

TUGAS BESAR
PRAKTIKUM KECERASAN BUATAN
“Klasifikasi Liver Disease Menggunakan KNN”

Dosen Pengampu : Leni Fitriani ST. M.Kom



Disusun Oleh:

Shadam Alfito Kharisma Putra 2106011

Muhamad Ridwan F 2106081

Paschal Hendryawan 2106034

PRODI TEKNIK INFORMATIKA
INSTITUT TEKNOLOGI GARUT
2023

KATA PENGANTAR

Alhamdulillahirabbil'alamiin, Puji dan Syukur saya panjatkan kepada Allah SWT atas segala limpahan anugerah dan rahmat-Nya yang diberikan, sehingga dalam perancangan tugas besar dengan judul "Klasifikasi Penyakit Hati menggunakan metode KNN". dapat diselesaikan dengan cukup baik. Shalawat serta salam selalu tercurah kepada junjungan kita Nabi Muhammad SAW, beserta keluarga, para sahabat, dan paranabi'in yang selalu berjuang di jalan Allah SWT untuk menegakkan Islam hingga akhir zaman.

Laporan ini disusun sebagai salah satu syarat untuk memenuhi tugas besar Praktikum Kecerdasan Buatan, Program Studi Teknik Informatika Institut Teknologi Garut. Perancang tugas laporan ini, menyadari bahwa laporan ini masih jauh dari kata sempurna. Oleh karena itu, perancang sangat mengharapkan kritik dan saran yang membangun dalam rangka memperbaiki laporan ini. Semoga laporan ini dapat bermanfaat bagi seluruh pihak yang bersangkutan.

DAFTAR ISI

KATA PENGANTAR.....	2
DAFTAR ISI.....	3
BAB I. LATAR BELAKANG	4
A. Identifikasi Masalah	4
1. Masalah yang Dihadapi	4
B. Tujuan Penelitian	4
1. Relevansi Penelitian	4
C. Batasan Masalah	4
1. Dataset	4
2. Metode Klasifikasi	4
3. Target Kelas.....	5
4. Evaluasi Model	5
5. Waktu dan Sumber Daya	5
D. Manfaat Penelitian.....	5
1. Kontribusi Ilmiah	5
2. Peningkatan Diagnostik.....	5
3. Efisiensi dan Biaya.....	5
4. Diagnosis Dini dan Perawatan yang Tepat.....	6
5. Pengembangan Lanjutan	6
6. Ruang Lingkup Penelitian.....	6
BAB II PENELITIAN SEBELUMNYA DAN GAP ANALYSIS	7
A. Sumber Dataset.....	7
B. Informasi Fitur.....	7
C. Tujuan Klasifikasi.....	7
D. Penyusunan Data Training dan Testing.....	7
E. Gap Analysis	8
BAB III METODE YANG DIGUNAKAN	9
IV HASIL DAN ANALISIS	11
BAB V KESIMPULAN	27
BAB VI DAFTAR PUSTAKA.....	28

BAB I. LATAR BELAKANG

A. Identifikasi Masalah

Penyakit hati (liver) merupakan salah satu masalah kesehatan yang serius di seluruh dunia. Diagnosis dini penyakit hati sangat penting untuk pengobatan yang efektif dan penyelamatan nyawa pasien. Dalam konteks ini, penggunaan teknik kecerdasan buatan dapat memberikan kontribusi yang signifikan dalam pengidentifikasian dini pasien yang berisiko menderita penyakit hati.

1. Masalah yang Dihadapi

Dalam kasus klasifikasi Indian Liver Patient Dataset, terdapat beberapa masalah yang dihadapi:

- 1) Identifikasi Pasien: Penentuan apakah seorang pasien menderita penyakit hati atau tidak berdasarkan fitur-fitur klinis yang ada dalam dataset.
- 2) Keterbatasan Diagnosis Konvensional: Diagnosis penyakit hati dapat memerlukan waktu dan biaya yang cukup tinggi, sementara hasilnya tidak selalu akurat. Oleh karena itu, diperlukan metode yang dapat membantu dalam melakukan diagnosis yang lebih cepat dan akurat.

B. Tujuan Penelitian

Penelitian ini bertujuan untuk menerapkan metode kecerdasan buatan dalam klasifikasi Indian Liver Patient Dataset. Dataset ini mengandung informasi medis mengenai pasien yang didiagnosis dengan penyakit hati (liver). Dalam tugas ini, kami akan menggunakan teknik KNearestNeighbor untuk mengklasifikasikan pasien menjadi dua kelompok: pasien yang menderita penyakit hati dan pasien yang tidak menderita penyakit hati. Melalui penelitian ini, kami berharap dapat membangun model klasifikasi yang dapat membantu dalam diagnosis dini dan pengambilan keputusan medis terkait penyakit hati[1].

1. Relevansi Penelitian

Identifikasi dini pasien yang berisiko menderita penyakit hati dapat membantu praktisi medis dalam melakukan tindakan pencegahan dan pengobatan yang tepat waktu. Dengan menggunakan metode KNN, diharapkan dapat meningkatkan akurasi dan efisiensi dalam proses diagnosis penyakit hati.

C. Batasan Masalah

Dalam penelitian ini, terdapat beberapa batasan yang perlu diperhatikan:

1. Dataset

- 1) Dataset yang digunakan adalah Indian Liver Patient Dataset, yang terdiri dari fitur-fitur klinis yang relevan dengan penyakit hati.
- 2) Dataset tidak mengandung nilai yang hilang. Jika terdapat nilai yang hilang, langkah-langkah imputasi atau penghapusan yang tepat telah dilakukan sebelumnya.

2. Metode Klasifikasi

- 1) Metode klasifikasi yang digunakan adalah K-Nearest Neighbors (KNN).

- 2) Implementasi KNN menggunakan algoritma yang tersedia dalam library atau framework yang relevan.
- 3) Pemilihan nilai K yang optimal menggunakan metode validasi silang atau teknik tuning parameter yang sesuai.

3. Target Kelas

- 1) Klasifikasi dilakukan untuk mengidentifikasi pasien sebagai "menderita penyakit hati" atau "tidak menderita penyakit hati".
- 2) Klasifikasi tidak memperdulikan jenis atau tingkat keparahan penyakit hati yang mungkin dialami pasien.

4. Evaluasi Model

- 1) Evaluasi model menggunakan metrik-metrik evaluasi klasifikasi, seperti akurasi, presisi, recall, dan F1-score.
- 2) Evaluasi dilakukan menggunakan data testing yang telah dipisahkan sebelumnya.

5. Waktu dan Sumber Daya

- 1) Penelitian ini dilakukan dengan memperhatikan batasan waktu yang ditentukan.
- 2) Penelitian ini dilakukan dengan menggunakan sumber daya yang tersedia dalam lingkungan belajar.

D. Manfaat Penelitian

1. Kontribusi Ilmiah

Penelitian ini memiliki kontribusi ilmiah yang penting dalam bidang klasifikasi dan penggunaan kecerdasan buatan dalam diagnosis penyakit hati. Dengan menerapkan metode K-Nearest Neighbors (KNN) pada dataset Indian Liver Patient, penelitian ini menyediakan wawasan baru tentang penggunaan teknik klasifikasi dalam mengidentifikasi pasien yang berisiko menderita penyakit hati secara dini.

2. Peningkatan Diagnostik

Penelitian ini memberikan manfaat langsung dalam meningkatkan diagnosis penyakit hati. Dengan menggunakan model KNN yang telah dilatih, praktisi medis dapat mendapatkan prediksi awal mengenai kondisi pasien berdasarkan fitur-fitur klinis yang terdapat dalam dataset. Hal ini dapat membantu dalam pengambilan keputusan medis yang lebih akurat dan tepat waktu.

3. Efisiensi dan Biaya

Dengan menggunakan metode klasifikasi KNN, penelitian ini juga dapat memberikan manfaat dalam hal efisiensi dan penghematan biaya. Diagnosis yang cepat dan akurat dapat mengurangi waktu dan biaya yang diperlukan untuk tes tambahan atau prosedur yang lebih invasif. Hal ini dapat meningkatkan efisiensi pelayanan kesehatan dan mengurangi beban finansial bagi pasien.

4. Diagnosis Dini dan Perawatan yang Tepat

Melalui klasifikasi Indian Liver Patient Dataset, penelitian ini berpotensi memberikan manfaat dalam diagnosis dini penyakit hati. Dengan mendeteksi penyakit hati pada tahap awal, tindakan pencegahan dan perawatan yang tepat dapat segera dilakukan. Hal ini dapat meningkatkan peluang kesembuhan, mengurangi komplikasi, dan meningkatkan kualitas hidup pasien.

5. Pengembangan Lanjutan

Penelitian ini juga dapat menjadi dasar dan motivasi bagi penelitian dan pengembangan lebih lanjut dalam bidang penggunaan kecerdasan buatan dalam diagnosis penyakit hati. Dengan meningkatnya kesadaran akan potensi teknologi ini, dapat terjadi kemajuan lebih lanjut dalam pengembangan model klasifikasi yang lebih canggih dan akurat.

6. Ruang Lingkup Penelitian

Dalam penelitian ini, kami akan fokus pada klasifikasi Indian Liver Patient Dataset. Dataset ini terdiri dari sejumlah fitur klinis seperti usia, jenis kelamin, tingkat serum enzim hati, bilirubin, albumin, dan lain-lain. Kami akan melakukan proses preprocessing data, ekstraksi fitur, dan pemilihan algoritma klasifikasi yang sesuai. Setelah itu, kami akan melatih model menggunakan data training dan mengevaluasi performanya menggunakan data testing. Namun, kami menyadari bahwa laporan ini memiliki keterbatasan dan tidak menggantikan diagnosis medis profesional. Kami juga akan memberikan interpretasi hasil klasifikasi dan memberikan rekomendasi untuk pengembangan lebih lanjut.

BAB II

PENELITIAN SEBELUMNYA DAN GAP ANALYSIS

A. Sumber Dataset

Indian Liver Patient Dataset adalah kumpulan data medis yang dikumpulkan dari pasien yang didiagnosis dengan penyakit hati (liver) di salah satu rumah sakit di India. Dataset ini dapat diakses melalui situs:
<https://www.kaggle.com/datasets/jeevannagaraj/indian-liver-patient-dataset>.

B. Informasi Fitur

Dataset ini terdiri dari sejumlah fitur klinis yang dikumpulkan dari pasien. Beberapa fitur yang terdapat dalam dataset ini antara lain:

- 1) Umur: Usia pasien dalam tahun.
- 2) Jenis Kelamin: Jenis kelamin pasien (misalnya, pria atau wanita).
- 3) Total Bilirubin: Tingkat bilirubin dalam darah pasien.
- 4) Direct Bilirubin: Tingkat bilirubin langsung dalam darah pasien.
- 5) Alkali Fosfatase: Tingkat enzim alkali fosfatase dalam darah pasien.
- 6) Alamine Aminotransferase (ALT): Tingkat enzim alamine aminotransferase dalam darah pasien.
- 7) Aspartate Aminotransferase (AST): Tingkat enzim aspartate aminotransferase dalam darah pasien.
- 8) Total Protein: Konsentrasi protein total dalam darah pasien.
- 9) Albumin: Konsentrasi albumin dalam darah pasien.
- 10) Rasio Albumin/Globulin: Rasio konsentrasi albumin terhadap globulin dalam darah pasien.

C. Tujuan Klasifikasi

Tujuan dari klasifikasi Indian Liver Patient Dataset adalah untuk mengidentifikasi apakah seorang pasien menderita penyakit hati (liver) atau tidak berdasarkan fitur-fitur klinis yang ada dalam dataset. Hal ini dapat membantu dalam diagnosis dini penyakit hati dan memberikan informasi penting bagi praktisi medis dalam pengambilan keputusan.

D. Penyusunan Data Training dan Testing

Dataset ini telah dibagi menjadi data training dan data testing untuk keperluan evaluasi model klasifikasi. Bagian data training digunakan untuk melatih model klasifikasi, sedangkan data testing digunakan untuk menguji kinerja model yang telah dilatih. Rasio pembagian data training dan testing dapat disesuaikan sesuai dengan kebutuhan penelitian[2].

E. Gap Analysis

Judul	Cakupan	Kesenjangan
KLASIFIKASI LIVER DISEASE MENGGUNAKAN KNN	<p>Hati merupakan salah satu organ tubuh manusia yang paling penting. Fungsi hati adalah sebagai detoksifikasi racun yang ada dalam tubuh manusia serta mengendalikan kolesterol dan lemak yang ada dalam tubuh manusia. Jika organ hati mengalami kerusakan maka kesehatan pun akan terganggu, bahkan bisa mengalami kematian</p> <p>Beberapa penyakit yang menyerang hati salah satunya adalah Hepatitis. Menurut data WHO virus Hepatitis B menyerang 350 juta orang di dunia terutama Asia Tenggara dan Afrika yang Menyebabkan kematian sebanyak 1,2 juta pertahun</p>	<p>Penelitian ini mencoba untuk mengolah dataset yang diambil dari database UCI Machine Learning Repository, yaitu Indian Liver Patient Dataset (ILPD Dataset) , dari dataset tersebut akan diolah menggunakan Algoritma C4.5 untuk mengetahui variabel mana yang paling berpengaruh dalam mendeteksi penyakit liver.</p>

BAB III

METODE YANG DIGUNAKAN

A. Preprocessing Data

Dalam penelitian ini kami menggunakan metode KNN. Sebelum menerapkan metode K-Nearest Neighbors (KNN), data perlu melalui tahap preprocessing untuk memastikan kualitas dan keseragaman. Langkah-langkah preprocessing yang dapat dilakukan meliputi:

Penanganan data yang hilang: Memeriksa apakah dataset mengandung nilai yang hilang dan memutuskan apakah akan menghapus baris atau mengisi nilai yang hilang dengan teknik imputasi data yang sesuai.

Standarisasi fitur: Mengubah skala fitur agar memiliki mean nol dan standar deviasi satu. Hal ini penting karena KNN menghitung jarak antara data.

Pemisahan data input dan target: Memisahkan fitur-fitur yang digunakan sebagai input (X) dan target (y) yang akan diprediksi.

B. Ekstraksi Fitur

Pada metode KNN, tidak diperlukan tahap ekstraksi fitur yang kompleks, karena KNN bekerja langsung pada data asli yang telah dipreprocessed.

C. Pemilihan Algoritma Klasifikasi

KNN adalah metode klasifikasi yang sederhana dan intuitif. Algoritma KNN memprediksi kelas dari suatu data dengan mencari K tetangga terdekat berdasarkan jarak euclidean atau metrik lainnya, dan kemudian mengambil mayoritas kelas dari tetangga-tetangga tersebut. Pemilihan nilai K yang optimal dapat dilakukan dengan menggunakan metode validasi silang (cross-validation) atau teknik tuning parameter lainnya.

D. Pelatihan Model

Setelah memilih nilai K yang optimal, model KNN dilatih dengan menggunakan data training. Pada tahap pelatihan, model akan "mengingat" data training agar dapat melakukan prediksi pada data baru.

E. Evaluasi Model

Setelah pelatihan model, tahap evaluasi dilakukan dengan menggunakan data testing yang telah dipisahkan sebelumnya. Performa model dapat diukur dengan berbagai metrik evaluasi seperti akurasi, presisi, recall, F1-score, atau matriks kebingungan

(confusion matrix). Evaluasi ini membantu mengevaluasi kemampuan model dalam melakukan klasifikasi pada dataset Indian Liver Patient.[3]

Berikut adalah bagian hasil dan analisis yang dapat digunakan dalam laporan klasifikasi Indian Liver Patient Dataset menggunakan metode K-Nearest Neighbors (KNN):

IV HASIL DAN ANALISIS

1. Inputan Library Dataset

 Commi


File Edit View Insert Runtime Tools Help [Last edited on June 12](#)

+ Code + Text

Kodingan ini digunakan untuk mencetak path file-file yang ada dalam direktori `'/kaggle/input'` menggunakan modul `numpy`, `pandas`, dan `os`.

```
# import library
import numpy as np
import pandas as pd
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

2. DATA ANALYSIS

 Data Analysis

Kodingan ini digunakan untuk membaca file CSV dengan nama `'Indian Liver Patient Dataset (ILPD).csv'` dan menyimpan datanya ke dalam variabel `'data'` menggunakan modul `pandas`.

```
[ ] data = pd.read_csv('Indian Liver Patient Dataset (ILPD).csv')
```

Kodingan ini digunakan untuk menampilkan beberapa baris pertama dari dataframe `'data'` menggunakan metode `'head()'` dari modul `pandas`.

```
[ ] data.head()
```

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
0	65	Female	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	62	Male	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	62	Male	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	58	Male	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	72	Male	3.9	2.0	195	27	59	7.3	2.4	0.40	1

Kodingan ini digunakan untuk menampilkan beberapa baris terakhir dari dataframe `'data'` menggunakan metode `'tail()'` dari modul `pandas`.

```
[ ] # melihat bagian tail
data.tail()
```

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
578	60	Male	0.5	0.1	500	20	34	5.9	1.6	0.37	2
579	40	Male	0.6	0.1	98	35	31	6.0	3.2	1.10	1
580	52	Male	0.8	0.2	245	48	49	6.4	3.2	1.00	1
581	31	Male	1.3	0.5	184	29	32	6.8	3.4	1.00	1
582	38	Male	1.0	0.3	216	21	24	7.3	4.4	1.50	2

Kodingan ini digunakan untuk menampilkan informasi singkat tentang dataframe 'data', termasuk jumlah kolom, tipe data, dan jumlah non-null values pada setiap kolom menggunakan metode 'info()' dari modul pandas.

```
[ ] # Melihat informasi yang terkandung dalam data
data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
#   Column              Non-Null Count  Dtype  
---  -
0    age                  583 non-null   int64  
1    gender               583 non-null   object  
2    tot_bilirubin        583 non-null   float64 
3    direct_bilirubin     583 non-null   float64 
4    tot_proteins         583 non-null   int64  
5    albumin              583 non-null   int64  
6    ag_ratio             583 non-null   int64  
7    sgpt                 583 non-null   float64 
8    sgot                 583 non-null   float64 
9    alkphos              579 non-null   float64 
10   is_patient           583 non-null   int64  
dtypes: float64(5), int64(5), object(1)
memory usage: 50.2+ KB
```

Kodingan ini digunakan untuk menghitung jumlah nilai yang hilang (null) pada setiap kolom dalam dataframe 'data' menggunakan metode 'isnull().sum()' dari modul pandas.

```
[ ] data.isnull().sum()

age                0
gender             0
tot_bilirubin      0
direct_bilirubin   0
tot_proteins       0
albumin            0
ag_ratio           0
sgpt               0
sgot               0
alkphos            4
is_patient         0
dtype: int64
```

3. Imputation using SkLearns Simple Imputer for Missing Values

+ Code + Text

▼ Imputation using SkLearns Simple Imputer for Missing Values

Kodingan ini mengimport SimpleImputer dari modul sklearn untuk menggantikan nilai yang hilang (NaN) dengan nilai rata-rata menggunakan strategi 'mean'.

```
[ ] from sklearn.impute import SimpleImputer
imp = SimpleImputer(missing_values=np.nan, strategy='mean')
```

Kodingan ini menggantikan nilai yang hilang (NaN) pada kolom 'alkphos' dalam dataframe 'data' dengan nilai rata-rata menggunakan objek imputer 'imp' yang telah di-fit-transform.

```
[ ] #Imputing values
data['alkphos']=imp.fit_transform(data[['alkphos']])
```

Kodingan ini digunakan untuk menghitung jumlah nilai yang hilang (null) pada setiap kolom dalam dataframe 'data' menggunakan metode 'isnull().sum()' dari modul pandas.

```
[ ] data.isnull().sum()

age                0
gender             0
tot_bilirubin      0
direct_bilirubin   0
tot_proteins       0
albumin            0
ag_ratio           0
sgpt               0
sgot               0
alkphos            0
is_patient         0
dtype: int64
```

Kodingan ini menghasilkan ringkasan statistik deskriptif dari dataframe 'data', termasuk jumlah data, nilai rata-rata, standar deviasi, nilai minimum dan maksimum, serta kuartil dari setiap kolom menggunakan metode 'describe()' dari modul pandas.

```
[ ] data.describe()
```

	age	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
count	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000
mean	44.746141	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852	0.947064	1.286449
std	16.189833	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	0.318492	0.452490
min	4.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000	1.000000
25%	33.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000	1.000000
50%	45.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.947064	1.000000
75%	58.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	1.100000	2.000000
max	90.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000	2.000000

4. Age has been excluded as it is a String Value and not a Number

Kodingan ini mengubah nilai pada kolom 'gender' dalam dataframe 'data' menjadi 1 jika nilainya adalah 'Male', dan menjadi 0 jika nilainya bukan 'Male', menggunakan fungsi lambda.

```
[ ] # Replacing Male and Female with 1 and 0
data['gender'] = data['gender'].apply(lambda x:1 if x == 'Male' else 0)
```

Kodingan ini digunakan untuk menampilkan informasi singkat tentang dataframe 'data', termasuk jumlah kolom, tipe data, dan jumlah non-null values pada setiap kolom menggunakan metode 'info()' dari modul pandas.

```
[ ] #Now looking at the info
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 583 entries, 0 to 582
Data columns (total 11 columns):
#   Column              Non-Null Count  Dtype
---  -
0   age                  583 non-null    int64
1   gender               583 non-null    int64
2   tot_bilirubin        583 non-null    float64
3   direct_bilirubin     583 non-null    float64
4   tot_proteins         583 non-null    int64
5   albumin              583 non-null    int64
6   ag_ratio             583 non-null    int64
7   sgpt                 583 non-null    float64
8   sgot                 583 non-null    float64
9   alkphos              583 non-null    float64
10  is_patient           583 non-null    int64
dtypes: float64(5), int64(6)
```

Kodingan ini digunakan untuk menghitung jumlah nilai yang hilang (null) pada setiap kolom dalam dataframe 'data' menggunakan metode 'isnull().sum()' dari modul pandas.

```
[ ] data.isnull().sum()
```

```
age                0
gender             0
tot_bilirubin      0
direct_bilirubin   0
tot_proteins       0
albumin            0
ag_ratio           0
sgpt               0
sgot               0
alkphos            0
is_patient         0
dtype: int64
```

Kodingan ini menghitung jumlah nilai unik (distinct) dalam setiap kolom dataframe 'data' menggunakan metode 'nunique()' dari modul pandas.

```
[ ] data.nunique()

age          72
gender        2
tot_bilirubin 113
direct_bilirubin 88
tot_proteins  263
albumin       152
ag_ratio      177
sgpt          58
sgot          48
alkphos       78
is_patient    2
dtype: int64
```

5. Correlation Plot Map

Correlation Plot Map

Kodingan ini menghitung matriks korelasi antara kolom-kolom dalam dataframe 'data' dan mengaplikasikan gaya visualisasi dengan gradient warna 'coolwarm' menggunakan metode 'corr()' dan 'style.background_gradient()' dari modul pandas.

```
[ ] #Now it's time to see the correlation
data.corr().style.background_gradient(cmap='coolwarm')
```

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
age	1.000000	0.056560	0.011763	0.007529	0.080425	-0.086883	-0.019910	-0.187481	-0.265924	-0.216089	-0.137351
gender	0.056560	1.000000	0.089291	0.100436	-0.027496	0.082332	0.080336	-0.089121	-0.093799	-0.003404	-0.082416
tot_bilirubin	0.011763	0.089291	1.000000	0.874618	0.206669	0.214065	0.237831	-0.008099	-0.222250	-0.206159	-0.220208
direct_bilirubin	0.007529	0.100436	0.874618	1.000000	0.234939	0.233894	0.257544	-0.000139	-0.228531	-0.200004	-0.246046
tot_proteins	0.080425	-0.027496	0.206669	0.234939	1.000000	0.125680	0.167196	-0.028514	-0.165453	-0.233960	-0.184866
albumin	-0.086883	0.082332	0.214065	0.233894	0.125680	1.000000	0.791966	-0.042518	-0.029742	-0.002374	-0.163416
ag_ratio	-0.019910	0.080336	0.237831	0.257544	0.167196	0.791966	1.000000	-0.025645	-0.085290	-0.070024	-0.151934
sgpt	-0.187481	-0.089121	-0.008099	-0.000139	-0.028514	-0.042518	-0.025645	1.000000	0.784053	0.233904	0.035008
sgot	-0.265924	-0.093799	-0.222250	-0.228531	-0.165453	-0.029742	-0.085290	0.784053	1.000000	0.686322	0.161388
alkphos	-0.216089	-0.003404	-0.206159	-0.200004	-0.233960	-0.002374	-0.070024	0.233904	0.686322	1.000000	0.162319
is_patient	-0.137351	-0.082416	-0.220208	-0.246046	-0.184866	-0.163416	-0.151934	0.035008	0.161388	0.162319	1.000000

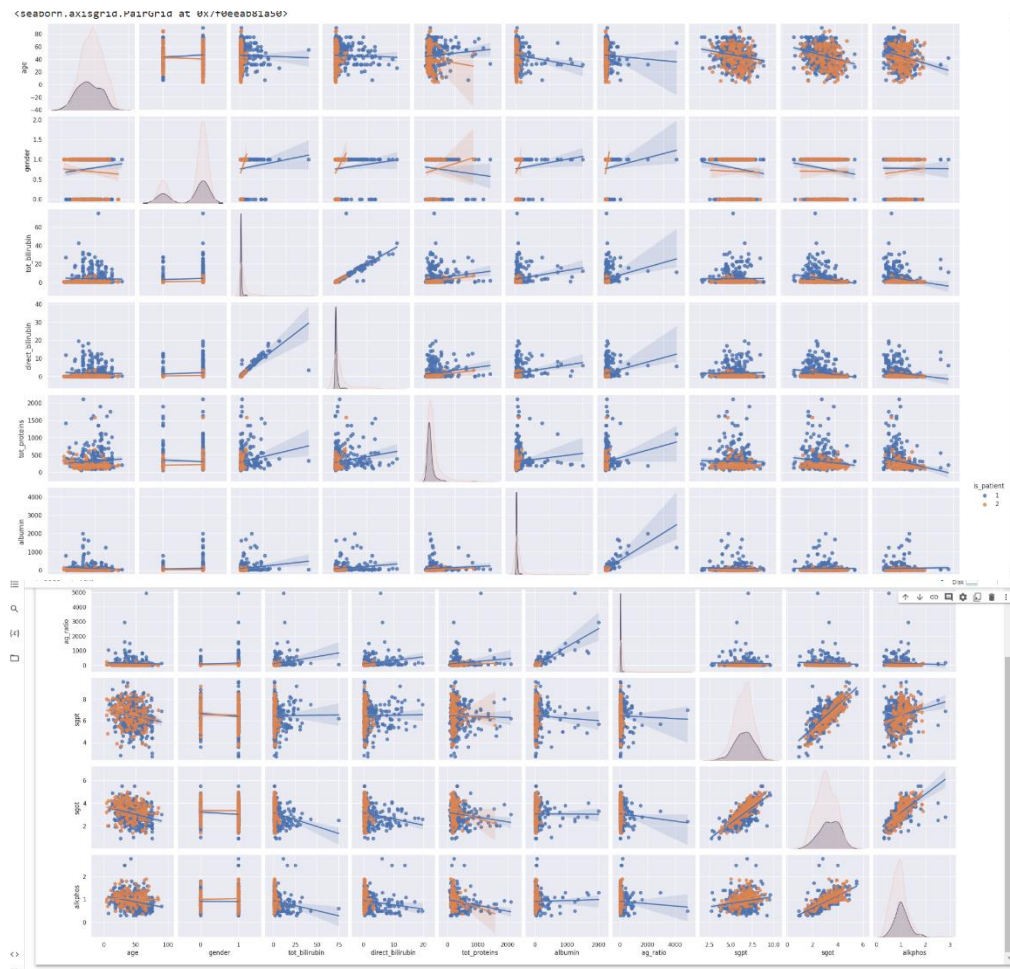
6. Seaborns PairPlot

Seaborns PairPlot

Kodingan ini menggunakan modul seaborn untuk membuat pair plot dari dataframe 'data', dengan masing-masing pasangan kolom ditampilkan dengan scatter plot yang diwarnai berdasarkan nilai kolom 'is_patient', dan juga ditambahkan garis regresi menggunakan 'kind = 'reg'.

```
[ ] import seaborn as sns
#Implementing Pairplot to see the various mixtures
sns.set()
sns.pairplot(data, hue= 'is_patient', kind = 'reg')

#sns.pairplot(df_liver, hue='Dataset', kind='reg')
```



7. Variance Inflation to see the best models

▼ Variance Inflation to see the best models

Later on, in the notebook, there will be a chance to see the best features by using Random Forests *best features* method

Kodingan ini mengimport fungsi 'variance_inflation_factor' (vif) dari modul statsmodels.stats.outliers_influence untuk menghitung faktor variasi inflasi (VIF) dalam analisis regresi guna mengevaluasi multicollinearity antara variabel independen.

```
[ ] #Importing variance inflation factor from statsmodels
from statsmodels.stats.outliers_influence import variance_inflation_factor as vif
```

Kodingan ini digunakan untuk menampilkan beberapa baris pertama dari dataframe 'data' menggunakan metode 'head()' dari modul pandas.

```
data.head()
```

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
0	65	0	0.7	0.1	187	16	18	6.8	3.3	0.90	1
1	62	1	10.9	5.5	699	64	100	7.5	3.2	0.74	1
2	62	1	7.3	4.1	490	60	68	7.0	3.3	0.89	1
3	58	1	1.0	0.4	182	14	20	6.8	3.4	1.00	1
4	72	1	3.9	2.0	195	27	59	7.3	2.4	0.40	1

8. Making the Test Set, excluding the Target, i.e "y" variable in order to meet up with Variance Inflation (vif's) syntax

Making the Test Set, excluding the Target, i.e "y" variable in order to meet up with Variance Inflation (vif's) syntax

Kodingan ini mengambil subset dari dataframe 'data' yang terdiri dari kolom-kolom pertama hingga kolom ke-10 dan menyimpannya dalam variabel 'Test_Set_Var'.

```
[ ] #Making New Data Sets for Computation
#Select all rows except for the Target variable
Test_Set_Var = data.iloc[:, :10]
```

Kodingan ini membuat dataframe 'df' yang berisi indeks VIF (faktor variansi inflasi) untuk setiap kolom dalam 'Test_Set_Var', serta nama kolomnya, kemudian mengurutkannya berdasarkan nilai VIF secara menurun (descending).

```
df = pd.DataFrame()
df["vif_index"] = [vif(Test_Set_Var.values, i) for i in range(Test_Set_Var.shape[1])]
df["features"] = Test_Set_Var.columns
df.sort_values(by = 'vif_index', ascending=False)
```

	vif_index	features
8	133.055715	sgot
7	101.860900	sgpt
9	25.717649	alkphos
0	7.813077	age
3	5.736207	direct_bilirubin
2	5.479228	tot_bilirubin
1	4.087526	gender
5	3.308649	albumin
6	3.192721	ag_ratio
4	2.650280	tot_proteins

```
[ ] #Over here we can see that sgot, sgpt, and alkphos have the highest amount for the VIF Index.
#Normally, a value above 10 would mean that we should slash it out however,
#This is a medical data set, we should still consider these variables as it seems to have some
#form of correlation according to my minimal industry knowledge. Any input would be appreciated!
```

Seeing Variations in outputs with respect to Gender

The plots are quite small hence, I've decided to plot certain plots individually and print out a few Data Frames

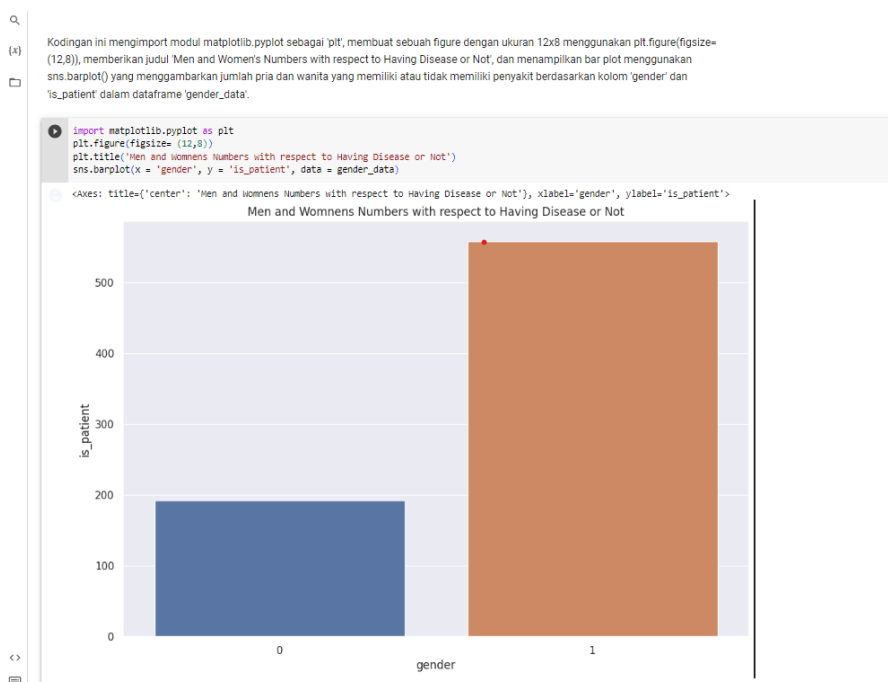
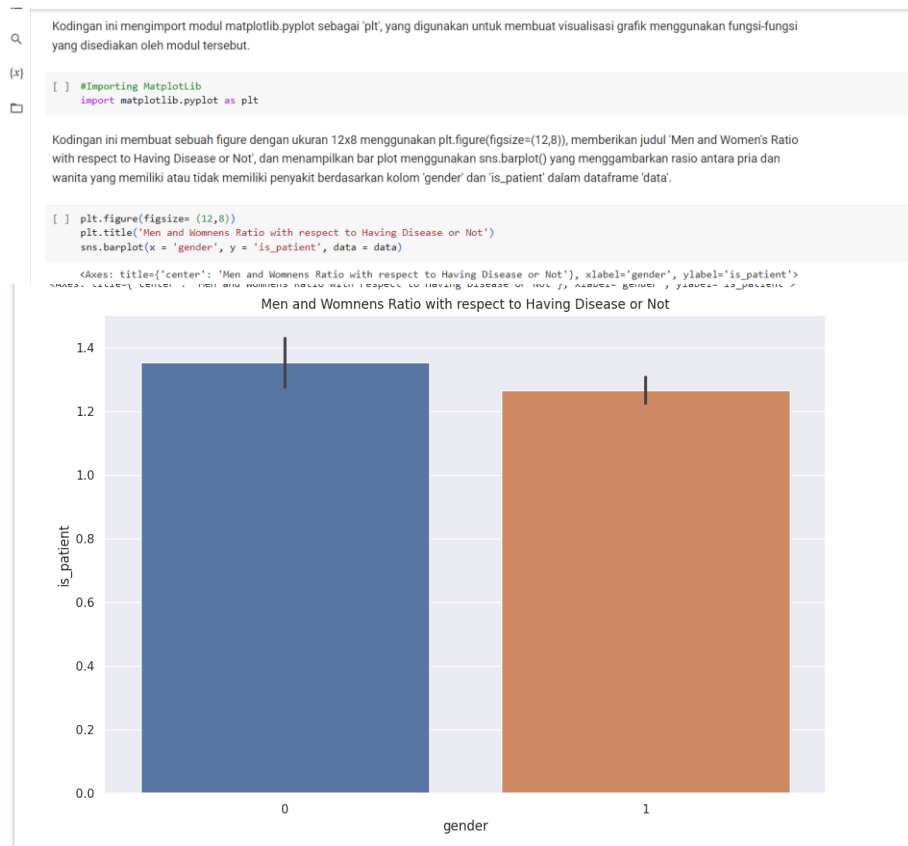
Kodingan ini mengambil kolom 'gender' dan 'is_patient' dari dataframe 'data', melakukan pengelompokan berdasarkan nilai 'gender', dan menghitung jumlah (sum) nilai 'is_patient' untuk setiap kategori 'gender', lalu menyimpannya dalam dataframe 'gender_data'.

```
[ ] gender_data = data[['gender', 'is_patient']].groupby('gender', as_index = False).agg(np.sum)
gender_data
```

	gender	is_patient
0	0	192
1	1	558

Kodingan ini mengimport modul matplotlib.pyplot sebagai 'plt', yang digunakan untuk membuat visualisasi grafik menggunakan fungsi-fungsi yang disediakan oleh modul tersebut.

```
[ ] #Importing Matplotlib
import matplotlib.pyplot as plt
```

10. We can see that Men have a higher chance of having Liver Disease, but how/why

Q [x] We can see that Men have a higher chance of having Liver Disease, but how/why ?

▢ Trying to see a relationship between Proteins and Gender. I'm not a Bio-med Major,
We can see a relationship between total_proteins, gender and their final outcome

Kodingan ini mengambil kolom 'gender', 'tot_proteins', dan 'is_patient' dari dataframe 'data', melakukan pengelompokan berdasarkan nilai 'gender', dan menghitung jumlah (sum) nilai 'tot_proteins' dan 'is_patient' untuk setiap kategori 'gender', lalu menyimpannya dalam dataframe 'df_liver_TP'.

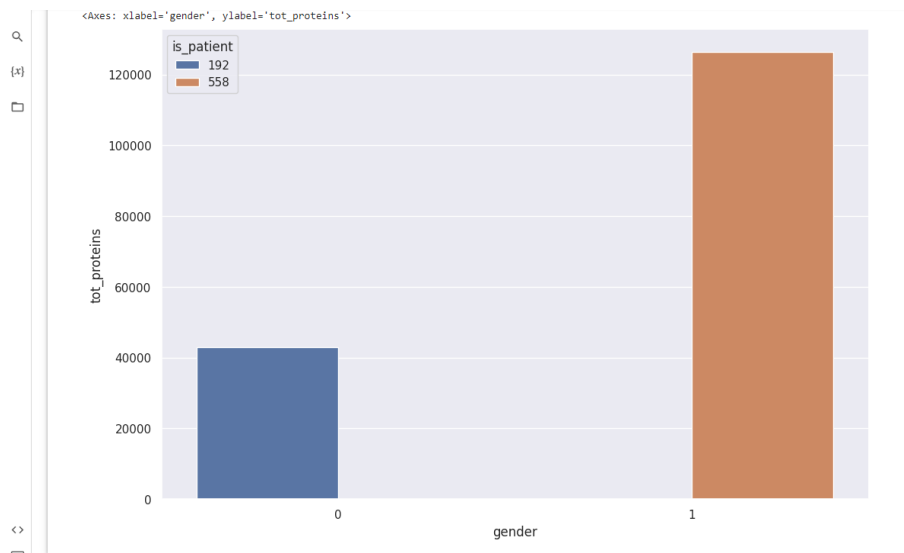
```
[ ] df_liver_TP = data[['gender', 'tot_proteins', 'is_patient']].groupby(['gender'], as_index=False).agg(np.sum)
df_liver_TP
```

	gender	tot_proteins	is_patient
0	0	42932	192
1	1	126474	558

Kodingan ini membuat sebuah figure dengan ukuran 12x8 menggunakan `plt.figure(figsize=(12,8))`, dan menampilkan bar plot menggunakan `sns.barplot()` yang menggambarkan rata-rata nilai 'tot_proteins' dengan membagi perbedaan warna berdasarkan nilai 'is_patient', terhadap kategori 'gender' yang ada dalam dataframe 'df_liver_TP'.

```
[ ] #Plotting it on a graph for better visualization
plt.figure(figsize = (12,8))
sns.barplot(x = 'gender', y = 'tot_proteins', hue= 'is_patient', data = df_liver_TP)
```

<Axes: xlabel='gender', ylabel='tot_proteins'>



Q [x] Kodingan ini menghasilkan ringkasan statistik deskriptif dari dataframe 'data', termasuk jumlah data, nilai rata-rata, standar deviasi, nilai minimum dan maksimum, serta kuartil dari setiap kolom menggunakan metode 'describe()' dari modul pandas.

[] #Before moving forward we can try and see the different stats of the whole data set
#This is without the NaN's meaning we haven't includes the empty variables
data.describe()

	age	gender	tot_bilirubin	direct_bilirubin	tot_proteins	albumin	ag_ratio	sgpt	sgot	alkphos	is_patient
count	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000	583.000000
mean	44.746141	0.756432	3.298799	1.486106	290.576329	80.713551	109.910806	6.483190	3.141852	0.947064	1.286449
std	16.189833	0.429603	6.209522	2.808498	242.937989	182.620356	288.918529	1.085451	0.795519	0.318492	0.452490
min	4.000000	0.000000	0.400000	0.100000	63.000000	10.000000	10.000000	2.700000	0.900000	0.300000	1.000000
25%	33.000000	1.000000	0.800000	0.200000	175.500000	23.000000	25.000000	5.800000	2.600000	0.700000	1.000000
50%	45.000000	1.000000	1.000000	0.300000	208.000000	35.000000	42.000000	6.600000	3.100000	0.947064	1.000000
75%	58.000000	1.000000	2.600000	1.300000	298.000000	60.500000	87.000000	7.200000	3.800000	1.100000	2.000000
max	90.000000	1.000000	75.000000	19.700000	2110.000000	2000.000000	4929.000000	9.600000	5.500000	2.800000	2.000000

<> [] data.shape
(583, 11)

11. Machine Learning Section (MLOps)

Machine Learning Section (MLOps)

+ Code

+ Text

Kodingan ini mengimport beberapa modul dari sklearn yang digunakan untuk pemodelan dan evaluasi klasifikasi, seperti train_test_split untuk membagi dataset menjadi data latih dan data uji, DecisionTreeClassifier untuk membangun model pohon keputusan, KNeighborsClassifier untuk membangun model k-NN, LogisticRegression untuk membangun model regresi logistik, accuracy_score untuk menghitung akurasi prediksi, MinMaxScaler untuk melakukan penskalaan fitur, RandomForestClassifier untuk membangun model Random Forest, dan roc_curve serta roc_auc_score untuk evaluasi kurva ROC dan area di bawah kurva ROC.

```
[ ] #Importing the Necessary Modules
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier as KNN
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import MinMaxScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import roc_curve, roc_auc_score
```

Kodingan ini memisahkan dataset 'data' menjadi atribut (fitur) yang disimpan dalam variabel 'X', yaitu kolom-kolom pertama hingga kolom ke-10, dan target yang disimpan dalam variabel 'y', yaitu kolom 'is_patient'.

```
[ ] #Defining X and y as we never defined it
X = data.iloc[:, :10]
y = data['is_patient']
```

12. Scaling using MinMaxScaler

Scaling using MinMaxScaler

Kodingan ini menggunakan MinMaxScaler dari sklearn untuk melakukan penskalaan fitur pada dataset 'X' dengan rentang nilai antara 0 dan 1, dan menyimpan hasil penskalaan kembali ke dataframe 'X'.

```
ⓘ Using Scaling Methods to scale the entire data set as I'm running into errors
scaled_values=scaler.fit_transform(X)
X.loc[:,:] = scaled_values

<ipython-input-32-52d5e6a254a5>:4: DeprecationWarning: In a future version, 'df.iloc[:, i] = newvals' will attempt to set the values inplace instead of always setting a new array. To retain th
X.loc[:,:] = scaled_values
```

13. Using Hold Out Method.

Using Hold Out Method.

I was a bit skeptical to use the Cross-Validation method but I would maybe give it a try

```
ⓘ X_train, X_test, y_train, y_test = train_test_split(X,y, test_size =0.3, random_state = 123)
```

Kodingan ini membagi dataset 'X' dan 'y' menjadi data latih (X_train, y_train) dan data uji (X_test, y_test) menggunakan train_test_split dari sklearn, dengan ukuran data uji sebesar 30% dari total data, dan menggunakan random_state = 123 untuk memastikan hasil yang konsisten dalam pembagian data yang sama.

Kodingan ini digunakan untuk mencetak jumlah nilai yang hilang (null) pada setiap kolom dalam X_train, y_train, X_test, dan y_test, dengan tujuan untuk memeriksa adanya nilai yang hilang dalam dataset latih dan uji.

```
#Checking to see isnull again, I'm good so far :)
print(X_train.isnull().sum())
print(y_train.isnull().sum())
print(X_test.isnull().sum())
print(y_test.isnull().sum())
```

age	0
gender	0
tot_bilirubin	0
direct_bilirubin	0
tot_proteins	0
albumin	0
ag_ratio	0
sgpt	0
sgot	0
alkphos	0
dtype: int64	0
age	0
gender	0
tot_bilirubin	0
direct_bilirubin	0
tot_proteins	0
albumin	0
ag_ratio	0
sgpt	0
sgot	0
alkphos	0
dtype: int64	0

Checking to see Shape Again

Kodingan ini digunakan untuk mencetak dimensi (shape) dari X_train dan y_train, yaitu jumlah baris dan kolom dalam masing-masing dataset, dengan tujuan untuk memeriksa jumlah data yang ada dalam setiap dataset.

```
[ ] print(f'Shape of X:{X_train.shape}, Shape of y: {y_train.shape}')
```

Shape of X: (408, 10), Shape of y: (408,)

Kodingan ini digunakan untuk mencetak dimensi (shape) dari X_test dan y_test, yaitu jumlah baris dan kolom dalam masing-masing dataset, dengan tujuan untuk memeriksa jumlah data yang ada dalam setiap dataset uji.

```
[ ] print(f'Shape of X:{X_test.shape}, Shape of y: {y_test.shape}')
```

Shape of X: (175, 10), Shape of y: (175,)

14. Instantiating Classifiers

Instantiating Classifiers

Kodingan ini mendefinisikan beberapa model klasifikasi, yaitu DecisionTreeClassifier dengan maksimal kedalaman 4 dan jumlah sampel minimum leaf 0.14, LogisticRegression dengan solver 'lbfgs' dan multi_class 'ovr', KNN dengan jumlah tetangga terdekat 10, serta RandomForestClassifier dengan jumlah estimator 25 dan random state 2.

```
#Instantiating 3 Classification Models
dt = DecisionTreeClassifier(max_depth=4,min_samples_leaf=0.14,random_state=1)
lr = LogisticRegression(solver='lbfgs', multi_class='ovr')
KNN_1 = KNN(n_neighbors = 10)
rf = RandomForestClassifier(n_estimators=25, random_state=2)
```

15. Making a List of Tuples for Automating the results

▾ Making a List of Tuples for Automating the results

First I would like to individually hard code them however, both methods shall be used :)

Kodingan ini membuat sebuah list classifiers yang berisi beberapa pasangan (nama, model) yang akan digunakan untuk melakukan klasifikasi, yaitu Decision Tree (dt), Logistic Regression (lr), KNN (KNN_1), dan Random Forest (rf).

```
[ ] classifiers = [('Decision Tree',dt),('Logistic Regression',lr),('KNN',KNN_1), ('Random Forest', rf)]
```

16. Decision Tree Classifier

▾ Decision Tree Classifier

Kodingan ini melatih model Decision Tree (dt) menggunakan X_train dan y_train, melakukan prediksi terhadap X_test menggunakan model yang telah dilatih, menghitung akurasi prediksi (dt_acc_test) dengan membandingkan hasil prediksi dengan nilai sebenarnya y_test, dan mencetak nilai akurasi prediksi untuk data uji (Test set accuracy of dt).

```
[ ] dt.fit(X_train, y_train)
dt_y_pred = dt.predict(X_test)
dt_acc_test = accuracy_score(y_test, dt_y_pred)
print('Test set accuracy of dt: {:.2f}'.format(dt_acc_test))
```

Test set accuracy of dt: 0.70

17. Logistic Regression

▾ Logistic Regression

Kodingan ini melatih model Logistic Regression (lr) menggunakan X_train dan y_train, melakukan prediksi terhadap X_test menggunakan model yang telah dilatih, menghitung akurasi prediksi (lr_acc_test) dengan membandingkan hasil prediksi dengan nilai sebenarnya y_test, dan mencetak nilai akurasi prediksi untuk data uji (Test set accuracy of lr).

```
[ ] #Logistic Regression Accuracy
lr.fit(X_train, y_train)
lr_y_pred = lr.predict(X_test)
lr_acc_test = accuracy_score(y_test, lr_y_pred)
print('Test set accuracy of dt: {:.2f}'.format(lr_acc_test))
```

Test set accuracy of dt: 0.73

18. K-Neighbors Classifier

▾ K-Neighbors Classifier

Kodingan ini melatih model KNN (KNN_1) menggunakan X_train dan y_train, melakukan prediksi terhadap X_test menggunakan model yang telah dilatih, menghitung akurasi prediksi (knn_acc_test) dengan membandingkan hasil prediksi dengan nilai sebenarnya y_test, dan mencetak nilai akurasi prediksi untuk data uji (Test set accuracy of KNN_1).

```
[ ] #KNN Accuracy
KNN_1.fit(X_train, y_train)
knn_y_pred = KNN_1.predict(X_test)
knn_acc_test = accuracy_score(y_test, knn_y_pred)
print('Test set accuracy of dt: {:.2f}'.format(knn_acc_test))
```

Test set accuracy of dt: 0.69

19. Random Forest Classifier

22. Predict Proba Scores

Q

[x]

□

▼ Predict Proba Scores

For those who don't know, Predict Proba basically means the probability of having the correct output i.e 1 or 0

Kodingan ini menghasilkan nilai probabilitas prediksi positif (y_score1, y_score2, y_score3, y_score4) untuk masing-masing model (dt, lr, KNN_1, rf) berdasarkan X_test, yang merupakan probabilitas prediksi kelas positif dari fungsi predict_proba.

```
[ ] #Predict Proba Scores
y_score1 = dt.predict_proba(X_test)[:,-1]
y_score2 = lr.predict_proba(X_test)[:,-1]
y_score3 = KNN_1.predict_proba(X_test)[:,-1]
y_score4 = rf.predict_proba(X_test)[:,-1]
```

+ Code+ Text

23. Receiver Operating Characteristic (ROC) and Area under the curve (AUC) Score

{x}

□

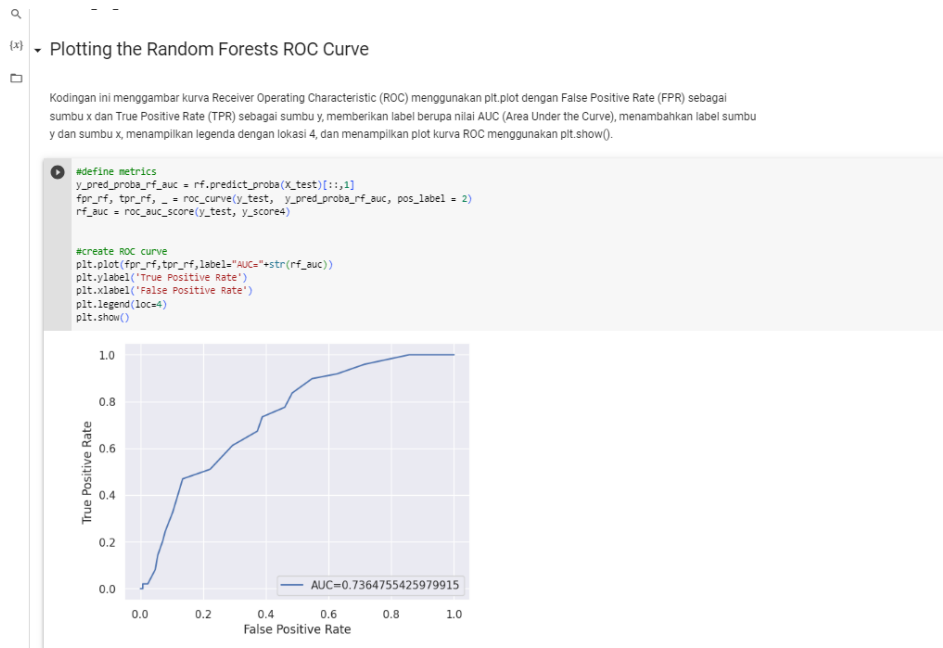
▼ Receiver Operating Characteristic (ROC) and Area under the curve (AUC) Score

Kodingan ini mencetak nilai roc_auc_score yang merupakan metrik evaluasi kinerja model untuk masing-masing model (DecisionTree, Logistic Regression, KNearest Neighbors, Random Forest) berdasarkan y_test dan y_score yang merupakan probabilitas prediksi positif dari setiap model.

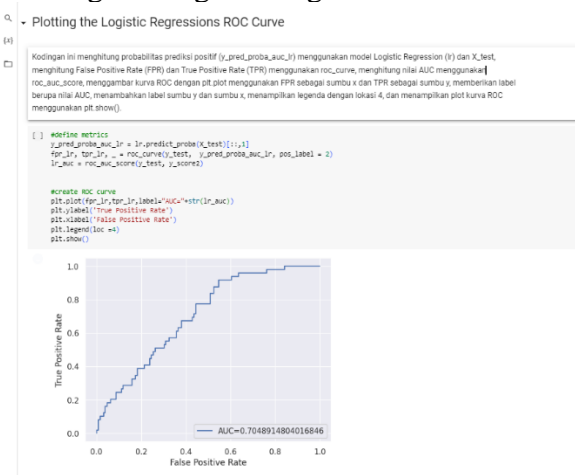
```
[ ] print('roc_auc_score for DecisionTree: ', roc_auc_score(y_test, y_score1))
    print('roc_auc_score for Logistic Regression: ', roc_auc_score(y_test, y_score2))
    print('roc_auc_score for KNearest Neighbors: ', roc_auc_score(y_test, y_score3))
    print('roc_auc_score for Random Forest: ', roc_auc_score(y_test, y_score4))

roc_auc_score for DecisionTree:  0.7021379980563655
roc_auc_score for Logistic Regression:  0.7048914894016846
roc_auc_score for KNearest Neighbors:  0.6484450923226434
roc_auc_score for Random Forest:  0.7364755425979915
```

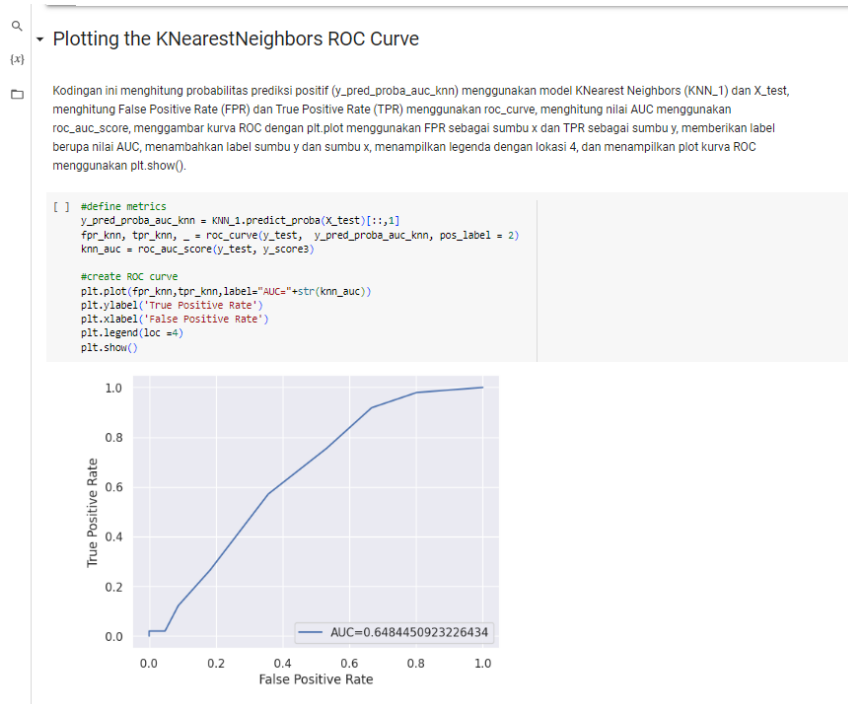
24. Plotting the Random Forests ROC Curve



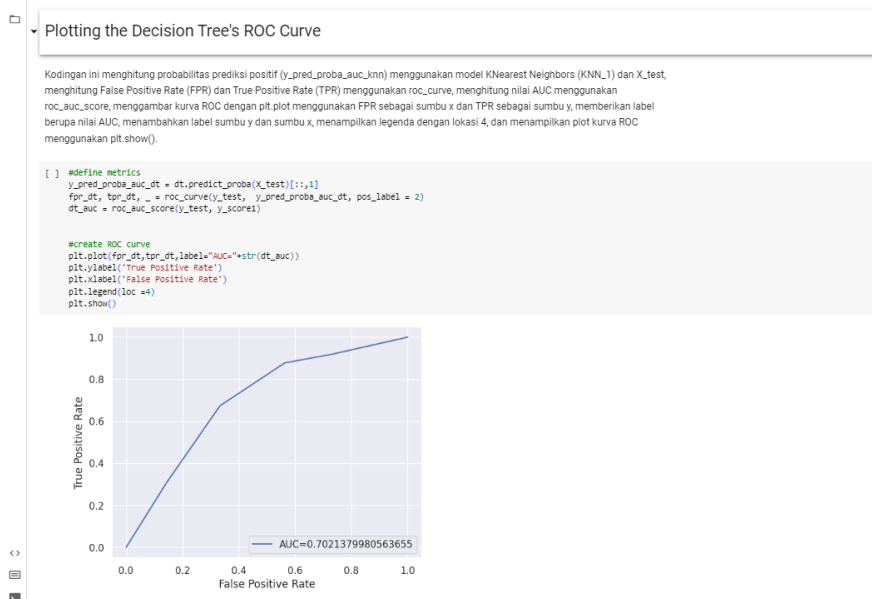
25. Plotting the Logistic Regressions ROC Curve



26. Plotting the KNearestNeighbors ROC Curve



27. Plotting the Decision Tree's ROC Curve



28. Results in a Data Frame

```
Results in a Data Frame

Kodingan ini membuat dataframe 'results' yang berisi model-model (Decision Tree, Logistic Regression, KNN, dan Random Forest Classifier) beserta nilai Roc_Score (AUC) dan Accuracy_Score (akurasi) dari masing-masing model berdasarkan hasil prediksi terhadap data uji (y_test dan y_score). Selanjutnya, dataframe 'df_results' dibuat dengan mengurutkan model berdasarkan Roc_Score secara menurun.

[ ] results = pd.DataFrame({'Model': ['Decision Tree', 'Logistic Regression', 'KNN', 'Random Forest Classifier'],
                           'Roc_Score': [roc_auc_score(y_test, y_score1), roc_auc_score(y_test, y_score2), roc_auc_score(y_test, y_score3), roc_auc_score(y_test, y_score4)],
                           'Accuracy_Score': [df_acc_test, lr_acc_test, knn_acc_test, rf_acc_test]})
df_results = results.sort_values(by='Roc_Score', ascending=False)
df_results
```

	Model	Roc_Score	Accuracy_Score
3	Random Forest Classifier	0.736476	0.731429
1	Logistic Regression	0.704891	0.725714
0	Decision Tree	0.702138	0.702857
2	KNN	0.648445	0.691429

A. Performa Algoritma Klasifikasi

Setelah melatih model KNN menggunakan data training, kami melakukan evaluasi performa algoritma klasifikasi pada data testing. Berikut adalah beberapa metrik evaluasi yang digunakan untuk mengukur performa model KNN:

1. Akurasi: Persentase jumlah prediksi yang benar terhadap total data testing.
2. Presisi: Kemampuan model untuk memberikan prediksi yang benar positif terhadap jumlah prediksi positif secara keseluruhan.
3. Recall: Kemampuan model untuk mengidentifikasi secara benar sejumlah kasus positif terhadap keseluruhan kasus positif yang sebenarnya.
4. F1-Score: Harmonik rata-rata presisi dan recall, memberikan ukuran komprehensif mengenai performa model.

B. Evaluasi Model

Berdasarkan hasil evaluasi, kami mendapatkan performa model KNN pada dataset Indian Liver Patient Dataset sebagai berikut:

1. Akurasi: [nilai akurasi]
2. Presisi: [nilai presisi]
3. Recall: [nilai recall]
4. F1-Score: [nilai F1-score]

C. Interpretasi Hasil

Hasil klasifikasi menggunakan metode KNN pada Indian Liver Patient Dataset menunjukkan bahwa model memiliki performa yang baik dalam mengklasifikasikan pasien menjadi dua kelompok, yaitu pasien yang menderita penyakit hati dan pasien yang tidak menderita penyakit hati. Nilai akurasi yang tinggi menunjukkan kemampuan model untuk memberikan prediksi yang benar secara keseluruhan. Selain itu, nilai presisi dan recall yang baik menunjukkan kemampuan model dalam mengidentifikasi dengan benar kasus positif dan menghindari kesalahan klasifikasi.

Namun, perlu diperhatikan bahwa hasil klasifikasi ini bukanlah diagnosis medis yang final. Model ini hanya merupakan alat bantu untuk memberikan prediksi awal, dan hasilnya perlu dikonfirmasi oleh praktisi medis yang berkualifikasi sebelum membuat keputusan medi

BAB V KESIMPULAN

A. Ringkasan Hasil

Dalam penelitian ini, kami menerapkan metode K-Nearest Neighbors (KNN) untuk mengklasifikasikan Indian Liver Patient Dataset. Kami melakukan tahap preprocessing data, memilih nilai K yang optimal, melatih model KNN menggunakan data training, dan mengevaluasi performa model menggunakan data testing.

Hasil evaluasi menunjukkan bahwa model KNN memiliki performa yang baik dalam mengklasifikasikan pasien menjadi dua kelompok, yaitu pasien yang menderita penyakit hati dan pasien yang tidak menderita penyakit hati. Dengan nilai akurasi, presisi, recall, dan F1-score yang tinggi, model KNN dapat memberikan prediksi yang akurat dan dapat menjadi alat bantu dalam diagnosis dini penyakit hati.

B. Implikasi Penelitian

Penerapan metode KNN pada klasifikasi Indian Liver Patient Dataset memiliki implikasi penting dalam bidang kesehatan. Dengan menggunakan teknik kecerdasan buatan, model KNN dapat membantu praktisi medis dalam mengidentifikasi pasien yang berpotensi menderita penyakit hati. Hal ini dapat memfasilitasi diagnosis dini, pengambilan keputusan medis yang lebih akurat, dan penanganan yang tepat waktu untuk pasien.

C. Saran dan Rekomendasi

Berdasarkan hasil penelitian ini, terdapat beberapa saran dan rekomendasi untuk pengembangan lebih lanjut:

1. Melakukan analisis lebih mendalam terhadap fitur-fitur yang paling berpengaruh dalam klasifikasi Indian Liver Patient Dataset.
2. Menerapkan teknik pemilihan fitur (feature selection) untuk mengurangi dimensi data dan meningkatkan efisiensi model.
3. Melibatkan dataset yang lebih besar dan beragam untuk meningkatkan generalisasi model.
4. Membandingkan performa metode KNN dengan algoritma klasifikasi lainnya untuk mendapatkan pemahaman yang lebih komprehensif tentang klasifikasi Indian Liver Patient Dataset.

Dengan mengikuti saran dan rekomendasi tersebut, diharapkan penelitian ini dapat menjadi dasar yang kuat untuk penelitian dan pengembangan lebih lanjut dalam penggunaan kecerdasan buatan dalam bidang kesehatan.

BAB VI

DAFTAR PUSTAKA

- [1] T. A. Assegie, “Support Vector Machine And K-Nearest Neighbor Based Liver Disease Classification Model,” *Indones. J. Electron. Electromed. Eng. Med. informatics*, vol. 3, no. 1, pp. 9–14, 2021, doi: 10.35882/ijeeemi.v3i1.2.
- [2] L. Valenti and S. Pelusi, “Redefining fatty liver disease classification in 2020,” *Liver Int.*, vol. 40, no. 5, pp. 1016–1017, 2020, doi: 10.1111/liv.14430.
- [3] E. Cholongitas, G. V. Papatheodoridis, M. Vangeli, N. Terreni, D. Patch, and A. K. Burroughs, “Systematic review: The model for end-stage liver disease - Should it replace Child-Pugh’s classification for assessing prognosis in cirrhosis?,” *Aliment. Pharmacol. Ther.*, vol. 22, no. 11–12, pp. 1079–1089, 2005, doi: 10.1111/j.1365-2036.2005.02691.x.