BAB IV

HASIL DAN PEMBAHASAN

4.1 Persiapan

4.1.1 Studi Literatur

Ketika Peneliti membuat laporan penelitian ini, Peneliti harus mengikuti metode penelitian yang di buat sebelumnya. Langkah pertama adalah mencari daftar buku, jurnal, Esai, dan laporan penelitian sebelumnya untuk menjadi dasar penelitian ini. Bab II mencakup tinjauan literatur yang berfungsi sebagai landasan teori untuk penelitian ini. Tinjauan ini menjelaskan teori yang mendukung penelitian.

4.1.2 Pengumpulan Data dan Informasi

Pada tahap awal pengumpulan data dan informasi, kegiatan pengumpulan data dan informasi yang diperlukan untuk penelitian ini dilakukan dan beberapa langkah harus dilakukan untuk melakukan kegiatan tersebut.

a. Observasi

Setelah peneliti memutuskan alat yang dibutuhkan, mereka melakukan observasi untuk menemukan dataset *breast cancer* di kaagle.com.

b. Dokumentasi

Pada fase ini peneliti mendokumentasikan hasil dokumentasi yang telah dilakukan sebelumnya, dan hasil dokumentasi dari kegiatan tersebut digunakan sebagai urutan/urutan laporan dan penelitian yang dilakukan.

4.2 Pelaksanaan Eksperiment

4.2.1 Pemodelan Data

Pemodelan data dilakukan dengan melakukan langkah-langkah preprocessing atau pengolahan data. Artinya, beberapa proses melakukan langkah-langkah analisis data. Detail dari proses-proses tersebut adalah sebagai berikut:

a. Intalasi dan Pengimporan Pustaka

Untuk memulai analisis data dan membangun model machine learning di Google Colab, pertama-tama perlu menginstal beberapa pustaka penting. Perintah !pip install pandas scikit-learn tensorflow digunakan untuk menginstal tiga pustaka utama: pandas, scikit-learn, dan tensorflow. Pustaka pandas membantu dalam manipulasi dan analisis data, memudahkan pengolahan tabel data. Scikit-learn menyediakan berbagai algoritma dan alat untuk pembelajaran mesin, termasuk teknik untuk klasifikasi, regresi, dan seleksi fitur. Sementara itu, tensorflow digunakan untuk membangun dan melatih model deep learning, seperti neural networks. Setelah menginstal pustaka-pustaka ini, Anda dapat mengimpor mereka ke dalam lingkungan kerja dengan perintah import sklearn. Scikit-learn adalah salah satu alat utama dalam machine learning yang memungkinkan menerapkan berbagai algoritma dan teknik analisis data dengan efisien

b. Impor Library

```
# Import library
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.neural_network import MLPClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.metrics import (
    accuracy_score,
    precision_score,
    recall_score,
    confusion_matrix,
    classification_report,
    f1_score
)
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
```

Dalam kode ini, beberapa pustaka penting diimpor untuk mendukung analisis data dan pembangunan model machine learning. **pandas** digunakan untuk manipulasi data, memudahkan pengolahan dan analisis data dalam format tabel. Untuk visualisasi data, digunakan **matplotlib.pyplot** dan **seaborn**, yang memungkinkan pembuatan grafik dan plot untuk memahami pola dan hubungan dalam data. **numpy** menyediakan fungsi matematika dan operasi array yang efisien untuk pemrosesan data.

Dalam hal model machine learning, **MLPClassifier** dan **KNeighborsClassifier** dari pustaka **sklearn.neural_network** dan **sklearn.neighbors** masing-masing digunakan untuk membangun model neural network dan K-Nearest Neighbors. Untuk pemilihan parameter model dan membagi data menjadi set pelatihan dan pengujian, digunakan **GridSearchCV** dan **train_test_split** dari

sklearn.model_selection. **StandardScaler** membantu dalam normalisasi data, sedangkan **cross_val_score** dan **StratifiedKFold** digunakan untuk evaluasi model dan pembagian data secara stratifikasi.

Pustaka **sklearn.metrics** menyediakan berbagai metrik evaluasi seperti **accuracy_score**, **precision_score**, **recall_score**, **confusion_matrix**, **classification_report**, dan **f1_score** yang digunakan untuk mengukur kinerja model. Terakhir, **RFE** dari **sklearn.feature_selection** digunakan untuk seleksi fitur, sementara **LogisticRegression** dari **sklearn.linear_model** adalah model regresi logistik yang sering diterapkan dalam analisis klasifikasi.

c. Membaca dataset

Pada tahap pertama peneliti akan membaca dataset terlebih dahulu yang dapat dilihat dibawah ini.

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...

5 rows × 32 columns

Gambar 4. 1 Dataset Breast Cancer

Pada gambar 4.1 Dataset Breast Cancer yang tersedia di Kaggle berisi data tentang diagnosis kanker payudara dan terdiri dari 32 atribut yang menggambarkan berbagai karakteristik sel kanker. Atribut-atribut ini termasuk ukuran radius, tekstur, perimeter, area, dan berbagai ukuran lain dari sel-sel kanker. Contoh atribut yang terdapat dalam dataset adalah radius_mean, texture_mean, perimeter_mean,

area_mean, serta ukuran statistik lainnya seperti deviasi standar dan variansi dari fitur-fitur tersebut. Selain fitur-fitur ini, dataset juga mencakup kolom label target yang menunjukkan jenis diagnosis kanker, yaitu M (Malignant - ganas) atau B (Benign - jinak). Dengan total 569 sampel, dataset ini digunakan untuk keperluan klasifikasi dalam memprediksi jenis kanker payudara berdasarkan karakteristik sel yang terukur.

d. Pemrosesan Awal dan Evaluasi Korelasi Fitur pada Dataset Diagnosis Kanker Payudara

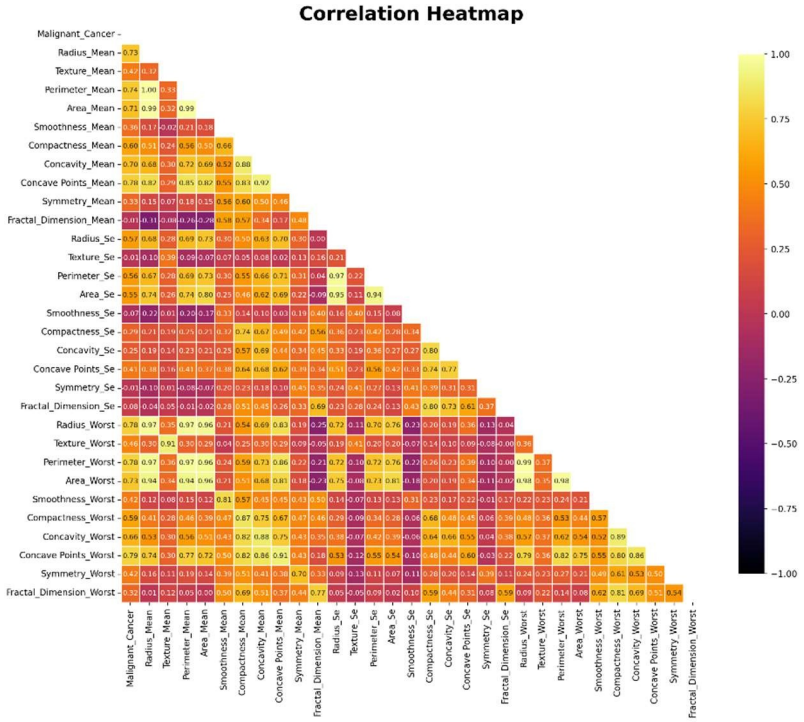
```
# Drop kolom 'id'
df = df.drop("id", axis="columns")

# Map kolom 'diagnosis' ke nilai biner
df["diagnosis"] = df["diagnosis"].map({"M": 1, "B": 0})
df = df.rename({"diagnosis": "malignant_cancer"}, axis="columns")

# Buat dataframe korelasi
data = df.corr()
data_cancer = pd.DataFrame(data["malignant_cancer"]).drop("malignant_cancer", axis="rows"
data_cancer = data_cancer.sort_values("malignant_cancer", ascending=False)
```

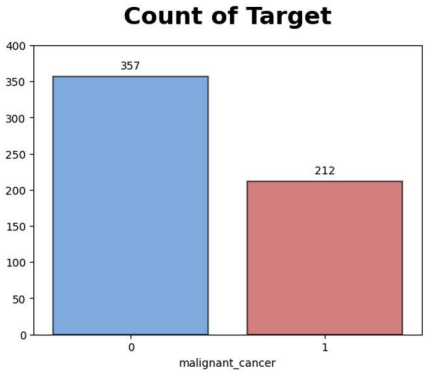
Pada langkah awal, kolom id dihapus dari dataframe karena hanya berfungsi sebagai identifikasi unik tiap sampel dan tidak memberikan kontribusi informasi yang relevan untuk analisis atau pemodelan. Selanjutnya, kolom diagnosis yang semula berisi nilai kategorikal M (Malignant - ganas) dan B (Benign - jinak) diubah menjadi nilai biner, dengan M diubah menjadi 1 dan B diubah menjadi 0. Transformasi ini bertujuan untuk mempermudah proses analisis dan pemodelan, terutama dalam konteks machine learning. Setelah itu, nama kolom diagnosis diubah menjadi malignant_cancer untuk memperjelas bahwa kolom tersebut mengindikasikan apakah kanker bersifat ganas (1) atau tidak (0).

Langkah berikutnya adalah menghitung matriks korelasi untuk semua kolom dalam dataframe, yang memberikan gambaran hubungan linier antara pasangan fitur dalam dataset. Dari matriks korelasi ini, dibuat DataFrame baru yang hanya berisi korelasi antara setiap fitur dengan kolom malignant_cancer, menghapus kolom malignant_cancer sendiri dari daftar ini untuk fokus pada hubungan antara fitur-fitur lainnya dengan kanker ganas. DataFrame ini kemudian diurutkan berdasarkan nilai korelasi dalam urutan menurun, untuk mengidentifikasi fitur-fitur yang memiliki korelasi tertinggi dengan kanker ganas. Analisis ini membantu dalam proses seleksi fitur, memungkinkan kita untuk memilih fitur-fitur yang paling relevan dan signifikan dalam pemodelan prediktif kanker payudara. Hasil gambar korelasi dapat dilihat pada gambar dibawah ini.



Gambar 4. 2 Korelasi Heatmap

e. Pemilihan jumlah data Target



Gambar 4. 3 Jumlah data Target

Pada Gambar 4.3 Jumlah data target malignan_cancer gambar diatas menunjukan jumlah sampel dengan kanker ganas (1) sebanyak 212 data dan tidak ganas (0) 357 data.

f. Standarisasi dan Seleksi Fitur

Dalam langkah ini, dilakukan standarisasi fitur dan seleksi fitur menggunakan Recursive Feature Elimination (RFE). Pertama, standarisasi fitur dilakukan dengan menggunakan StandardScaler dari Scikit-Learn. Standarisasi ini penting untuk memastikan semua fitur memiliki skala yang sama, mengurangi bias yang mungkin timbul karena perbedaan skala. StandardScaler menghitung rata-rata dan standar deviasi dari data pelatihan (X_train), lalu menggunakan nilai-nilai ini untuk menstandarisasi data pelatihan dan data uji (X_test).

Setelah standarisasi, seleksi fitur dilakukan menggunakan metode RFE. Logistic Regression dengan maksimal iterasi 5000 digunakan sebagai estimator dalam RFE. RFE bekerja dengan secara iteratif menghapus fitur paling tidak penting, berdasarkan koefisien model regresi logistik, hingga hanya tersisa sejumlah fitur

yang diinginkan, dalam hal ini 10 fitur. Fitur-fitur yang terpilih kemudian dicatat dan digunakan untuk membangun kembali set pelatihan dan uji yang hanya mencakup fitur-fitur terpilih tersebut.

Hasil akhir dari proses ini adalah data pelatihan dan data uji yang telah distandarisasi dan disederhanakan, hanya menyisakan 10 fitur yang paling relevan untuk prediksi kanker ganas. Langkah-langkah ini penting untuk meningkatkan performa model dan mengurangi kompleksitas, sehingga model dapat belajar dan membuat prediksi dengan lebih efisien. Hasil seleksi fitur dapat dilihat pada gambar 4.4 dibawah ini.

```
Fitur yang terpilih: Index(['concave points_mean', 'radius_se', 'perimeter_se', 'area_se',
       'compactness_se', 'radius_worst', 'texture_worst', 'perimeter_worst',
       'concavity_worst', 'concave points_worst'],
      dtype='object')
```

Gambar 4. 4 Hasil Seleksi Fitur

4.3 Hasil Eksperimen

4.3.1 Hasil Pengujian Performa Algoritma K-Nearest Neighbors

Tabel 4. 1 Pengujian Performa Algoritma K-Nearest Neighbors

Seleksi Fitur	Algoritma	Performa			
		Akurasi %	Presisi %	Recall %	F1 Score %
RFE	KNN	97.56	97.32	96.29	96.71
	KNN	89.03	88.09	89.21	90.00

Berdasarkan hasil evaluasi performa model KNN (K-Nearest Neighbors) dengan dan tanpa menggunakan seleksi fitur menggunakan metode RFE (Recursive Feature Elimination), berikut adalah narasi penjelasannya:

Performa KNN dengan Seleksi Fitur (RFE)

Menggunakan RFE sebagai metode seleksi fitur, model KNN mencapai hasil sebagai berikut:

- **Akurasi:** 97,56%
- **Presisi:** 97,32%
- **Recall:** 96,29%
- **F1 Score:** 96,71%

Ini menunjukkan bahwa model KNN dengan seleksi fitur RFE memberikan kinerja yang sangat baik. Akurasi yang tinggi (97,56%) menunjukkan bahwa model mampu memprediksi dengan benar sebagian besar sampel. Presisi yang tinggi (97,32%) mengindikasikan bahwa ketika model memprediksi positif, sebagian besar prediksi tersebut benar. Recall yang mencapai 96,29% menunjukkan bahwa

model berhasil mengidentifikasi sebagian besar sampel positif yang sebenarnya. F1 Score yang hampir mendekati nilai Presisi dan Recall (96,71%) menunjukkan keseimbangan yang baik antara keduanya.

Performa KNN tanpa Seleksi Fitur

Tanpa menggunakan seleksi fitur, performa model KNN menurun dengan hasil sebagai berikut:

- **Akurasi:** 89,03%
- **Presisi:** 88,09%
- **Recall:** 89,21%
- **F1 Score:** 90,00%

Meskipun performanya masih cukup baik, semua metrik menunjukkan penurunan yang signifikan dibandingkan dengan model yang menggunakan seleksi fitur. Akurasi menurun menjadi 89,03%, yang menunjukkan bahwa model ini membuat lebih banyak kesalahan dalam prediksi. Presisi turun menjadi 88,09%, yang berarti model lebih sering memberikan prediksi positif yang salah. Recall sedikit lebih baik pada 89,21%, tetapi tetap lebih rendah dibandingkan dengan model yang menggunakan RFE. F1 Score sebesar 90,00% mengindikasikan bahwa meskipun ada keseimbangan antara Presisi dan Recall, performa keseluruhan masih lebih rendah dibandingkan dengan model yang menggunakan seleksi fitur.

4.3.2 Hasil Pengujian Performa Algoritma Artificial Neural Network

Tabel 4. 2 Pengujian Performa Algoritma Artificial Neural Network

Seleksi Fitur	Algoritma	Performa			
		Akurasi %	Presisi %	Recall %	F1 Score %
RFE	ANN	98.64	99.31	97.03	98.13
	ANN	92.24	91.89	92.21	92.09

Berdasarkan hasil evaluasi performa model ANN (Artificial Neural Network) dengan dan tanpa menggunakan seleksi fitur menggunakan metode RFE (Recursive Feature Elimination), berikut adalah penjelasan narasinya:

Performa ANN dengan Seleksi Fitur (RFE)

Ketika RFE digunakan sebagai metode seleksi fitur, performa model ANN meningkat secara signifikan dengan hasil sebagai berikut:

- **Akurasi:** 98,64%
- **Presisi:** 99,31%
- **Recall:** 97,03%
- **F1 Score:** 98,13%

Model ANN dengan seleksi fitur menunjukkan kinerja yang sangat kuat. Akurasi sebesar 98,64% menandakan bahwa model mampu memprediksi hampir semua sampel dengan benar. Presisi yang sangat tinggi, yaitu 99,31%, mengindikasikan bahwa prediksi positif yang dibuat oleh model hampir semuanya benar, sehingga model sangat efektif dalam mengurangi kesalahan prediksi positif. Recall sebesar 97,03% menunjukkan bahwa model dapat mengenali sebagian besar sampel positif

yang ada. F1 Score yang mencapai 98,13% menandakan adanya keseimbangan yang sangat baik antara Presisi dan Recall, menunjukkan kinerja model yang sangat handal secara keseluruhan.

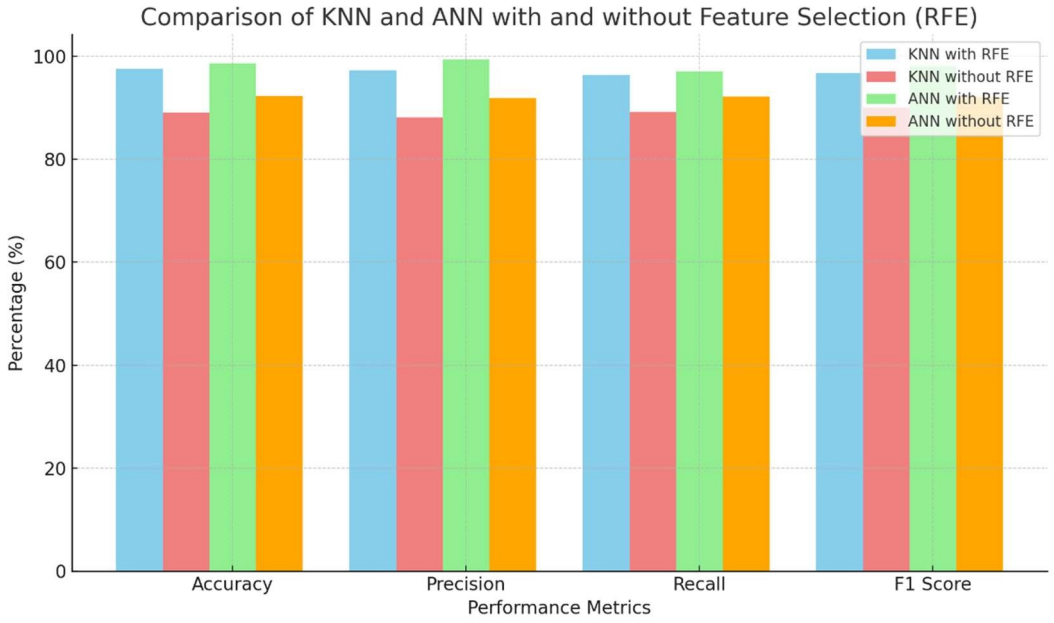
Performa ANN tanpa Seleksi Fitur

Tanpa menggunakan seleksi fitur, performa model ANN mengalami penurunan pada semua metrik evaluasi:

- **Akurasi:** 92,24%
- **Presisi:** 91,89%
- **Recall:** 92,21%
- **F1 Score:** 92,09%

Akurasi menurun menjadi 92,24%, menunjukkan bahwa model lebih sering membuat kesalahan prediksi dibandingkan dengan model yang menggunakan RFE. Presisi turun menjadi 91,89%, yang berarti model lebih sering memberikan prediksi positif yang salah. Recall sedikit lebih tinggi pada 92,21%, namun tetap lebih rendah dibandingkan dengan model yang menggunakan RFE. F1 Score sebesar 92,09% menunjukkan keseimbangan yang masih ada antara Presisi dan Recall, tetapi dengan performa yang secara keseluruhan lebih rendah dibandingkan dengan model yang menggunakan seleksi fitur.

4.3.3 Hasil Perbandingan Performa Kedua Algoritma



Gambar 4. 5 Hasil Perbandingan Algoritma

Berdasarkan hasil analisis performa antara algoritma KNN (K-Nearest Neighbors) dan ANN (Artificial Neural Network) dengan dan tanpa penggunaan seleksi fitur melalui metode RFE (Recursive Feature Elimination), terdapat perbedaan yang signifikan dalam efisiensi dan efektivitas kedua algoritma tersebut. Grafik yang dibandingkan menunjukkan bahwa ANN secara konsisten menunjukkan performa yang lebih unggul dibandingkan KNN dalam hal akurasi, presisi, recall, dan F1 Score, baik saat menggunakan seleksi fitur maupun tanpa seleksi fitur.

a) Performa dengan Seleksi Fitur (RFE)

Ketika seleksi fitur menggunakan RFE diterapkan, baik ANN maupun KNN menunjukkan peningkatan performa yang signifikan. ANN dengan RFE berhasil mencapai akurasi sebesar 98,64%, yang menunjukkan bahwa hampir semua prediksi yang dibuat oleh model ini benar. Dalam hal presisi, ANN mencapai

99,31%, yang mengindikasikan bahwa hampir semua prediksi positif yang dibuat oleh model adalah benar. Dengan recall sebesar 97,03%, ANN juga berhasil mengidentifikasi sebagian besar sampel positif yang ada, menunjukkan kemampuan deteksi yang sangat kuat. F1 Score yang mencapai 98,13% menunjukkan bahwa ANN dengan RFE memiliki keseimbangan yang sangat baik antara presisi dan recall, membuatnya sangat andal dalam klasifikasi.

Di sisi lain, KNN dengan RFE juga mengalami peningkatan performa, meskipun tidak sekuat ANN. KNN dengan RFE memiliki akurasi sebesar 97,56%, sedikit di bawah ANN. Presisi KNN tercatat sebesar 97,32%, yang berarti bahwa model ini juga cukup efektif dalam mengurangi kesalahan prediksi positif. Recall sebesar 96,29% menunjukkan bahwa KNN mampu mengidentifikasi sebagian besar sampel positif, meskipun tidak sebaik ANN. F1 Score sebesar 96,71% menunjukkan bahwa KNN dengan RFE masih mempertahankan keseimbangan yang baik antara presisi dan recall, meskipun sedikit lebih rendah daripada ANN.

b) Performa tanpa Seleksi Fitur

Tanpa penggunaan seleksi fitur, kedua algoritma mengalami penurunan performa, namun penurunan ini lebih terasa pada KNN dibandingkan ANN. ANN tanpa seleksi fitur memiliki akurasi sebesar 92,24%, yang masih cukup tinggi tetapi lebih rendah dibandingkan performa saat menggunakan RFE. Presisi ANN tanpa seleksi fitur sebesar 91,89%, menunjukkan bahwa model ini lebih sering membuat kesalahan dalam prediksi positif. Recall sebesar 92,21% menunjukkan bahwa meskipun terjadi penurunan, ANN tetap mampu mengenali sebagian besar sampel

positif yang ada. F1 Score sebesar 92,09% menunjukkan bahwa keseimbangan antara presisi dan recall tetap terjaga, meskipun performanya secara keseluruhan lebih rendah dibandingkan saat menggunakan seleksi fitur.

KNN tanpa seleksi fitur menunjukkan penurunan performa yang lebih tajam. Dengan akurasi hanya sebesar 89,03%, KNN tanpa RFE menunjukkan bahwa model ini lebih rentan terhadap kesalahan prediksi. Presisi sebesar 88,09% menunjukkan bahwa KNN lebih sering membuat prediksi positif yang salah. Meskipun recall-nya mencapai 89,21%, nilai ini tetap lebih rendah dibandingkan dengan model yang menggunakan RFE, menunjukkan bahwa KNN tanpa seleksi fitur kurang efisien dalam mendeteksi sampel positif. F1 Score sebesar 90,00% mengindikasikan bahwa meskipun ada keseimbangan antara presisi dan recall, performa keseluruhan KNN tanpa seleksi fitur tetap lebih rendah dibandingkan model yang menggunakan RFE.