BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian-penelitian sebelumnya dapat menjadi sumber masukan dan ide-ide dalam membuat proyek akhir ini. Beberapa penelitian terdahulu dapat menjadi pembanding yang dapat diambil kesimpulannya dalam mengerjakan proyek akhir ini.

Penelitian Pertama oleh Diniyatullah & Septyani (2024), dengan judul "Aplikasi Naive Bayes Classifier untuk Memprediksi Status Gizi Berdasarkan Data Antropometri Santri di Pondok Pesantren Darunnajah". Penelitian ini bertujuan memprediksi status gizi santri dengan memanfaatkan data antropometri berupa berat badan, tinggi badan, dan usia menggunakan algoritma *Naïve Bayes Classifier*. Aplikasi yang dikembangkan berupa aplikasi web berbasis *python* dengan *framework Flask* dan *Bootstrap*, yang dilengkapi fitur perhitungan otomatis Indeks Massa Tubuh (IMT), klasifikasi status gizi ke dalam kategori gizi kurang, normal, dan kelebihan berat badan/obesitas, serta fitur unggah data CSV untuk prediksi massal.

Penelitian Kedua dilakukan oleh Suwita (2024), dengan judul "Aplikasi Simak Gizi untuk Penderita Obesitas di Wilayah Kota Malang". Penelitian ini bertujuan meningkatkan pengetahuan remaja obesitas mengenai pola konsumsi, aktivitas fisik, diet rendah energi, pemilihan makanan jajanan, dan pemahaman label informasi gizi. Tidak ada model *machine learning* yang dipakai, melainkan aplikasi *android* berbasis edukasi. Aplikasi Simak Gizi dilengkapi fitur kalkulator gizi (IMT, Berat Badan Ideal, kebutuhan energi, protein, lemak, karbohidrat), pengingat pola makan, rekomendasi menu diet rendah kalori, informasi kandungan gizi, tips diet, aktivitas fisik, serta kuis/*edutainment* untuk meningkatkan kesadaran gizi pengguna.

Penelitian Ketiga dilakukan oleh Nur Azizah Eka Budiarti (2024), dengan judul "Klasifikasi Status Gizi Anak dengan Decision Tree Berdasarkan Data Stunting di Kabupaten Gowa". Penelitian ini adalah mengklasifikasikan status gizi anak usia 0–4 tahun berdasarkan data *stunting* dengan menggunakan algoritma

decision tree yang menghasilkan akurasi sebesar 90%. Model ini diimplementasikan dalam bentuk aplikasi *android* dengan fitur utama prediksi status gizi berdasarkan input berat badan, tinggi badan, usia, dan jenis kelamin. Hasil prediksi kemudian diklasifikasikan ke dalam enam kategori, yaitu gizi baik, gizi buruk, gizi kurang, gizi lebih, obesitas, dan risiko gizi lebih, sehingga aplikasi ini dapat membantu orang tua dalam memantau status gizi anak secara praktis.

Penelitian Keempat dilakukan oleh Oktaviani & Triana (2023), dengan judul "Perancangan Aplikasi BMI Calculator Untuk Memprediksi Tingkat Obesitas Pada Mahasiswa Dengan Metode K-Nearest Neighbor". Penelitian ini bertujuan untuk merancang aplikasi kalkulator BMI berbasis *mobile* yang dapat memprediksi tingkat obesitas pada mahasiswa. Penelitian memakai algoritma *K-Nearest Neighbor* untuk melakukan klasifikasi tingkat obesitas berdasarkan 17 variabel dengan 128 jumlah data. Hasil akurasi yang dihasilkan dari model KNN yang dipakai sebesar 47%, yang menandakan kualitas dataset yang digunakan kurang baik sehingga prediksi tidak akurat untuk jangka panjang. Penelitian ini juga menghasilkan desain aplikasi dengan fitur kalkulator BMI, *insight* berbasis data, serta infografis kesehatan untuk memberikan informasi dan rekomendasi kepada pengguna.

Tabel 2.1 Penelitian Terdahulu

Penulis	Judul	Metode	Kesimpulan
Diniyatullah & Septyani (2024)	Aplikasi Naive Bayes Classifier untuk Memprediksi Status Gizi Berdasarkan Data Antropometri Santri di Pondok Pesantren Darunnajah	*Naïve Bayes Classifier*	Aplikasi web berhasil memprediksi status gizi santri berdasarkan data antropometri dengan fitur hitung IMT, klasifikasi gizi, dan upload CSV. Model *machine learning* menunjukkan akurasi baik untuk pemantauan gizi.
Suwita (2024)	Aplikasi Simak Gizi untuk Penderita Obesitas di	(Tanpa ML)	Aplikasi Simak Gizi meningkatkan pengetahuan remaja obesitas mengenai pola

	Wilayah Kota Malang		konsumsi, aktivitas fisik, diet rendah energi, pemilihan makanan jajanan, serta label informasi gizi melalui berbagai fitur interaktif.
Nur Azizah Eka Budiarti (2024)	Klasifikasi Status Gizi Anak dengan Decision Tree Berdasarkan Data Stunting di Kabupaten Gowa	*Decision Tree*	Model *Decision Tree* dengan akurasi 90% mampu mengklasifikasikan status gizi anak usia 0–4 tahun ke enam kategori. Untuk aplikasi a*ndroid* sangat membantu orang tua memantau status gizi anak secara praktis.
Oktaviani & Triana (2023)	Perancangan Aplikasi BMI Calculator Untuk Memprediksi Tingkat Obesitas Pada Mahasiswa Dengan Metode K-Nearest Neighbor	*K-Nearest Neighbor* (KNN)	Aplikasi kalkulator BMI berbasis mobile dapat memprediksi tingkat obesitas mahasiswa, namun akurasi model KNN hanya 47% sehingga kurang efektif. Aplikasi dilengkapi fitur kalkulator BMI, insight data, dan infografis kesehatan.

Berdasarkan beberapa penelitian terdahulu, dapat disimpulkan bahwa masing-masing penelitian memiliki fokus yang berbeda, baik dari segi metode maupun implementasi sistem. Oleh karena itu, penelitian ini mencoba untuk membuat aplikasi *machine learning* dengan variabel yang lebih beragam sehingga diharapkan mampu menghasilkan estimasi tingkat obesitas yang lebih akurat serta dapat membantu pengguna melakukan deteksi dini risiko obesitas secara mandiri.

2.2 Landasan Teori

2.2.1 Obesitas

Kelebihan berat badan atau bisa disebut obesitas adalah penyakit multifaktorial yang disebabkan oleh penumpukan lemak yang berlebih. Secara fisiologis, obesitas didefinisikan sebagai suatu kondisi dengan akumulasi lemak yang tidak normal atau berlebihan pada jaringan adiposa yang dapat mengganggu kesehatan (Dianah dkk., 2022).

Penumpukan lemak ini terjadi karena ketidakseimbangan antara energi yang masuk kedalam tubuh dengan energi yang dikeluarkan tubuh. Selain karena itu, obesitas dapat disebabkan banyak faktor seperti perubahan keseimbangan energi termasuk dengan mengonsumsi makanan cepat saji, kurangnya aktivitas fisik, usia, jenis kelamin, status sosial ekonomi, dan faktor psikologis. Faktor lingkungan juga mendorong pola hidup tidak sehat (obesogenik), variasi genetik, ataupun karena pengaruh konsumsi obat-obatan (Engagement, Journal, Health, & Survey, 2021).

Akibat yang ditimbulkan obesitas dapat mempengaruhi kesehatan tulang dan reproduksi, juga meningkatkan resiko terkena diabetes tipe II dan penyakit jantung serta resiko terkena kanker tertentu. Selain masalah fisik, obesitas juga bisa mengganggu kenyamanan hidup sehari-hari, seperti susah tidur ataupun kurang leluasa untuk bergerak bebas (World Health Organization, 2025).

Untuk mengetahui kategori status gizi seseorang digunakan perhitungan BMI (*Body Mass Index*), merupakan perhitungan yang digunakan untuk menilai apakah seseorang memiliki berat badan yang sesuai dengan tinggi badannya (Mohajan & Mohajan, 2023). Berikut rumus hitung BMI :

$$BMI = \frac{Mass(kg)}{Height(m)^2}$$

Hasil dari perhitungan inilah yang dipakai untuk mengelompokkan kategori berat badan seseorang. Mengutip dari jurnal “Body Mass Index (BMI) is a Popular Anthropometric Tool to Measure Obesity Among Adults” oleh Mohajan & Mohajan (2023), kategori berat badan berdasarkan BMI dapat dilihat pada tabel berikut:

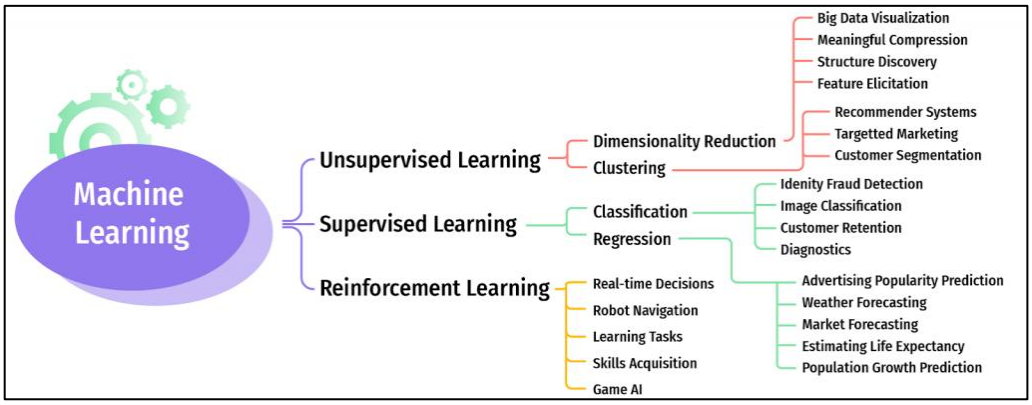
Tabel 2.2 Kategori Berat Badan

Kategori Berat Badan	BMI(kg/m^2)
Underweight (Severe thinness)	< 16.0
Underweight (Moderate thinness)	16.0–16.9
Underweight (Mild thinness)	17.0–18.4
Normal weight (Normal)	18.5 – 24.9
Overweight (Pre-obese)	25.0 – 29.9
Moderately obese (Class I-low risk)	30.0 – 34.9
Severely obese (Class II-moderate risk)	35.0 – 39.9
Very severely obese (Class III-high risk)	≥ 40.0

2.2.2 Machine Learning

Machine learning adalah cabang dari ilmu kecerdasan buatan (*Artificial Intelligence*) dan merupakan ilmu komputer yang berfokus pada algoritma dan data yang memungkinkan mesin belajar dan berkembang secara otomatis tanpa campur tangan manusia. Proses pembelajaran dimulai dengan memberi instruksi kepada mesin untuk mengetahui dan mempelajari pola yang diberikan, sehingga mesin dapat membuat keputusan yang baik berdasarkan contoh yang diberikan (Javid dkk., 2021).

Pada estimasi tingkatan obesitas, *machine learning* memungkinkan penggunaan data seperti berat badan, tinggi badan, pola makan, aktivitas fisik, dan gaya hidup untuk mengklasifikasikan tingkat obesitas seseorang. Oleh karena itu, *machine learning* memberikan potensi yang besar dalam meningkatkan kesadaran akan kondisi kesehatan, serta membantu masyarakat melakukan tindakan preventif lebih awal. Hal ini didukung oleh penelitian yang dilakukan oleh Verma & Verma (2022) bahwa *machine learning* dapat dimanfaatkan untuk meningkatkan efisiensi dan akurasi dalam pengambilan keputusan medis. Informasi lebih lanjut terkait dengan



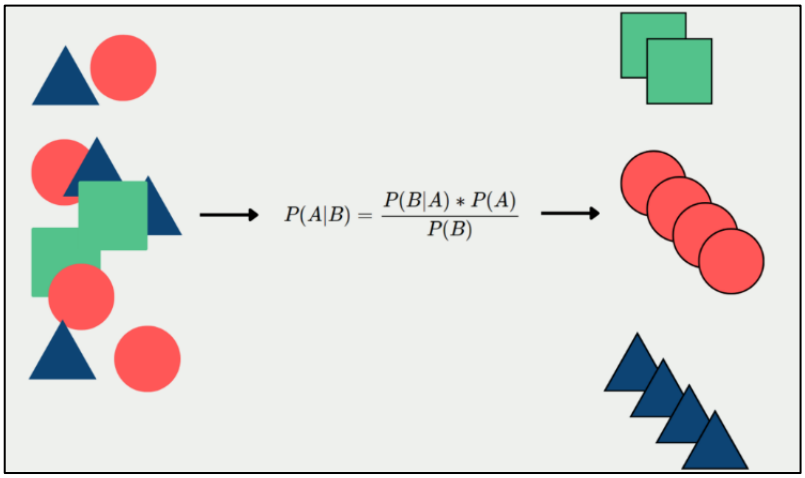
Gambar 2.1 Tipe *Machine Learning*

Machine learning memiliki banyak sekali tipe seperti yang ditunjukkan pada Gambar 2.1 (Thakkar, 2024), yang dikelompokkan berdasarkan tujuan yang ingin dicapai. Jenis-jenis yang biasa dipakai yakni sebagai berikut (Oladipupo, 2010):

1. *Supervised learning* (pembelajaran terawasi), algoritma belajar dari data yang sudah memiliki label (input-output). Misalkan pada proses klasifikasi, algoritma mempelajari pola dari contoh data untuk bisa menentukan kelas dari data baru.
2. *Unsupervised learning* (pembelajaran tanpa pengawasan), algoritma belajar dari data yang tidak memiliki label, hanya berupa kumpulan input, dan mencoba menemukan pola atau pengelompokan di dalamnya.
3. *Semi-supervised learning* (pembelajaran semi-terawasi), gabungan dari data yang berlabel dan tidak berlabel untuk membantu algoritma menghasilkan model atau pengklasifikasi yang lebih baik.
4. *Reinforcement learning* (pembelajaran penguatan), algoritma belajar dengan cara mencoba berbagai aksi dalam suatu lingkungan. Setiap aksi mendapat umpan balik (reward atau hukuman) yang digunakan untuk memperbaiki strategi pengambilan keputusan.
5. *Transduction*, mirip dengan supervised learning akan tetapi tidak membangun fungsi umum secara eksplisit. Sebaliknya, algoritma langsung mencoba memprediksi output baru berdasarkan input-output yang sudah ada dan input baru.
6. *Learning to learn* (belajar untuk belajar), algoritma belajar bagaimana cara belajar yang lebih baik, yaitu dengan menyesuaikan strategi pembelajaran berdasarkan pengalaman sebelumnya.

2.2.3 Naive Bayes Classifiers

Naive Bayes adalah algoritma klasifikasi berbasis probabilitas yang dibangun berdasarkan Teorema Bayes. Meskipun dianggap sebagai metode yang sederhana, algoritma ini telah terbukti sangat efektif dalam berbagai kasus klasifikasi, termasuk ketika asumsi dasarnya yaitu independensi antar atribut tidak sepenuhnya terpenuhi. Berdasarkan penelitian oleh Domingos & Pazzani (1997), menunjukkan bahwa algoritma *Naive Bayes* tetap dapat menghasilkan kinerja yang optimal dalam mengurangi tingkat kesalahan klasifikasi (*zero-one loss*), bahkan pada dataset yang memiliki korelasi antar fitur. Hal ini disebabkan oleh sifat algoritma yang memiliki bias tinggi akan tetapi varians rendah, sehingga sangat cocok untuk digunakan pada dataset yang kecil atau yang memiliki banyak fitur. Informasi terkait cara sederhana klasifikasi menggunakan *naive bayes* dilihat pada Gambar 2.2 (Samaraweera, 2024).



Gambar 2.2 *Naive Bayes Classifiers*

Secara matematis, prediksi kelas C untuk sebuah data $x = (x1,x2,\ldots,xn)$ dilakukan dengan cara memilih kelas yang memaksimalkan probabilitas posterior, yang dihitung menggunakan rumus Teorema Bayes sebagai berikut:

$$P(C|x) = \frac{P(x|C).P(C)}{P(x)}$$

Keterangan rumus:

P(C|x) : Probabilitas kelas C berdasarkan data x (probabilitas posterior)

P(C) : Probabilitas awal (prior) untuk kelas C

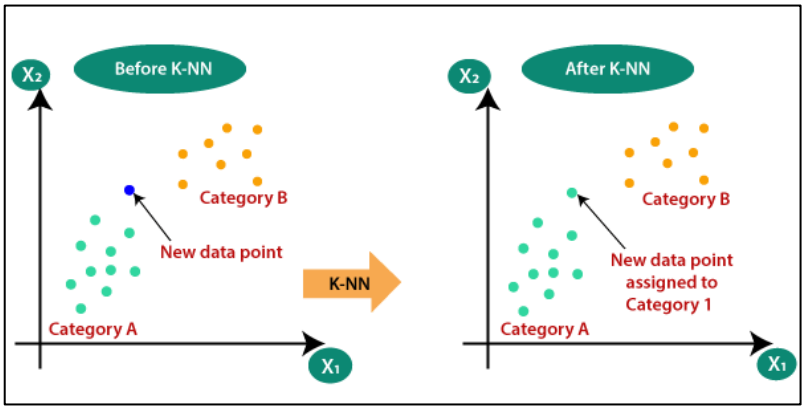
P(x|C) : Probabilitas data x berdasarkan kelas C

P(x) : Probabilitas dari data secara keseluruhan

Asumsi "*naive*" pada algoritma ini adalah bahwa setiap fitur dianggap bersifat independen secara kondisional terhadap kelas. Meski asumsi ini jarang terpenuhi dalam praktik, Domingos & Pazzani menunjukkan bahwa *Naive Bayes* tetap dapat berkinerja baik karena kesalahan dalam estimasi probabilitas tidak selalu menyebabkan kesalahan klasifikasi. Dengan demikian, dapat disimpulkan bahwa *Naive Bayes* tidak hanya mudah untuk diterapkan dan cepat, tetapi juga tahan terhadap pelanggaran asumsi independensi yang menjadikannya pilihan yang tepat untuk banyak kasus klasifikasi, termasuk dalam bidang kesehatan seperti prediksi obesitas.

2.2.4 K-Nearest Neighbor (KNN)

Algoritma *K-Nearest Neighbor* (KNN) merupakan salah satu metode klasifikasi pada *machine learning* yang bekerja berdasarkan prinsip pembelajaran terawasi (*supervised learning*), di mana kelas dari sebuah objek baru ditentukan dengan mengukur jarak kemiripan terhadap sekumpulan data latih yang sudah ada (Fasnuari, Yuana, & Chulkamdi, 2022). Konsep K-NN pertama kali diperkenalkan oleh Fix dan Hodges (1951) dan kemudian diformalkan oleh Cover dan Hart (1967).



Gambar 2.3 *K-Nearest Neighbor*

Metode ini mengklasifikasikan sebuah sampel uji berdasarkan mayoritas label dari *k* sampel pelatihan terdekat berdasarkan jarak geometrik yang kemudian hasil keputusan akhirnya dieksekusi melalui mekanisme penentuan kelas mayoritas atau *majority voting* yang dapat dilihat di Gambar 2.4 (Bzubeda, 2024). Umumnya, jarak *Euclidean* digunakan untuk mengukur kedekatan antar data. Jika x_i dan x_l merupakan dua titik dalam ruang berdimensi p, maka jarak *Euclidean* dapat didefenisikan sebagai berikut :

$$d_i = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Keterangan rumus :

d : Jarak Euclidean

n : Dimensi data

i : Variabel data

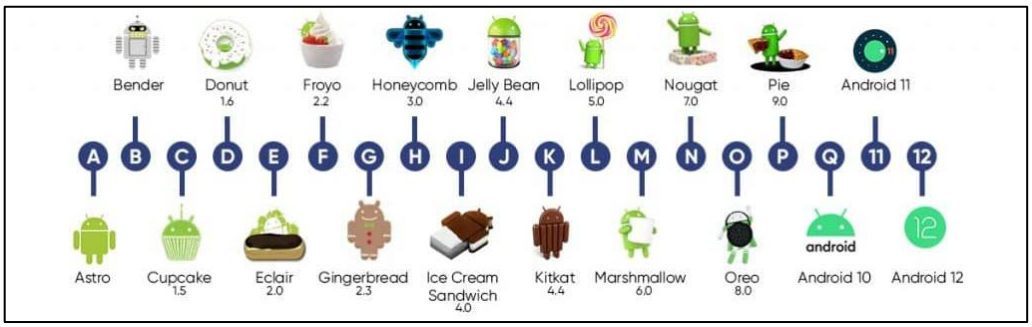
x : Data training

y : Data testing

Secara konseptual, algoritma ini tidak membangun sebuah model prediktif secara eksplisit selama fase pelatihan, melainkan menyimpan keseluruhan data historis untuk melakukan kalkulasi pencocokan hanya pada saat terdapat instans data pengujian baru; sebuah pendekatan yang secara teoritis dikenal sebagai *lazy learning* atau pembelajaran berbasis instans (Hendriyanto & Sari, 2022).

2.2.5 Android

Android merupakan sistem operasi mobile yang populer digunakan. *Android* berfokus dengan pendekatan pengembangan aplikasi. Dalam bahasa Inggris, "*Android*" diartikan sebagai robot yang menyerupai manusia. Sebagai sistem operasi, *android* memiliki fungsi sebagai penghubung antara perangkat keras dengan pengguna (*User*). *Android* adalah sistem operasi berbasis *Linux* yang dirancang untuk perangkat bergerak layar sentuh (Firly dkk., 2021).



Gambar 2.4 *Android Versions*

Perkembangan *android* dapat dilihat pada Gambar 2.4 (Shresth, 2021) yang dikembangkan oleh sebuah startup bernama *Android, Inc*.. Pada tahun 2005, Google membeli *android* beserta tim pengembangnya sebagai bagian dari strategi masuk ke pasar ponsel. Google ingin *android* bersifat terbuka dan gratis, sehingga sebagian besar kodenya dirilis dengan lisensi *open-source Apache License*. Artinya,

siapa saja bisa menggunakan *android* dengan mengunduh kode sumbernya (Lee, 2011). Hingga saat ini, *android* sudah memiliki banyak versi semenjak perilisan pertamanya yang dapat dilihat di Gambar 2.4. Berikut penjelasan singkat dari setiap versi android tersebut:

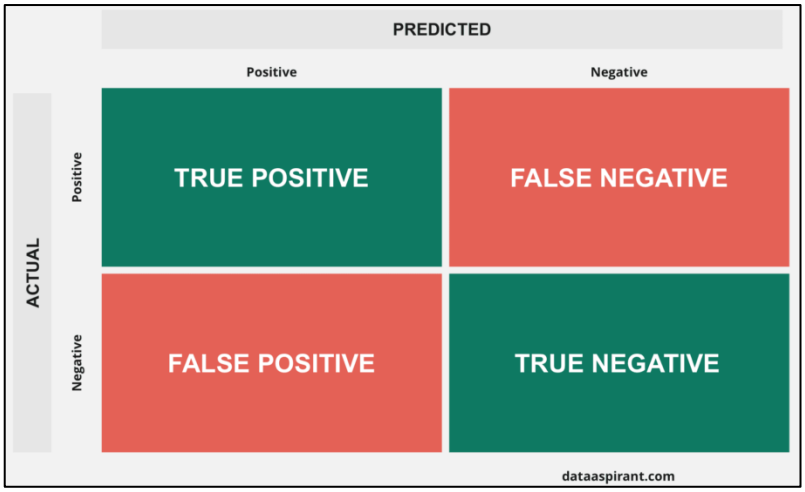
1. Android 1.0 (Alpha), Rilis 2008 dengan fitur dasar Gmail, YouTube, browser, WiFi, Bluetooth, dan Android Market.
2. Android 1.1 (Beta), Update bug (2009) dengan fitur detail lokasi di Maps, simpan lampiran email, dan dukungan teks berjalan (marquee).
3. Android 1.5 (Cupcake), Rilis 2009 dengan keyboard virtual, widget, rekam video, upload YouTube, dan pairing Bluetooth.
4. Android 1.6 (Donut), Rilis 2009 dengan pencarian suara, integrasi kamera-galeri, dukungan multi bahasa, dan resolusi WVGA.
5. Android 2.0–2.1 (Eclair), Rilis 2009 dengan multi-touch, sinkronisasi akun, Bluetooth 2.1, dan fitur kamera (flash, zoom, makro).
6. Android 2.2 (Froyo), Rilis 2010 dengan USB tethering, WiFi hotspot, update aplikasi otomatis, GIF di browser, dan Adobe Flash.
7. Android 2.3 (Gingerbread), Rilis 2010 dengan desain UI lebih simpel, dukungan NFC, VoIP, multikamera, giroskop, dan manajemen daya.
8. Android 3.0–3.2 (Honeycomb), Rilis 2011 khusus tablet dengan UI virtual, System Bar, Recent Apps, dukungan multi-core, dan enkripsi data.
9. Android 4.0 (Ice Cream Sandwich), Rilis 2011 dengan tombol navigasi virtual, screenshot, pengenal wajah, dan kontrol penggunaan data.
10. Android 4.1–4.3 (Jelly Bean), Rilis 2012 dengan UI lebih halus (60 fps), notifikasi lebih canggih, pencarian suara, dan kamera lebih baik.
11. Android 4.4 (KitKat), Rilis 2013 dengan tampilan lebih ringan, respon layar cepat, mode fullscreen baca, dan telepon pintar dengan kontak prioritas.
12. Android 5.0 (Lollipop), Rilis 2014 dengan desain Material Design, Project Volta hemat baterai, dan Factory Reset Protection.
13. Android 6.0 (Marshmallow), Rilis 2015 dengan izin aplikasi terkontrol, mode doze hemat daya, dan dukungan sidik jari.
14. Android 7.0–7.1 (Nougat), Rilis 2016 dengan split-screen, kalibrasi warna, layar zoom, dan penghapusan Recent Apps.

15. Android 8.0–8.1 (Oreo), Rilis 2017 dengan Project Treble, performa 2x lebih cepat, Play Protect, dan instalasi aplikasi dari sumber tidak dikenal.
16. Android 9 (Pie), Rilis 2018 dengan Adaptive Battery, Adaptive Brightness, navigasi gesture, dashboard penggunaan aplikasi, dan pembatasan waktu.
17. Android 10, Rilis 2019 tanpa nama makanan, dengan Live Caption, Smart Reply di notifikasi, dan mode gelap sistem.
18. Android 11, Rilis 2020 dengan balon chat, screen recorder bawaan, izin satu kali aplikasi, dan notifikasi percakapan terpisah.
19. Android 12 (Snow Cone), Rilis 2021 dengan desain "Material You," tema warna otomatis dari wallpaper, screenshot bergulir, dan kontrol privasi kamera/mikrofon.

Kelebihan utama dari *android* adalah memberikan pendekatan terpadu dalam pengembangan aplikasi. Pengembang hanya perlu membuat aplikasi untuk *android*, dan aplikasi itu bisa dijalankan di berbagai perangkat yang menggunakan *android*.

2.2.6 Confusion Matrix

Confusion Matrix adalah tabel N x N, dimana N merupakan jumlah kelas atau label yang berisi jumlah perkiraan yang benar dan salah dari model klasifikasi. Tujuan dari metode *confusion matrix* ini yaitu untuk membandingkan nilai aktual dengan nilai prediksi. Baris matriks mewakili kelas yang sebenarnya, sedangkan kolom mewakili kelas yang diprediksi (Syahputra & Wibowo, 2020).



Gambar 2.5 *Confusion Matrix*

Keterangan:

1. *True Positive* (TP): Jumlah record positif yang diklasifikasikan sebagai positif.

2. *True Negative* (TN): Jumlah record positif yang diklasifikasikan sebagai negatif.
3. *False Positive* (FP) : Jumlah record negatif yang diklasifikasikan sebagai positif.
4. *False Negative*(FN) : Jumlah record negatif yang diklasifikasikan sebagai negatif.

Hasil dari *confusion matrix* akan dilakukan evaluasi untuk melihat kinerja dari model dalam melakukan proses klasifikasi dengan cara menghitung *metrics* yang dapat mengukur kinerja model yang disebut dengan *classification report*. Berikut penjelasan dari *metrics* tersebut:

1) *Accuracy* : Menunjukkan sejauh mana model mampu mengklasifikasikan data dengan benar. Nilai *accuracy* dapat dihitung dengan menggunakan rumus:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

2) *Precision* : Menunjukkan tingkat ketepatan antara data yang relevan dengan hasil prediksi model. Nilai *precision* dapat dihitung dengan menggunakan rumus:

$$Precision = \frac{TP}{TP + FP}$$

3) *Recall* : Menunjukkan kemampuan model untuk mendeteksi atau menemukan kembali informasi yang relevan. Nilai *recall* dapat dihitung dengan rumus:

$$Recall = \frac{TP}{TP + FN}$$

4) *F1-Score* : Menggambarkan perbandingan rata-rata antara *Precision* dan *Recall*. Nilai *F1-Score* dapat dihitung dengan menggunakan rumus:

$$F1 - Score = \frac{recall \times precision}{recall + precision}$$