IMPLEMENTASI KLASIFIKASI SUPERVISED LEARNING MENGGUNAKAN ALGORITMA RANDOM FOREST UNTUK DETEKSI HEPATITIS

Dhafa Nur Fadhillah[1], Fadli Dwi Ramadhan[2], Marcel Binggi Antonio[3]
dhafanurf@student.telkomuniversity.ac.id[1],
fadlidwiramadhan@student.telkomuniversity.ac.id[2],
marcelbinggi@student.telkomuniversity.ac.id[3]

ABSTRAK

Hepatitis merupakan salah satu penyakit yang menjadi ancaman serius bagi kesehatan global. Oleh karena itu, diperlukan penanganan khusus untuk mengatasi penyakit hepatitis ini. Penelitian ini difokuskan pada penentuan hyperparameter terbaik untuk model prediksi hepatitis menggunakan algoritma Random Forest, dengan penilaian kinerja utama menggunakan metrik akurasi. Variabel hyperparameter yang dipelajari mencakup Jumlah Pohon (*number of trees*), Kedalaman Maksimum Pohon (*maximum depth*), Jumlah Minimum Sampel yang dibutuhkan untuk membuat split pada node (*min sample split*), dan Jumlah Fitur yang dipertimbangkan pada setiap node (n *features*). Proses penelitian melibatkan langkah-langkah preprocessing data, termasuk penanganan nilai yang hilang, untuk memastikan kualitas data yang optimal. Pembagian dataset menjadi dataset pelatihan dan uji menjadi dasar evaluasi performa model. Hasil eksperimen menunjukkan bahwa konfigurasi terbaik ditemukan saat hyperparameter diatur dengan jumlah Pohon sebanyak 30, Kedalaman Maksimum sebanyak 5, Jumlah Minimum Sampel Split sebanyak 2, dan Jumlah Fitur sebanyak 6, mencapai tingkat akurasi sebesar 83.87%. Temuan ini berkontribusi sebagai panduan praktis untuk pemilihan hyperparameter optimal dalam konteks prediksi hepatitis menggunakan Random Forest, berfokus pada tujuan mencapai akurasi prediksi yang maksimal. Sehingga, penelitian ini tidak hanya memberikan solusi efektif untuk prediksi hepatitis, tetapi juga memberikan kontribusi berarti pada pemahaman dan praktik terbaik pengembangan model prediktif di berbagai bidang aplikasi.

Kata kunci: Random Forest, hyperparameter, model prediksi hepatitis.

I. PENDAHULUAN

Hepatitis adalah suatu penyakit yang menjadi ancaman signifikan bagi kesehatan global[2]. Oleh karena itu, dibutuhkan pendekatan khusus dalam penanganan penyakit hepatitis ini. Penelitian ini berfokus pada penentuan hyperparameter terbaik untuk model prediksi hepatitis menggunakan algoritma Random Forest, dan memiliki tujuan utama dalam meningkatkan kemampuan prediktif model. Pertimbangan menggunakan model ini muncul dari kebutuhan mendesak untuk memahami dan memprediksi perkembangan penyakit hepatitis, yang merupakan masalah kesehatan global dengan dampak signifikan terhadap populasi. Dengan mengimplementasikan metode machine learning, seperti Random Forest, penelitian ini diharapkan dapat memberikan kontribusi dalam peningkatan akurasi prediksi dan pemahaman mendalam terkait faktor-faktor yang mempengaruhi perkembangan hepatitis.

Dalam proses penelitian, digunakan dataset Hepatitis [1]. Tahapan preprocessing data, termasuk penanganan nilai yang hilang, dilakukan untuk memastikan kualitas optimal sebelum melibatkan data dalam pelatihan model. Dataset dibagi menjadi dataset pelatihan dan uji, dan performa model dievaluasi dengan menggunakan metrik akurasi. Selanjutnya, dilakukan eksplorasi terhadap hyperparameter seperti Jumlah Pohon (*number of trees*), Kedalaman Maksimum Pohon (*maximum depth*), Jumlah Minimum Sampel yang dibutuhkan untuk membuat split pada node (*min sample split*), dan Jumlah Fitur yang dipertimbangkan pada setiap node (n *features*).

Sebagai contoh, input model dapat berupa data pasien hepatitis yang mencakup informasi seperti Umur, Jenis Kelamin, Steroid, Antivirus, Kelelahan, Malaise, Anoreksia, Ukuran Hati, Kekerasan Hati, Teraba Limpa, Teraba Laba-laba Pembuluh Darah, Ascites, Varises, Bilirubin, Alkalin Fosfatase, SGOT, Albumin, dan Histologi. Outputnya merupakan prediksi apakah pasien memiliki penyakit hepatitis atau tidak.

Kontribusi dari penelitian ini melibatkan pengembangan model prediktif yang tidak hanya akurat tetapi juga dapat memberikan pemahaman yang lebih dalam tentang faktor-faktor yang mempengaruhi hepatitis. Model ini dapat menjadi alat yang berharga bagi praktisi kesehatan untuk mendukung diagnosis dan perencanaan perawatan. Selain itu, penelitian ini dapat membuka jalan untuk penelitian lebih lanjut dalam bidang prediksi penyakit menggunakan pendekatan machine learning, dengan harapan dapat meningkatkan upaya pencegahan dan pengelolaan penyakit hepatitis secara lebih efektif.

II. TINJAUAN PUSTAKA

Random Forest, atau Hutan Acak, merupakan suatu metode dalam machine learning yang memanfaatkan sejumlah pohon keputusan (decision trees) untuk melakukan prediksi. Dalam proses inisialisasi, beberapa hyperparameter, seperti jumlah pohon (*number of trees*), kedalaman maksimum pohon (*maximum depth*), jumlah minimum sampel yang dibutuhkan untuk membuat split pada node (*min sample split*), dan jumlah fitur yang dipertimbangkan pada setiap node (n *features*), perlu ditentukan sebelum pelatihan dimulai. Setiap pohon dalam Random Forest dibuat dengan menggunakan sampel data yang diambil secara acak dengan penggantian, menghasilkan prediksi berdasarkan aturan-aturan yang ditemuinya.

Proses pelatihan melibatkan penciptaan sejumlah pohon keputusan, dan setiap pohon dibuat dengan menggunakan sampel data acak. Prediksi akhir dari Random Forest diperoleh dengan mengambil hasil mayoritas dari prediksi semua pohon. Dengan kata lain, hasil prediksi akhir untuk suatu data baru didasarkan pada mayoritas keputusan yang diambil oleh setiap pohon.

Contoh penggunaan Random Forest melibatkan inisialisasi model, pelatihan dengan dataset pelatihan, dan akhirnya, melakukan prediksi terhadap dataset uji. Metode ini dikenal karena kemampuannya mengatasi overfitting, kehandalannya dalam berbagai situasi data, dan kemampuannya menangani dataset dengan banyak fitur. Dengan struktur pohon keputusan yang beragam, Random Forest menjadi pilihan yang kuat untuk tugas klasifikasi dan regresi dalam machine learning.

III. METODE

3.1 Data *Exploration*

Dataset yang digunakan dalam penelitian ini berasal dari UCI Machine Learning Repository dan dikenal sebagai dataset Hepatitis. Dataset Hepatitis ini terdiri dari 155 baris dan 20 kolom, yang mencakup informasi seputar Umur, Jenis Kelamin, Steroid, Antivirus, Kelelahan, Malaise, Anoreksia, Ukuran Hati, Kekerasan Hati, Teraba Limpa, Teraba Laba-laba Pembuluh Darah, Ascites, Varises, Bilirubin, Alkalin Fosfatase, SGOT, Albumin, dan

Histologi. Tipe data dalam dataset ini terdiri dari float64 dan int64.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 155 entries, 0 to 154
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Class            155 non-null    int64
 1   AGE              155 non-null    int64
 2   SEX              155 non-null    int64
 3   STEROID          154 non-null    float64
 4   ANTIVIRALS       155 non-null    int64
 5   FATIGUE          154 non-null    float64
 6   MALAISE          154 non-null    float64
 7   ANOREXIA         154 non-null    float64
 8   LIVER BIG        145 non-null    float64
 9   LIVER FIRM       144 non-null    float64
 10  SPLEEN PALPABLE  150 non-null    float64
 11  SPIDERS          150 non-null    float64
 12  ASCITES          150 non-null    float64
 13  VARICES          150 non-null    float64
 14  BILIRUBIN        149 non-null    float64
 15  ALK PHOSPHATE    126 non-null    float64
 16  SGOT             151 non-null    float64
 17  ALBUMIN          139 non-null    float64
 18  PROTIME          88 non-null     float64
 19  HISTOLOGY        155 non-null    int64
dtypes: float64(15), int64(5)
memory usage: 24.3 KB
```

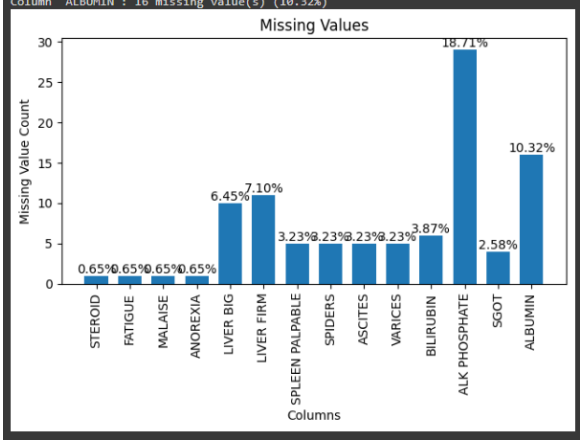
Gambar 1. Informasi umum dari dataset Hepatitis

3.2 Data *Preprocessing*

Tahap awal dilakukan dengan menghapus fitur-fitur yang dianggap kurang relevan atau memiliki data yang kurang jelas. Sebagai contoh, fitur "protime" dihapus karena isi datanya dianggap kurang jelas atau ambigu.

Tahap berikutnya adalah melakukan pemeriksaan terhadap data kosong pada dataset. Jika terdapat sejumlah besar data yang kosong, opsi yang mungkin adalah menghapus fitur yang terkait atau mengisi nilai kosong dengan rata-rata dari kolom fitur tersebut. Dalam kasus ini, hasil pemeriksaan menunjukkan bahwa dataset tidak mengandung data yang kosong.

```
Column 'STEROID': 1 missing value(s) (0.65%)
Column 'FATIGUE': 1 missing value(s) (0.65%)
Column 'MALAISE': 1 missing value(s) (0.65%)
Column 'ANOREXIA': 1 missing value(s) (0.65%)
Column 'LIVER BIG': 10 missing value(s) (6.45%)
Column 'LIVER FIRM': 11 missing value(s) (7.10%)
Column 'SPLEEN PALPABLE': 5 missing value(s) (3.23%)
Column 'SPIDERS': 5 missing value(s) (3.23%)
Column 'ASCITES': 5 missing value(s) (3.23%)
Column 'VARICES': 5 missing value(s) (3.23%)
Column 'BILIRUBIN': 6 missing value(s) (3.87%)
Column 'ALK PHOSPHATE': 29 missing value(s) (18.71%)
Column 'SGOT': 4 missing value(s) (2.58%)
Column 'ALBUMIN': 16 missing value(s) (10.32%)
```



Gambar 2. Persentase nilai null pada fitur di dataset

Kolom yang memiliki nilai non-binary akan diisi dengan nilai rata-rata dari kolom tersebut, sementara kolom binary akan diisi dengan nilai yang paling sering muncul pada kolom tersebut.

```
# Buat list column non binary
column_non_binary = ['ALK PHOSPHATE', 'SGOT', 'ALBUMIN', 'AGE']

# Buat list column binary
column_binary = [col for col in df.columns if col not in column_non_binary]

# Isi column non binary dengan mean column tersebut
for col in column_non_binary:
    df[col] = df[col].fillna(df[col].mean())

# Isi column binary dengan value paling sering muncul di column tersebut
for col in column_binary:
    df[col] = df[col].fillna(df[col].mode()[0])

# Check ulang missing values
check_missing_values(df)
```

```
Tidak ada missing values pada Data Frame ini
```

Gambar 3. Menangani fitur dengan nilai null

Selanjutnya, dilakukan pengubahan nilai 2 menjadi 1 (true) dan nilai 1 menjadi 0 (false) pada kolom binary. Hal ini dilakukan untuk konsistensi interpretasi dan kejelasan representasi nilai pada kolom binary, di mana nilai 1 dan 2 mungkin awalnya memiliki makna kategorikal tertentu yang lebih baik direpresentasikan sebagai 0 dan 1. Dengan demikian, perubahan ini membantu dalam interpretasi dan analisis data yang lebih mudah.

```
# Membuat dictionary mapping value yang diubah
value_mapping = {1: 0, 2: 1}

# looping kolom binary dan isi dengan value_mapping
for column in df.columns:
    if column not in column_non_binary:
        df[column] = df[column].replace(value_mapping)
```

Gambar 4. Mengubah nilai pada fitur binary

Langkah selanjutnya adalah mengubah fitur non-binary menjadi biner, kecuali untuk fitur *AGE*. Proses ini dilakukan karena beberapa algoritma machine learning lebih efektif atau lebih mudah beroperasi dengan fitur biner (1/0). Pengubahannya melibatkan konversi nilai-nilai non-binary menjadi dua kategori, normal(1) dan tidak normal(2) sesuai jangkauan fitur

Penting untuk mencatat bahwa fitur *AGE* tidak diubah menjadi biner karena sifat intrinsiknya yang bersifat numerik dan dapat mewakili rentang nilai kontinu.

Untuk column 'BILIRUBIN' jika range nya 0,1 - 1,2 mg/dL termasuk normal(1) dan diluar range itu tidak normal(0)

```
[591] # Buat batasan normal
      min = 0.1
      max = 1.2

      # Update the 'BILIRUBIN' column based on the range condition
      df['BILIRUBIN'] = np.where((df['BILIRUBIN'] >= min) & (df['BILIRUBIN'] <= max), 1, 0)
```

Untuk column 'ALK PHOSPHATE' jika range nya 44 - 147 IU/L termasuk normal(1) dan diluar range itu tidak normal(0)

```
# Buat batasan normal
min = 44
max = 147

# Update the 'BILIRUBIN' column based on the range condition
df['ALK PHOSPHATE'] = np.where((df['ALK PHOSPHATE'] >= min) & (df['ALK PHOSPHATE'] <= max), 1, 0)
```

Gambar 5. Perubahan nilai pada fitur BILIRUBIN dan ALK PHOSPHATE

Untuk column 'SGOT':

- Range normal(1) untuk laki-laki adalah 8 - 50 U/L
- Range normal(1) untuk perempuan adalah 8 - 45 U/L
- Selain range itu tidak normal(0).

```
# Define the normal range for 'SGOT'
min_male = 8
max_male = 50
min_female = 8
max_female = 45

# Update the 'SGOT' column based on the range condition
df['SGOT'] = np.where(
    (df['SEX'] == 0) & (df['SGOT'] >= min_male) & (df['SGOT'] <= max_male) |
    (df['SEX'] == 1) & (df['SGOT'] >= min_female) & (df['SGOT'] <= max_female),
    1, 0
)
```

Untuk column 'ALBUMIN' jika range nya 3,5 - 5,5 g/dL termasuk normal(1) dan diluar range itu tidak normal(0)

```
[594] # Buat batasan normal
      min = 3.5
      max = 5.5

      # Update the 'BILIRUBIN' column based on the range condition
      df['ALBUMIN'] = np.where((df['ALBUMIN'] >= min) & (df['ALBUMIN'] <= max), 1, 0)
```

Gambar 6. Perubahan nilai pada fitur SGOT dan ALBUMIN

Selanjutnya, dilakukan pengecekan keberadaan duplikasi pada baris-baris data. Penghapusan baris yang duplikat diperlukan untuk memastikan keunikan dan ketelitian data dalam dataset. Baris duplikat dapat menyebabkan bias yang tidak diinginkan dalam hasil analisis dan kinerja model machine learning. Dengan menghilangkan duplikasi, kita memastikan setiap baris memberikan kontribusi unik dan proporsional pada model, meningkatkan integritas dan validitas dataset.

```
# Function untuk handle data duplikat
def handle_duplicates(df):
    # Periksa data duplikat
    duplicates = df[df.duplicated()]

    # Jika ada data duplikat, cetak data tersebut dan hapus duplikat dari DataFrame
    if not duplicates.empty:
        print("Data duplikat yang ditemukan:")
        print(duplicates)
        df.drop_duplicates(inplace=True)  # Drop semua rows yang mengandung duplikat
        print("Data duplikat telah dihapus.")
    else:
        print("Tidak ada Duplicate values pada Data Frame ini")

# Panggil function
handle_duplicates(df)
```

```
Data duplikat yang ditemukan:
    Class  AGE  SEX  STEROID  ANTIVIRALS  FATIGUE  MALAISE  ANOREXIA  \
44      1   34    0      1.0           1      1.0      1.0       1.0

    LIVER BIG  LIVER FIRM  SPLEEN PALPABLE  SPIDERS  ASCITES  VARICES  \
44        1.0         1.0              1.0      1.0      1.0      1.0

    BILIRUBIN  ALK PHOSPHATE  SGOT  ALBUMIN  HISTOLOGY
44          0              1     0        1          0
Data duplikat telah dihapus.
```

Gambar 7. Menangani baris duplikat

Tahap terakhir, yaitu membagi data menjadi fitur (X) dan target (Y), dilakukan karena memisahkan variabel-variabel yang akan digunakan sebagai masukan model (fitur) dan variabel yang akan diprediksi oleh model (target). Dengan membagi data ini, proses pelatihan model machine learning dapat dilakukan dengan lebih efisien. Fitur (X) akan menjadi input atau atribut yang digunakan oleh model untuk membuat prediksi, sedangkan target (Y) adalah variabel yang ingin diprediksi oleh model.

```
[597] X = df.iloc[:, 1:].values
      Y = df.iloc[:, 0].values
```

```
X
```

```
array([[30.,  1.,  0., ...,  1.,  1.,  0.],
       [50.,  0.,  0., ...,  1.,  1.,  0.],
       [78.,  0.,  1., ...,  1.,  1.,  0.],
       ...,
       [61.,  0.,  0., ...,  1.,  1.,  1.],
       [53.,  1.,  0., ...,  1.,  1.,  1.],
       [43.,  0.,  1., ...,  1.,  0.,  1.]])
```

```
[599] Y
```

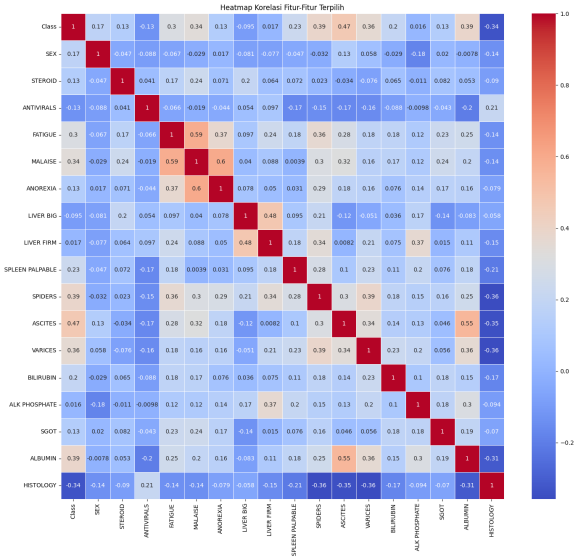
```
array([1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0,
       1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1,
       0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0])
```

Gambar 8. Pembagian fitur(X) dan target(Y)

3.3 Data Visualization

Berikut adalah visualisasi menggunakan *heatmap* untuk melihat korelasi antar fitur pada *dataset*. Didapatkan

bahwa fitur yang sangat berkorelasi dengan fitur target (*Class*) adalah fitur *Ascites*.



Gambar 9. Heatmap korelasi antar fitur

3.4 Membuat Model *Baseline*

Baseline model dibuat dengan menerapkan seluruh tahapan proses pada tinjauan pustaka. Kemudian untuk *hyperparameter* dari *Baseline* model adalah sebagai berikut:

Number of trees	10
Maximum depth	5
Min samples split	2
n *features*	2

Tabel 1. Hyperparameter Baseline model

3.5 Model *Exploration*

Pada eksplorasi model, dilakukan perubahan satu *hyperparameter* dari tiap model, pada *Exploration* model 1 dilakukan perubahan pada jumlah *Maximum depth*, pada *Exploration* model 2 dilakukan perubahan pada jumlah *Min sample split*, kemudian pada *Exploration* model 3 dilakukan perubahan pada jumlah *n features*.

Number of trees	10
Maximum depth	**8**
Min samples split	2
n *features*	2

Tabel 2. Hyperparameter Exploration model 1

Number of trees	10
Maximum depth	5
Min samples split	**3**
n *features*	2

Tabel 3. Hyperparameter Exploration model 2

Number of trees	10
Maximum depth	5
Min samples split	2
n *features*	**3**

Tabel 4. Hyperparameter Exploration model 3

IV. HASIL DAN DISKUSI

Setelah dilakukan pembuatan model *Baseline* dan melakukan eksplorasi model, didapatkan akurasi dari setiap model adalah sebagai berikut:

```
# BaseModel
numberOfTrees = 10
maximumDepth = 5
minSamplesSplit = 2
nFeatures = 2

rf1 = RandomForest(numberOfTrees, maximumDepth, minSamplesSplit, nFeatures)
rf1.fit(x_train, y_train)
predictions = rf1.predict(x_test)

acc = calc_accuracy(y_test, predictions)
print("akurasi =", acc)
```

```
akurasi = 0.7419354838709677
```

Gambar 13. Akurasi base model

```
# Model 1
numberOfTrees = 10
maximumDepth = 8
minSamplesSplit = 2
nFeatures = 2

rf2 = RandomForest(numberOfTrees, maximumDepth, minSamplesSplit, nFeatures)
rf2.fit(x_train, y_train)
predictions = rf2.predict(x_test)

acc = calc_accuracy(y_test, predictions)
print("akurasi =", acc)
```

```
akurasi = 0.7419354838709677
```

Gambar 14. Akurasi model 1

```
# Model 2
numberOfTrees = 10
maximumDepth = 5
minSamplesSplit = 3
nFeatures = 2

rf3 = RandomForest(numberOfTrees, maximumDepth, minSamplesSplit, nFeatures)
rf3.fit(x_train, y_train)
predictions = rf3.predict(x_test)

acc = calc_accuracy(y_test, predictions)
print("akurasi =", acc)
```

```
akurasi = 0.7096774193548387
```

Gambar 15. Akurasi model 2

```
# Model 3
numberOfTrees = 10
maximumDepth = 5
minSamplesSplit = 2
nFeatures = 3

rf4 = RandomForest(numberOfTrees, maximumDepth, minSamplesSplit, nFeatures)
rf4.fit(x_train, y_train)
predictions = rf4.predict(x_test)

acc = calc_accuracy(y_test, predictions)
print("akurasi =", acc)

akurasi = 0.6774193548387096
```

Gambar 16. Akurasi model 3

```
# Membuat objek RFHyperparameterExplorer dengan data pelatihan dan pengujian serta nilai hyperparameter yang akan dieksplorasi
eksplorasi = RFHyperparameterExplorer(x_train, y_train, x_test, y_test,
                                      n_trees_values=[30, 60, 100],
                                      max_depth_values=[5, 10, 15],
                                      min_samples_split_values=[2, 3, 5],
                                      n_features_values=[2, 4, 6])
# Melakukan eksplorasi hyperparameter
results = eksplorasi.explore_hyperparameters()

# Print hasilnya
for result in results:
    print(result)
```

Gambar 17. Eksplorasi model

```
# Print model terbaik
eksplorasi.print_best_result()

Hyperparameter Terbaik:
numberOfTrees    : 30
maximumDepth     : 5
minSamplesSplit  : 2
nFeatures        : 6
Accuracy         : 0.8387096774193549
```

Gambar 18. Akurasi eksplorasi model

V. KESIMPULAN

Penelitian ini menghasilkan model *Random Forest* terbaik untuk dataset Hepatitis dengan konfigurasi *Number of trees* 30, *Maximum depth* 5, *Min sample Split* 2, dan n *features* 6. Hasil evaluasi menunjukkan tingkat akurasi sebesar 83.87%, mencerminkan kemampuan model dalam mengatasi kompleksitas dan variasi data Hepatitis. Analisis parameter menunjukkan bahwa jumlah pohon keputusan yang moderat, batas kedalaman, sensitivitas terhadap detail kecil, dan pertimbangan enam fitur pada setiap langkah adalah kunci keberhasilan model ini. Meskipun demikian, keberlakuan hasil ini berdasarkan pada karakteristik khusus dataset Hepatitis, dan penggunaan model perlu dipertimbangkan dengan cermat sesuai dengan sifat unik dari setiap dataset yang berbeda. Kontribusi penelitian ini tidak hanya terletak pada pengembangan model yang handal untuk prediksi Hepatitis, tetapi juga memberikan wawasan berharga terhadap penyesuaian parameter yang optimal, memberikan landasan untuk penelitian lebih lanjut dalam pemodelan penyakit infeksi lainnya dan memberikan panduan bagi praktisi kesehatan dalam pengambilan keputusan di bidang diagnosis dan manajemen penyakit Hepatitis.

DAFTAR PUSTAKA

[1] Hepatitis. UCI Machine Learning Repository, 1988. DOI: https://doi.org/10.24432/C5Q59J.

[2] Zuckerman AJ. Hepatitis Viruses. In: Baron S, editor. Medical Microbiology. 4th ed. Galveston (TX): University of Texas Medical Branch at Galveston; 1996. Chapter 70. PMID: 21413272.

Link Collab: https://colab.research.google.com/drive/1QW-6IeJkJnBQasaRH-igO9SoPuopFVg3?usp=sharing#scrollTo=0v4NsZ5T3Hje