BAB III
METODE PENELITIAN

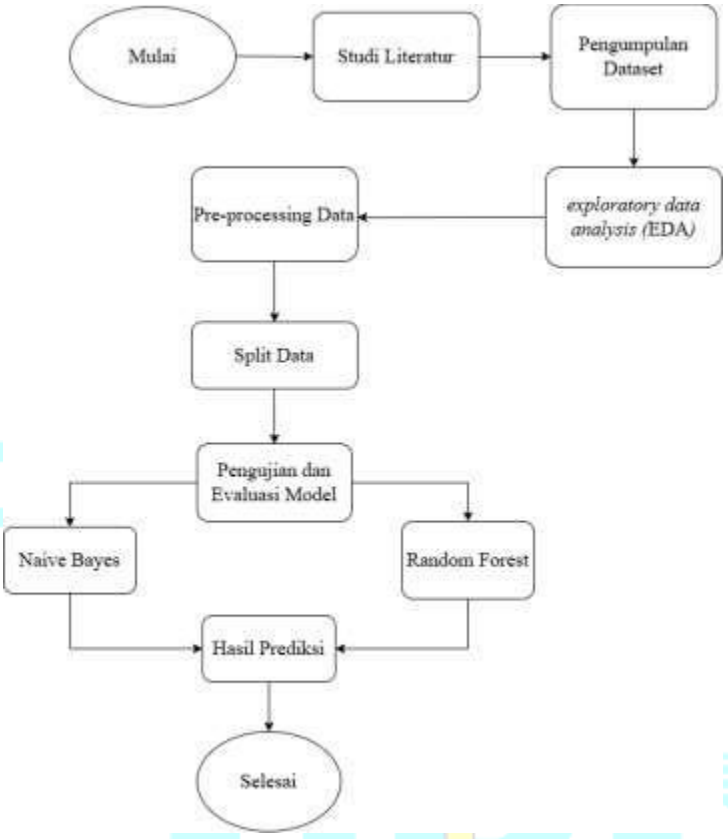
3.1. Objek Penelitian

Objek dalam penelitian ini yaitu data pertandingan dan performa tim dalam *English Premier League* (EPL) selama satu musim. Data yang digunakan mencakup berbagai atribut yang relevan, seperti jumlah gol yang dicetak, jumlah gol yang kebobolan, jumlah tembakan tepat sasaran, serta statistik lainnya yang dapat mempengaruhi hasil pertandingan hingga akhir musim.

Lokasi penelitian dilakukan di laboratorium Riset Universitas Buana Perjuangan Karawang. Waktu penelitian dimulai bulan September sampai dengan selesai.

3.2. Prosedur Penelitian

Penelitian ini mencakup delapan langkah prosedural yang akan dijalankan untuk melakukan prediksi menggunakan *machine learning*, dengan fokus pada perbandingan antara algoritma *Naïve Bayes* dan *Random Forest* dalam meramalkan tim juara *English Premier League* (EPL). Tahap pertama adalah melakukan studi literatur dengan cara membaca dan memahami konsep yang akan dilakukan untuk penelitian. Tahap selanjutnya melakukan pengumpulan data dari tahun 2022-2023 dan di sumber data yang terpercaya. Setelah itu dilakukan analisis data awal (*exploratory data analysis*/EDA). Tahap selanjutnya melakukan *pre-processing* data dilakukan, termasuk normalisasi atau standarisasi fitur, *encoding* variabel kategorikal, dan pembagian data menjadi data *testing* dan data *training*. Tahapan selanjutnya meliputi ekstraksi fitur, yakni pemilihan fitur kunci yang digunakan pembuatan model prediksi. Setelah proses ini selesai, model diuji dengan mengaplikasikan algoritma *Naïve Bayes* dan *Random Forest*.



Gambar 3. 1 *Flowchart Alur Penelitian*

3.2.1. Studi Literatur

Tahap awal yang penting dalam proses penelitian guna memahami landasan teoritis yaitu studi literatur. Tahapan ini dilakukan dengan mendalami berbagai referensi jurnal dan artikel ilmiah relevan dengan topik yang diangkat. Proses studi literatur dalam penelitian ini dilakukan melalui tiga aspek utama, yaitu mempelajari algoritma yang digunakan, meneliti studi-studi sebelumnya, dan mengidentifikasi parameter evaluasi.

Hasil dari studi literatur tidak hanya memberikan landasan teoritis, tetapi juga membantu dalam merancang metodologi penelitian yang tepat. Dengan memahami karakteristik algoritma, wawasan dari penelitian sebelumnya, dan metrik evaluasi, peneliti dapat membangun kerangka penelitian yang lebih solid, sistematis, dan relevan dengan tujuan penelitian.

3.2.2. Pengumpulan Data

Untuk mendapatkan dataset berkualitas, komprehensif, dan relevan guna mendukung proses model, perlu tahap pengumpulan data. Dalam studi ini, data dikumpulkan melalui *web scraping* dari situs *FBref.com*, yang dikenal sebagai sumber terpercaya dalam menyediakan statistik lengkap pertandingan sepak bola, termasuk kompetisi *English Premier League* (EPL). Proses pengumpulan data dilakukan dengan menggunakan pustaka *Python BeautifulSoup* untuk mengekstrak data dari halaman *web* yang relevan dan disimpan menjadi file *CSV*. Data yang diambil mencakup statistik dari 3 (tiga) musim yaitu musim 2021-2022, 2022-2023, 2023-2024.

A. Proses Pengumpulan Data

1. Mengidentifikasi URL Sumber:

   Peneliti menentukan halaman web spesifik yang mengandung data yang diperlukan, seperti halaman yang mencakup statistik pertandingan, skor, dan performa tim di setiap musim EPL.

Gambar 3. 2 Website *Fbref.com*

2. Menggunakan *BeautifulSoup* untuk Scraping:
   a. Kode *Python* menggunakan pustaka requests untuk mengambil konten HTML dari halaman web.
   b. *BeautifulSoup* digunakan untuk mem-parse HTML dan mengekstrak tabel data, nama tim, skor pertandingan, statistik pemain, dan informasi lainnya.
3. Struktur Data yang Dikumpulkan:

   Data yang diekstrak mencakup berbagai informasi historis dari

pertandingan sepak bola, yang meliputi skor akhir, jumlah gol yang tercetak, serta berbagai statistik permainan seperti penguasaan bola, jumlah tembakan, jumlah pelanggaran, jumlah offside, dan statistik lainnya yang relevan. Berikut data yang diambil dan diekstrak :

Tabel 3 1 Kolom Data

Data	Deskripsi
Date	Tanggal Pertandingan
Time	Jam Pertandingan
Comp	Nama Kompetisi
Round	Pekan Pertandingan
Day	Hari Pertandingan
Venue	Kandang/Tandang
Result	Hasil (Menang, Seri, Kalah).
GF (Goal For)	Jumlah goal yang dicetak tim
GA (Goal Against)	Jumlah goal yang di terima tim
Opponent	Nama tim lawan
xG	Expected goals untuk tim.
xGA	Expected goals melawan tim.
Poss	Persentase penguasaan bola.
Attendance	Jumlah Penonton yang hadir.
Captain	Nama kapten tim
Formation	Formasi tim saat pertandingan
Referee	Nama wasit memimpin pertandingan
Match Report	laporan pertandingan.
Notes	Catatan tambahan tentang pertandingan.
Sh (Shoots)	Total jumlah tembakan.
SoT (Shoots On Target)	Jumlah tembakan tepat sasaran.
Dist	Rata-rata jarak tembakan (dalam meter).
FK	Jumlah tendangan bebas.

PK (Penalty Kick)	Jumlah penalti yang dicetak.
PKatt (Penalty Kicks Attempted)	Jumlah penalti yang dieksekusi.
Season	Musim kompetisi.
Team	Nama tim.

4. Pengelolaan Data

Data hasil scraping disimpan dalam format terstruktur, seperti *CSV* atau *DataFrame* menggunakan pustaka *pandas* pada *Python*. Format ini memberikan kemudahan dalam mengelola dan menyimpan data dengan rapi, sehingga memudahkan proses analisis dan *preprocessing* di tahap selanjutnya. Dengan menggunakan *DataFrame*, data dapat diorganisir secara kolom-kolom, membuatnya lebih mudah untuk melakukan manipulasi, *filtering*, serta *transformasi* sesuai kebutuhan penelitian.

B. Proses Validasi Data

Untuk memastikan kualitas dan keakuratan data yang diperoleh dari scraping, langkah-langkah validasi data dilakukan dengan teliti, khususnya dalam memastikan bahwa data yang diambil sesuai dengan standar yang diinginkan. Berikut adalah penjelasan lebih detail mengenai proses validasi data:

a. Pengecekan Kelengkapan Data

Dalam *scraping data* dari website seperti *FBref.com*, data yang diambil harus melewati tahap pengecekan untuk memastikan bahwa setiap atribut memiliki data yang sesuai. Kolom seperti skor akhir, jumlah gol, statistik permainan, dan informasi tim harus dilengkapi dengan baik agar tidak ada data yang terlewatkan.

b. *Cross-Check* dengan Sumber Lain

Data yang telah diekstrak melalui scraping dari *FBref.com* perlu dibandingkan dengan sumber lain yang lebih terpercaya,

seperti laporan resmi dari pertandingan atau data dari platform lain yang sejenis. Jika terdapat ketidaksesuaian, maka perlu dilakukan pemeriksaan ulang dan penyesuaian pada data hasil scraping agar sesuai dengan fakta yang benar.

3.2.3. *Exploratory Data Analysis (*EDA)

Tahap *Exploratory Data Analysis (EDA)* adalah langkah awal yang esensial dalam proses analisis data. EDA bertujuan untuk memahami karakteristik dataset secara mendalam, mengidentifikasi pola dan anomali, serta menemukan hubungan antar fitur. Tahapan ini sangat penting sebelum melanjutkan ke proses analisis atau pemodelan machine learning, karena kualitas dan pemahaman data memiliki dampak langsung pada hasil akhir penelitian. Langkah-langkah EDA yang dilakukan:

a. Menggunakan fungsi seperti *describe()* dalam Python untuk memberikan gambaran umum tentang dataset.
b. Visualisasi Data

   Visualisasi digunakan untuk memahami distribusi dan pola dalam data secara intuitif:

   a. *Line Plot*
   b. Diagram Batang
   c. *Diagram Pareto*

   Tahap *EDA* memberikan wawasan mendalam tentang karakteristik dataset, membantu dalam penyesuaian dan pembersihan data, serta memberikan landasan yang kuat untuk proses pemodelan machine learning.

3.2.4. Pre-Processing

Preprocessing data merupakan tahap krusial dalam siklus penelitian, agar data siap dipakai dalam analisis serta pemodelan *machine learning*. Proses ini bertujuan untuk memastikan data berkualitas baik, bebas dari kesalahan, dan diformat sesuai dengan kebutuhan algoritma yang akan diterapkan. Selain itu, *preprocessing* juga berfungsi untuk mengenali potensi permasalahan pada data yang dapat mempengaruhi hasil analisis, seperti data

yang hilang, nilai yang tidak masuk akal, atau variabel yang kurang relevan. Dalam penelitian ini, tahap preprocessing mencakup beberapa tahap berikut:

1. Memuat Data

   Langkah pertama adalah memuat dataset yang berisi data pertandingan, performa tim, dan hasil pertandingan. Dataset ini memiliki berbagai fitur statistik pertandingan guna melakukan prediksi.

2. Identifikasi Missing Values

   Pada tahap ini, data hasil scraping diperiksa secara menyeluruh untuk memastikan bahwa data tersebut bebas dari nilai yang hilang (*missing values*) atau nilai yang tidak valid, yang dapat memengaruhi hasil analisis dan pemodelan. Pengecekan dilakukan pada setiap kolom data untuk menemukan nilai kosong (*null* atau *NaN*) yang mungkin ada akibat ketidaksempurnaan dalam proses scraping. Nilai kosong ini perlu diatasi dengan pendekatan yang sesuai agar tidak memengaruhi hasil analisis. Berikut tahap dalam proses pembersihan data (*data cleaning*):

   1. Identifikasi *Missing Values*

      Langkah pertama adalah melakukan pengecekan secara menyeluruh untuk menemukan nilai kosong yang ada di dalam dataset. Pengecekan dilakukan pada setiap kolom dan baris untuk mengidentifikasi seberapa banyak data yang hilang. Proses ini biasanya menggunakan fungsi-fungsi dari pustaka seperti *pandas*, misalnya *isnull()* atau *info()*, yang memberikan gambaran lengkap tentang jumlah nilai kosong pada setiap kolom. Tujuan dari langkah ini adalah memahami sejauh mana *missing values* tersebar dalam dataset sehingga dapat ditentukan strategi penanganannya.

   2. Penanganan *Missing Values*

      Setelah nilai kosong diidentifikasi, langkah berikutnya adalah mengatasi nilai kosong tersebut dengan strategi yang sesuai berdasarkan jenis data dan tingkat keberadaannya. Berikut adalah pendekatan yang dilakukan:

      a. Menghapus Kolom dengan *Missing Values* yang Signifikan: Kolom memiliki nilai kosong lebih dari 50% dianggap tidak

memberikan informasi yang cukup signifikan untuk analisis. Oleh karena itu, kolom tersebut dihapus dari *dataset*

b. Mengisi Missing Values pada Kolom Numerik: Kolom yang berisi data numerik, seperti jumlah gol, persentase penguasaan bola, atau jumlah tembakan, nilai kosong diisi menggunakan rata-rata (*mean*) dari kolom tersebut. Metode ini dipilih karena rata-rata memberikan nilai yang representatif tanpa mengubah distribusi data secara signifikan. Misalnya, jika kolom "Jumlah Gol" memiliki nilai kosong, nilai kosong tersebut akan diisi dengan rata-rata gol dari seluruh data.

c. Mengisi Missing Values pada Kolom Kategorikal: Untuk kolom yang berisi data kategorikal, seperti nama tim, hasil pertandingan (*Result*), atau formasi, nilai kosong diisi menggunakan nilai modus.

d. Validasi Setelah Penanganan *Missing Values*: Setelah proses pengisian atau penghapusan nilai kosong selesai, dataset diperiksa ulang untuk memastikan bahwa semua nilai kosong telah berhasil diatasi. Validasi ini dilakukan dengan cara menjalankan kembali fungsi pengecekan nilai kosong untuk memastikan dataset telah bersih dari *missing values*.

3. Seleksi Fitur

Seleksi fitur merupakan proses pemilihan atribut atau kolom dalam dataset yang memiliki tingkat relevansi tinggi terhadap variabel target yang akan diprediksi. Tujuan dari langkah ini adalah untuk menyederhanakan model, mengurangi beban komputasi, serta meningkatkan akurasi prediksi dengan hanya menggunakan fitur-fitur yang memang diperlukan. Pada penelitian ini, pemilihan fitur didasarkan pada kaitannya dengan hasil pertandingan (*Result*) sebagai variabel target prediksi.

4. *Encoding* Variabel Kategorikal

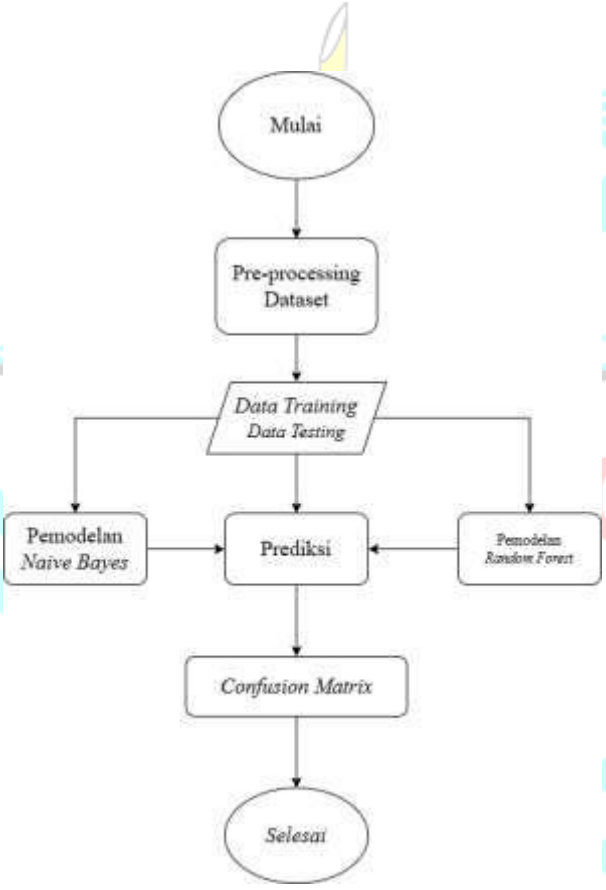
Pada penelitian ini, variabel target *result*, yang berisi kategori hasil pertandingan, dikonversi menjadi nilai numerik menggunakan metode *Label Encoding*.

Tabel 3 2 Label Numerik

Atribut	Keterangan	Label
Lose	Kalah	0
Draw	Seri	1
Win	Menang	2

3.2.5. Pengujian dan Evaluasi Model Algoritma

Pengujian model algoritma dalam penelitian ini dilakukan untuk mengevaluasi performa dari algoritma *machine learning*, yaitu *Naïve Bayes* dan *Random Forest*. Tujuan dari pengujian ini adalah untuk mengukur sejauh mana kedua algoritma dapat memprediksi hasil pertandingan *English Premier League* (EPL).



Gambar 3. 3 *Flowchart* Pengujian dan Evaluasi

1. Pembagian Data

   Sebelum tahap pengujian, data yang telah diproses akan dibagikan menjadi dua bahan data utama:

   a. *Data Training*: Digunakan untuk melatih model. Data training biasanya terdiri dari sekitar 70-80% dari keseluruhan dataset.
   b. *Data Testing*: Untuk menguji performa model. Data testing terdiri dari 20-30% dari keseluruhan dataset.

   Proses split data:

   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

2. *Naïve Bayes*

Model *Naïve Bayes* digunakan untuk melakukan klasifikasi dengan asumsi bahwa setiap fitur bersifat independen terhadap kelasnya. Model ini biasanya efektif untuk mengolah data numerik maupun kategori, seperti hasil pertandingan.

$$P(A|B) \frac{P(B|A) \; x \; P(A)}{P(B)} \quad (3.1)$$

Keterangan :

P(A|B) = Kemungkinan A terjadi dengan syarat B sudah terjadi.

P(B|A) = Kemungkinan B terjadi dengan syarat A sudah terjadi.

P(A) = Peluang terjadinya peristiwa A

P(B) = Peluang terjadinya peristiwa B

Proses kerja metode *Naïve Bayes* dalam penelitian dijelaskan sebagai berikut:

Langkah 1:

Baca dataset hasil praproses yang berisi fitur dan label (target)

Langkah 2:

Pilih kolom-kolom yang akan digunakan sebagai fitur:

['gf', 'ga', 'xg', 'xga', 'poss', 'sh', 'sot', 'dist', 'fk', 'pk']
Pilih kolom target: result

Langkah 3:
Gunakan StandardScaler untuk menstandarkan fitur agar memiliki distribusi normal (mean = 0, std = 1).
Fit dan transformasi pada data latih dan data uji

Langkah 4:
Pisahkan dataset menjadi data latih dan data uji.
80% data untuk pelatihan
20% data untuk pengujian
Gunakan train_test_split dengan random_state=42 untuk hasil yang konsisten.

Langkah 5:
Inisialisasi model Naive Bayes menggunakan *GaussianNB.*
Latih model menggunakan data latih (X_train, y_train).

nb_model = GaussianNB()
nb_model.fit(X_train, y_train)

Langkah 6:
Lakukan prediksi pada data uji (X_test) menggunakan model yang telah dilatih.
y_pred = nb_model.predict(X_test)

Langkah 7:
Evaluasi performa model:
Hitung akurasi
target_names = list(map(str, label_encoder.classes_))
Tampilkan classification report (precision, recall, f1-score)

Tampilkan Confusion Matrix untuk melihat distribusi prediksi terhadap label sebenarnya

cm = confusion_matrix(y_test, y_pred)

3. *Random Forest*

Model ini sangat ampuh dalam mencegah *overfitting* karena prediksi akhir diperoleh dari gabungan banyak pohon keputusan. *Random Forest* mampu mengolah data numerik maupun kategorikal, serta dapat menilai seberapa penting setiap fitur dalam memprediksi target. Metode ini terdiri dari sejumlah pohon keputusan yang digunakan bersama-sama untuk mengklasifikasikan data ke dalam kelas tertentu.

Rumus:

$$\boldsymbol{l(y) = argmac_c(\sum_{n=1}^{N} I\, n_n(y) = c)} \quad (3.2)$$

Keterangan :

a. Pohon pada *Random Forest* memberikan prediksi untuk input *y*.
b. Setiap prediksi pohon tersebut dihitung jumlahnya berdasarkan kelas *c*.
c. Kelas dengan suara terbanyak (argmax) dipilih sebagai prediksi akhir *l(y)*.

Alur kerja metode *Random Forest* dalam penelitian ini adalah sebagai berikut:

Langkah 1:

Baca dataset hasil praproses yang berisi fitur dan label (target)

Langkah 2:

Pilih kolom-kolom yang akan digunakan sebagai fitur:

['gf', 'ga', 'xg', 'xga', 'poss', 'sh', 'sot', 'dist', 'fk', 'pk']

Pilih kolom target: result

Langkah 3:

Gunakan StandardScaler untuk menstandarkan fitur agar memiliki distribusi normal (mean = 0, std = 1).

Fit dan transformasi pada data latih dan data uji

Langkah 4:

Pisahkan dataset menjadi data latih dan data uji.

80% data untuk pelatihan

20% data untuk pengujian

Gunakan train_test_split dengan random_state=42 untuk hasil yang konsisten.

Langkah 5:

Inisialisasi model Naive Bayes menggunakan *RandoForestClassifier*

Latih model menggunakan data latih (X_train, y_train).

```
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

Langkah 6:

Lakukan prediksi pada data uji (X_test) menggunakan model yang telah dilatih.

```
y_pred = nb_model.predict(X_test)
```

Langkah 7:

Evaluasi performa model:

Hitung akurasi

```
target_names = list(map(str, label_encoder.classes_))
```

Tampilkan classification report (precision, recall, f1-score)

Tampilkan Confusion Matrix untuk melihat distribusi prediksi terhadap label sebenarnya

cm = confusion_matrix(y_test, y_pred)

3.2.6. Confusion Matrix

Confusion matrix akan digunakan untuk evaluasi hasil prediksi klasemen *English Premier League* yang dibuat oleh algoritma *Naïve Bayes* dan *Random Forest*. Data dalam confusion matrix terdiri dari hasil prediksi model yang dihasilkan oleh kedua algoritma *Machine Learning* tersebut serta data real yang merupakan hasil klasemen aktual dalam musim *English Premier League* yang dianalisis. Dengan menggunakan confusion matrix, penelitian ini dapat mengukur seberapa baik kedua algoritma dalam memprediksi hasil klasemen, mengidentifikasi kesalahan prediksi seperti *false positive* (FP) dan *false negative* (FN), serta menentukan manakah algoritma dengan performa lebih baik.

Berikut ini adalah rumus untuk masing-masing metrik tersebut:

a. *Accuracy*

Menghitung rasio dari total prediksi yang benar terhadap semua prediksi.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+FP+FN+TN}} \quad (3.3)$$

b. *Precision*

Menghitung jumlah prediksi positif yang tepat dibandingkan dengan total keseluruhan prediksi positif yang dihasilkan.

$$\boldsymbol{Precision} = \frac{\boldsymbol{TP}}{\boldsymbol{TP=FP}} \quad (3.4)$$

c. *Recall*

Menghitung berapa banyak sampel positif yang dikenali oleh model dibandingkan dengan semua sampel positif.

$$\boldsymbol{Recall} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad (3.5)$$

d. *F1-Score*

Menggabungkan presisi dan recall ke dalam satu metrik.

$$\boldsymbol{F1} = \frac{\boldsymbol{2\ x\ Recall\ x\ Precision}}{\boldsymbol{Recall+Precision}} \quad (2.3)$$