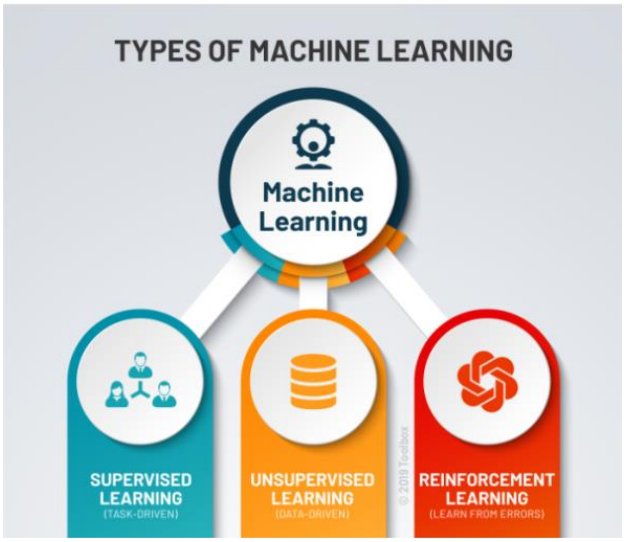
esensial tersebut haruslah selalu tersedia dan selalu dalam keadaan seimbang. Setiap jenis tanaman memiliki takaran unsur hara dengan besaran yang berbeda-beda. Tidak tepatnya pemberian unsur hara dapat menyebabkan tanaman tidak dapat tumbuh serta bereproduksi secara maksimal. Pertumbuhan dan mutu dari suatu tanaman sangat dipengaruhi oleh kadar nutrisi yang tersedia pada tanah. Namun apabila tersedia dalam konsentrasi yang tinggi, unsur-unsur tersebut dapat merusak tanaman [14].

2.2.3 *Machine Learning* (ML)

Machine Learning (ML) atau Mesin Pembelajar merupakan cabang dari *Artificial Intelligent* (AI) yang berfokus pada data (*learn from data*). ML bertujuan untuk mengembangkan sistem yang dapat belajar sendiri tanpa harus berulang kali diprogram oleh manusia [15].

Umumnya ML digunakan untuk membantu sistem untuk dapat belajar dari pengalaman sebelumnya sedangkan AI digunakan untuk pengambilan keputusannya. Tujuan utama dari kecerdasan buatan adalah untuk meningkatkan peluang keberhasilan sedangkan ML lebih berfokus kepada akurasi daripada peluang keberhasilan dari suatu sistem. Tujuan lain dari kecerdasan buatan adalah untuk merangsang kecerdasan alami untuk memecahkan masalah yang kompleks. Di sisi lain, tujuan lain dari pembelajaran mesin adalah untuk memeriksa data untuk tugas-tugas tertentu guna memaksimalkan kinerja mesin. AI mengembangkan sistem untuk meniru manusia sehingga sistem dapat merespon dan melakukan sesuatu, sedangkan ML membantu algoritma agar dapat bekerja secara otomatis [16].

ML umumnya digunakan bersama dengan AI tetapi merupakan bagian dari AI. ML mengacu pada sistem AI yang dapat belajar sendiri berdasarkan algoritma. Sistem yang menjadi lebih pintar dan lebih pintar dari waktu ke waktu tanpa campur tangan manusia adalah ML. *Deep Learning* (DL) adalah ML yang diterapkan pada kumpulan data besar. Sebagian besar pekerjaan AI melibatkan ML karena perilaku cerdas membutuhkan pengetahuan yang cukup [17].



Gambar 2.1 Jenis ML[18]

Algoritma ML dapat dapat diklasifikasikan kedalam beberapa jenis. Secara garis besar algoritma ML meliputi :

1. *Supervised learning*

   Supervised learning merupakan algoritma ML yang menghasilkan fungsi untuk melakukan pemetaan data *input* ke data *output* yang diinginkan [19]. *Supervised learning* melakukan pembelajaran dengan menggunakan sekumpulan sampel data yang memiliki label [20]. *Supervised learning* dapat digunakan untuk memecahkan masalah klasifikasi maupun masalah regresi. Masalah klasifikasi merupakan suatu masalah yang variabel *output*nya berupa kategori, seperti merah atau biru. Sedangkan masalah regresi merupakan suatu masalah ketika variabel *output*nya bersifat *continuous,* seperti perkiraan curah hujan [21].

2. *Unsupervised learning*

   Unsupervised learning merupakan algoritma ML yang tidak memiliki data *output* target (label) karena algoritma ini hanya memodelkan satu set data *input* dan mencari pengelompokan (*clustering*) pada data tersebut [19]. *Clustering* dilakukan karena tidak adanya data *output* target yang diinginkan sehingga algoritma ini dapat membedakan dengan benar pada sekumpulan data. *Unsupervised learning* digunakan untuk memecahkan masalah *clustering* dan masalah asosiasi. Masalah asosiasi adalah masalah terkait aturan yang digunakan untuk menggambarkan sebagian besar data yang ada, seperti orang yang membeli A juga cenderung membeli B. Sedangkan masalah pengelompokan

atau *clustering* merupakan masalah terkait tempat untuk menemukan pengelompokan yang melekat dalam data, seperti melakukan pengelompokan pelanggan yang didasari pada perilaku pelanggan dalam melakukan pembelian [21].

3. *Reinforcement learning*

   Reinforcement learning merupakan algoritma ML yang belajar melalui interaksi dengan lingkungan sehingga mendapatkan umpan balik tentang keakuratan responnya [19]. *Reinforcement learning* merupakan algoritma ML yang dapat secara otomatis melakukan evaluasi untuk dapat meningkatkan efisiensi sistem berdasarkan pendekatan berbasis lingkungan [20]. Jenis algoritma ini dapat secara otomatis menentukan kebijakannya yang diperoleh melalui coba-coba dan terus berinteraksi dengan lingkungan yang sifatnya dinamis [21]. Jenis algoritma ML ini didasarkan pada *reward* atau *penalty* dengan tujuan menggunakan wawasan yang diperoleh oleh sistem untuk meningkatkan *reward* dan meminimalisir resiko. Algoritma ini cocok untuk mengoptimalkan efisiensi operasional dari sistem canggih seperti robotika dan *autonomous driving* [20].

2.2.4 Klasifikasi

Klasifikasi merupakan bagian dari *supervised learning*. Klasifikasi bertujuan untuk melakukan prediksi secara akurat suatu kategori pada data yang tidak diketahui dalam setiap kasus [22]. Berikut ini adalah jenis klasifikasi yang banyak digunakan dalam penelitian :

1. Klasifikasi *biner*

   Klasifikasi *biner* mengacu pada tugas klasifikasi yang memiliki dua label kelas seperti "benar dan salah" atau "ya dan tidak".

2. Klasifikasi *Multiclass*

   Klasifikasi *multiclass* secara tradisional, ini mengacu pada tugas klasifikasi yang memiliki lebih dari dua label kelas. Contoh dari klasifikasi *multiclass* adalah pengklasifikasian dari *dataset* yang menyediakan data dari berbagai jenis serangan jaringan di NSL-KDD di mana kategori serangan diklasifikasikan menjadi empat label kelas, seperti DoS (*Denial of Service Attack*), U2R (*User to Root Attack*), R2L (*Root to Local Attack*), dan *Probing Attack* [20].

2.2.5 *K-Nearest Neighbor*

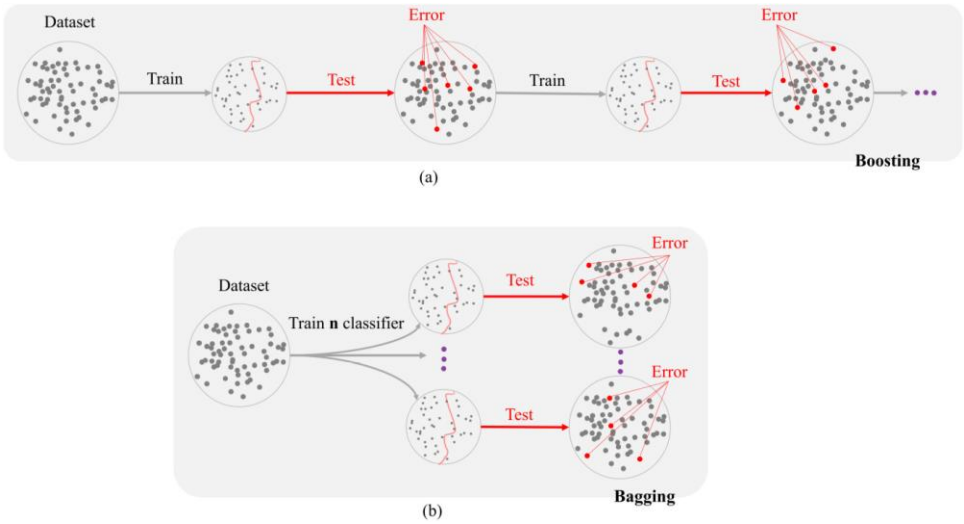
Algoritma *K-Nearest Neighbor* merupakan algoritma ML yang termasuk ke dalam *supervised learning* dan dapat digunakan untuk memecahkan permasalahan terkait masalah klasifikasi maupun masalah regresi [23]. Algoritma *K-Nearest Neighbor* menyimpan semua kemungkinan dan mengklasifikasikan data yang masuk tergantung pada seberapa mirip mereka dengan data aktual. Hal ini menunjukkan bahwa algoritma *K-Nearest Neighbor* dapat dengan cepat mengklasifikasikan *instance* baru ke dalam kategori yang ditentukan secara tepat [9].

Algoritma *K-Nearest Neighbor* melakukan analisis pembelajaran berdasarkan masalah atau sekumpulan data yang diberikan [24]. *K-Nearest Neighbor* bekerja berdasarkan jarak minimum dari *instance* kueri ke sampel pelatihan untuk menentukan tetangga K-terdekat [23]. Nilai K yang digunakan pada proses klasifikasi harus bernilai ganjil [25]. Pada algoritma *K-Nearest Neighbor*, nilai 'K' mewakili jumlah nilai data tetangga terdekat. Berdasarkan 'K', yaitu jumlah tetangga terdekat, keputusan dibuat oleh algoritma *K-Nearest Neighbor* untuk mengklasifikasikan *dataset* yang diberikan [24].

2.2.6 *Random Forests*

Algoritma *Random Forests* merupakan algoritma yang pendekatan belajarnya menggunakan metode *ensemble* yang dikembangkan oleh Breiman dengan tujuan memecahkan masalah klasifikasi dan regresi. *Ensemble leraning* merupakan skema ML yang dipergunakan untuk meningkatkan akurasi dengan mengintegrasikan beberapa model ML untuk memecahkan masalah yang sama. Metode *ensemble learning* yang banyak digunakan adalah *boosting* dan *bagging*. *Boosting* merupakan proses dalam membangun model yang berurutan, setiap model yang dibagun menggunakan *boosting* memiliki karakteristik untuk memperbaiki kesalahan yang terjadi pada urutan sebelumnya pada urutan tersebut. Namun, masalah yang sering dihadapi *boosting* adalah modelnya yang *overfitting*. Sedangkan *bootstrap agregat* atau yang dikenal sebagai *bagging* merupakan jenis lain dari *ensemble learning* yang dirancang untuk meningkatkan stabilitas dan

akurasi model. Dengan demikian *bagging* diakui lebih kuat terhadap permasalahan *overfitting* dibandingkan dengan menggunakan pendekatan *boosting* [26].



Gambar 2.2 Metode Pembelajaran. (a) Boosting. (b) Bagging[26]

Random Forests merupakan salah satu algoritma ML yang sangat popular dan kuat. Algoritma *Random Forests* bekerja dengan cara membuat banyak pohon keputusan yang dikumpulkan untuk melakukan klasifikasi yang didasari *majority vote* [27]. *Random Forests* membuat sejumlah pohon keputusan yang dilatih pada berbagai bagian dari rangkaian pelatihan yang sama dengan tujuan untuk meningkatkan tingkat klasifikasi dan untuk mengatasi masalah *overfitting*. *Random Forests* melakukan pemilihan atribut secara acak untuk membuat sejumlah pohon keputusan dengan atribut yang berbeda. Pada algoritma *decision tree*, *test data* diuji hanya pada satu pohon yang dibangun. Sedangkan *Random Forests*, *test data* diuji pada semua pohon yang dibangun kemudian akan dilakukan klasifikasi dengan mengambil hasil pemilihan terbanyak dari pohon yang dibangun [22].

2.2.7 *eXtreme Gradient Boosting*

Gradient boosting termasuk ke dalam kategori *supervised learning* yang dibuat berdasarkan *decision tree*. *Gradient boosting* dapat digunakan untuk melakukan klasifikasi [28]. *Gradient boosting* merupakan sebuah pendekatan di mana model baru dibuat berdasarkan kesalahan dari model sebelumnya yang kemudian menambahkannya bersama-sama untuk membuat prediksi akhir [29]. *Gradient boosting* membangun *decision tree* atau pohon keputusan yang didasari

dari peningkatan struktur pohon pada pembelajaran yang lemah untuk memperbaiki kesalahan pohon serta mencegah terjadinya *overfitting* [30].

Algoritma *eXtreme Gradient Boosting* merupakan sebuah algoritma yang ditemukan oleh Friedman. *eXtreme Gradient Boosting* merupakan algoritma yang digunakan untuk melakukan prediksi atau melakukan klasifikasi dengan basis pohon keputusan [31]. *eXtreme Gradient Boosting* adalah salah satu algoritma yang menggunakan metode *boosting* dengan kumpulan pohon keputusan dengan ketentuan pembangunan pohon berikutnya bergantung pada pohon keputusan sebelumnya [32]. *eXtreme Gradient Boosting* dapat digunakan untuk memecahkan masalah *overfitting* [33]. *eXtreme Gradient Boosting* dikembangkan dengan optimasi 10 kali lebih cepat dibandingkan *gradient boosting* lainnya [31].

2.2.8 *Support Vector Machine*

Support Vector Machine merupakan algoritma ML yang diperkenalkan pertama kali pada tahun 1970-an oleh Vapnik dan kelompoknya [26]. SVM adalah salah satu algoritma yang digunakan untuk melakukan klasifikasi dan saat ini banyak dikembangkan serta diterapkan [34]. SVM merupakan bagian dari *supervised learning* yang membuat fungsi pemetaan berupa fungsi klasifikasi dengan membuat *hyperplane* dengan *margin* maksimal [35].

Cara kerja SVM yaitu dengan mendefinisikan batas antara dua kelas dengan jarak maksimal dari data yang terdekat. Untuk mendapatkan batas maksimal antar kelas, dibentuk sebuah garis pemisah atau *hyperplane* terbaik untuk *input space* yang diperoleh dengan mengukur *margin hyperlane* dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperlane* dengan titik terdekat dari tiap kelas. Titik terdekat inilah yang disebut sebagai *support vector* [36].

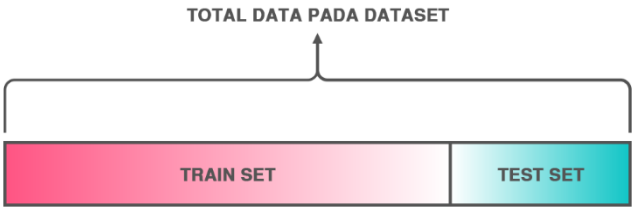
2.2.9 *Overfitting* dan *Underfitting*

Overfitting dan *underfitting* merupakan dua masalah utama yang terjadi pada pada saat melakukan *training* [37]. *Overfitting* adalah keadaan dari suatu model yang menunjukkan kinerja baik pada tahap *training* namun memiliki kinerja yang buruk pada tahap *testing* pada saat melakukan pengujian data baru. *Overfitting* dapat terjadi karena model yang dibuat terlalu banyak melakukan pembelajaran

pada *dataset training.* Sedangkan *underfitting* adalah keadaan dari suatu model yang menunjukkan kinerja buruk pada tahapan *training* maupun *testing. Underfitting* terjadi karena model yang dibuat kesulitan dalam melakukan penyesuaian performa pada *dataset* yang digunakan. Faktor yang dapat menyebabkan *underfitting* salah satunya adalah terlalu sederhananya model yang dibuat ataupun model yang dibuat memiliki regulasi terlalu banyak [38].

2.2.10 *Training* dan *Testing*

Proses *training* dan *testing* merupakan dua faktor penting yang mempengaruhi keberhasilan ML [39]. Pada *supervised learning*, data *training* dan data *testing* merupakan dua jenis data yang wajib ada. Hal tersebut dikarenakan kedua data tersebut merupakan karakteristik dari *supervised learning* [40].



Gambar 2.3 Ilustrasi Pembagian Data *Training* dan Data *Testing*[41]

Data *training* dapat diartikan sebagai sekumpulan data dengan atribut label atau kelas yang digunakan oleh ML untuk melakukan proses pembelajaran atau dapat ML gunakan untuk mengenal karakteristik dari kumpulan data sehingga dapat menghasilkan suatu pola ataupun model data. Dengan algoritma matematis, model dibuat *fit* (cocok) dengan data *training. Fitting* merupakan kemampuan ML untuk menggeneralisasi data sesuai dengan data *training* [9]. Sedangkan data *testing* merupakan sekumpulan data dengan label atau kelas yang digunakan untuk menguji ketepatan dari pola atau model yang telah dibuat dalam melakukan klasifikasi data *testing*. Pada proses *testing*, atribut dari data *testing* akan disembunyikan pada saat model melakukan proses klasifikasi. Label akan dipergunakan untuk membandingkan hasil dari proses klasifikasi sebagai tolak ukur besar akurasi atau ketepatan dari model yang telah dibuat dalam melakukan proses klasifikasi [40]. Pembagian data *testing* dan data *training* menunjukkan hasil terbaik pada komposisi 20% untuk data *testing* dan 80% untuk data *training* [42].

2.2.11 *Library* Python

Library merupakan sekumpulan modul yang bentuk dari sejumlah kode yang dapat digunakan berulang kali dalam program yang berbeda. *Library* berisi modul yang dibentuk dari bahasa pemrograman C dengan fitur yang menyediakan akses ke fungsionalitas sistem seperti file I/O. Dengan adanya *library* dapat membuat proses pemrograman menjadi lebih sederhana dan cepat karena *programmer* tidak perlu menulis kode yang sama secara berulang kali [43]. Beberapa *library* python yang popular dan sering digunakan dalam pengembangan ML adalah sebagai berikut.

1. *Numerical* Python (NumPy)

Gambar 2.4 *Icon* NumPy[44]

Numerical Python (NumPy) merupakan proyek *open-source* yang dikembangkan dengan tujuan untuk mengaktifkan komputasi numerik dengan Python. NumPy merupakan *library* dasar yang digunakan dalam melakukan komputasi ilmiah dengan menggunakan Python. NumPy adalah *library* Python yang menyediakan objek array multidimensi, berbagai objek turunan (seperti array dan matriks) dan bermacam-macam operasi pada array, termasuk matematika, manipulasi *shape*, penyortiran, pemilihan, I/O, transformasi Fourier diskrit, aljabar linier dasar, operasi statistik dasar, simulasi acak, dan banyak lagi [44].

2. Pandas

Gambar 2.5 *Icon* Pandas[45]

Library Pandas untuk ML bersifat *open-source*. Pandas memudahkan dalam melakukan analisis data, manipulasi data dan pembersihan data. *Library* pandas mendukung berbagai jenis operasi seperti penyortiran, pengindeksan

ulang, iterasi, penggabungan, konversi data, visualisasi, agregasi, dan lain sebagainya [45].

3. Matplotlib

Gambar 2.6 *Icon* Matplotlib[46]

Library Matplotlib digunakan untuk membuat visualisasi data menggunakan bahasa pemrograman Python. *Library* Matplotlib bersifat *open-source*. *Library* Matplotlib bertujuan untuk melakukan *ploting* angka-angka ke dalam bentuk diagram lingkaran, histogram, *chart* dan lainnya [46].

4. *Scikit-Learn* (Sk*learn*)

Gambar 2.7 *Icon Scikit-Learn*[47]

Scikit-Learn (Sk*learn*) merupakan *library* python yang banyak menyediakan algoritma *unsupervised* maupun *supervised learning*. Sk*learn* bersifat *open-source*. Sk*learn* difokuskan pada ML *tools* termasuk algoritma matematika, statistik dan tujuan umum yang membentuk dasar bagi banyak teknologi ML. Sk*learn* menyediakan pilihan alat yang efisien untuk ML dan pemodelan statistik termasuk klasifikasi, regresi, *clustering* dan lainnya. Sk*learn* bekerja sama dengan *library* NumPy, SciPy dan Matplotlib [47].

2.2.12 *Confusion Matrix*

Confusion Matrix adalah alat pengukuran untuk menghitung kinerja atau tingkat kebenaran dari suatu proses klasifikasi. *Confusion Matrix* merupakan sebuah tabel yang menyatakan klasifikasi jumlah data *testing* yang benar dan jumlah data *testing* yang salah atau secara sederhananya tabel dari *confusion matrix*

memuat informasi dari jumlah data yang tepat diklasifikasikan dan jumlah data yang tidak tepat diklasifikasikan [48].

Tabel 2.1 *Confusion Matrix* [49]

	Classified Positive	*Classified Negative*
Actual Positive	*True Positive* (TP)	*False Negative* (FN)
Actual Negative	*False Positive* (FP)	*True Negative* (TN)

Penjelasan singkat dari sejumlah keterangan yang ada pada tabel 2.1 meliputi :

1. *True Positive*

   True Positive (TP) adalah jumlah data pada kelas aktual *positive* dengan hasil dari kelas klasifikasinya menunjukkan nilai *positive*. TP berisi jumlah data pada kelas aktual *positive* dan diklasifikasikan sebagai kelas aktual *positive* [50].
2. *False Negative*

   False Negative (FN) adalah jumlah data pada kelas aktual *positive* namun hasil dari kelas klasifikasinya menunjukkan nilai *negative*. FN berisi jumlah data pada kelas aktual *positive* namun diklasifikasikan sebagai bukan kelas aktual *positive* melainkan diklasifikasi sebagai kelas aktual *negative* [50].
3. *False Positive*

   False Positive (FP) adalah jumlah data pada kelas aktual *negative* namun hasil dari kelas klasifikasinya menunjukkan nilai *positive*. FP berisi jumlah data pada kelas aktual *negative* namun diklasifikasikan sebagai bukan kelas aktual *negative* melainkan diklasifikasi sebagai kelas aktual *positive* [50].
4. *True Negative*

   True Negative (TN) adalah jumlah data pada kelas aktual *negative* dengan hasil dari kelas klasifikasinya menunjukkan nilai *negative*. TN berisi jumlah data pada kelas aktual *negative* dan diklasifikasikan sebagai kelas aktual *negative* [50].

2.2.13 *Accuracy*

Accuracy merupakan pengujian yang perhitungannya didasari oleh tingkat kedekatan antara nilai prediksi atau nilai hasil dari proses klasifikasi dengan nilai

aktual secara keseluruhan [50]. *Accuracy* mengukur rasio dari jumlah prediksi yang benar untuk semua prediksi [22]. *Accuracy* pada klasifikasi *multiclass* dapat diukur menggunakan rumus yang ada pada persamaan berikut [51].

$$Accuracy = \frac{The\ number\ of\ true\ predictions}{The\ number\ of\ predictions} = \frac{The\ diagonal\ values\ on\ matrix}{All\ values\ on\ confusion\ matrix} \quad (2.1)$$

2.2.14 *Precision*

Precision adalah ukuran keakuratan dari suatu model yang telah dibuat dalam melakukan prediksi kelas *positive* atau setiap baris yang diprediksi sebagai kelas *positive* [28]. *Precision* dapat diartikan sebagai jumlah informasi relevan dari keseluruhan informasi yang ditemukan oleh model baik relevan maupun tidak [49]. *Precision* akan mengukur total semua prediksi yang ditandai sebagai positif oleh *classifier* [22].

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (2.2)$$

2.2.15 *Recall*

Recall adalah pengujian dengan membandingkan jumlah informasi relevan yang dihasilkan oleh model dengan seluruh informasi relevan pada suatu kumpulan informasi atau dapat diartikan *recall* mengukur total kelas *positive* yang diprediksi dengan benar oleh model sebagai kelas *positive* [48]. *Recall* adalah ukuran kebaikan dari suatu model yang dapat digunakan untuk melihat keakuratan pada suatu kelas pada kasus *dataset* yang tidak seimbang [32].

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (2.3)$$

2.2.16 *Google Colaboratory*

Google Colaboratory (Colab) merupakan suatu layanan yang berjalan diatas layanan *cloud* yang konsepnya berasa dari *Jupyter Notebooks.* Pihak Google telah

menciptakan layanan baru yang disebut sebagai *Colaboratory* yang digunakan untuk mengedukasi dan melakukan penelitian terkait ML.

Gambar 2.8 *Icon Google Colaboratory*[52]

Runtime colab sudah terintegrasi dengan beberapa *library* AI yang sering digunakan dalam penelitian serta colab menyediakan GPU yang kuat. Colab merupakan layanan yang dibentuk oleh pihak google yang sifatnya gratis dan dapat ditautkan ke akun *google drive*. Colab tersedia dalam python versi 2 maupun python versi 3. Colab menyediakan *library* penting yang digunakan dalam pengembangan ML dan AI seperti *TensorFlow*, Matplotlib, dan Keras [52].