LAPORAN

CASE BASED 1 SUPERVISED LEARNING

Disusun untuk memenuhi tugas

Mata Kuliah: Pembelajaran Mesin

Dosen Pengampu: BEDY PURNAMA, S.SI, MT, Ph.D (BDP)

Oleh:

MUHAMMAD RAFI ANDEO PRAJA (1301200278)

“Saya mengerjakan tugas ini dengan cara yang tidak melanggar aturan perkuliahan dan kode etik akademisi.”

KELAS IF-44-01
JURUSAN S1 INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM

DAFTAR ISI

IKHTISAR

Tabel Dataset

	Age	Sex	Height	Weight	QRS_Dur	P-R_Int	Q-T_Int	T_Int	P_Int	QRS	...	V6270	V6271	V6272	V6273	V6274	V6275	V6276	V6277	V6278	V6279
0	75	0	190	80	91	193	371	174	121	-16	...	-0.3	0.0	9.0	-0.9	0.0	0.0	0.9	2.9	23.3	8
1	56	1	165	64	81	174	401	149	39	25	...	-0.5	0.0	8.5	0.0	0.0	0.0	0.2	2.1	20.4	6
2	54	0	172	95	138	163	386	185	102	96	...	0.9	0.0	9.5	-2.4	0.0	0.0	0.3	3.4	12.3	10
3	55	0	175	94	100	202	380	179	143	28	...	0.1	0.0	12.2	-2.2	0.0	0.0	0.4	2.6	34.6	1
4	75	0	190	80	88	181	360	177	103	-16	...	-0.4	0.0	13.1	-3.6	0.0	0.0	-0.1	3.9	25.4	7

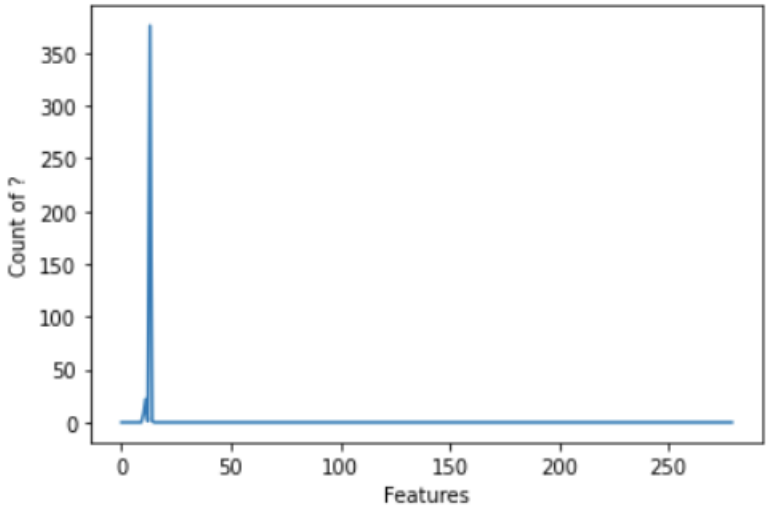
Dataset yang saya ambil berupa pengelompokkan orang atau pasien yang dikategorikan berdasarkan gangguan irama jantung yang terdapat pada pasian atau orang tersebut. Dataset tersebut memiliki beberapa features atau kolom berupa umur, jenis kelamin, tinggi badan, berat badan, dan beberapa ukuran irama jantung berdasarkan Bahasa ilmuwan. Dataset tersebut sebelumnya berupa sebuah data tanpa features dengan beberapa value yang memiliki value outlier (value yang tidak masuk akal) dan memiliki missing value yang dikategorikan sebagai "?". Berikut adalah tampilan dataset sebelum dikelompokkan berdasarkan features yang tersedia:

Tabel Dataset Tanpa Features

	0	1	2	3	4	5	6	7	8	9	...	270	271	272	273	274	275	276	277	278	279
0	75	0	190	80	91	193	371	174	121	-16	...	0.0	9.0	-0.9	0.0	0.0	0.9	2.9	23.3	49.4	8
1	56	1	165	64	81	174	401	149	39	25	...	0.0	8.5	0.0	0.0	0.0	0.2	2.1	20.4	38.8	6
2	54	0	172	95	138	163	386	185	102	96	...	0.0	9.5	-2.4	0.0	0.0	0.3	3.4	12.3	49.0	10
3	55	0	175	94	100	202	380	179	143	28	...	0.0	12.2	-2.2	0.0	0.0	0.4	2.6	34.6	61.6	1
4	75	0	190	80	88	181	360	177	103	-16	...	0.0	13.1	-3.6	0.0	0.0	-0.1	3.9	25.4	62.8	7

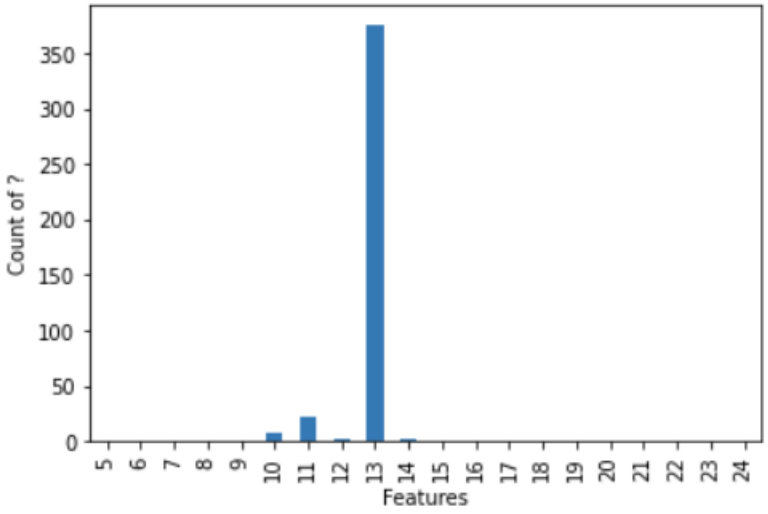
Terdapat 280 jumlah features dan 452 jumlah baris yang saya gunakan semua dengan perincian bahwa features dengan kolom bernama "V6279" atau kolom dengan nomor 279 yang saya gunakan sebagai data Y untuk digunakan dalam penggunaan algoritma. Maka dari itu untuk data X, saya hanya menggunakan data dengan jumlah features 277 atau sampai dengan kolom bernama "V6277" dan saya drop kolom atau drop features dengan nomor 278 karena data tersebut tidak dibutuhkan untuk dimasukkan sebagai data X yang nantinya akan digunakan dalam penggunaan algoritma.

Grafik Jumlah Missing Values



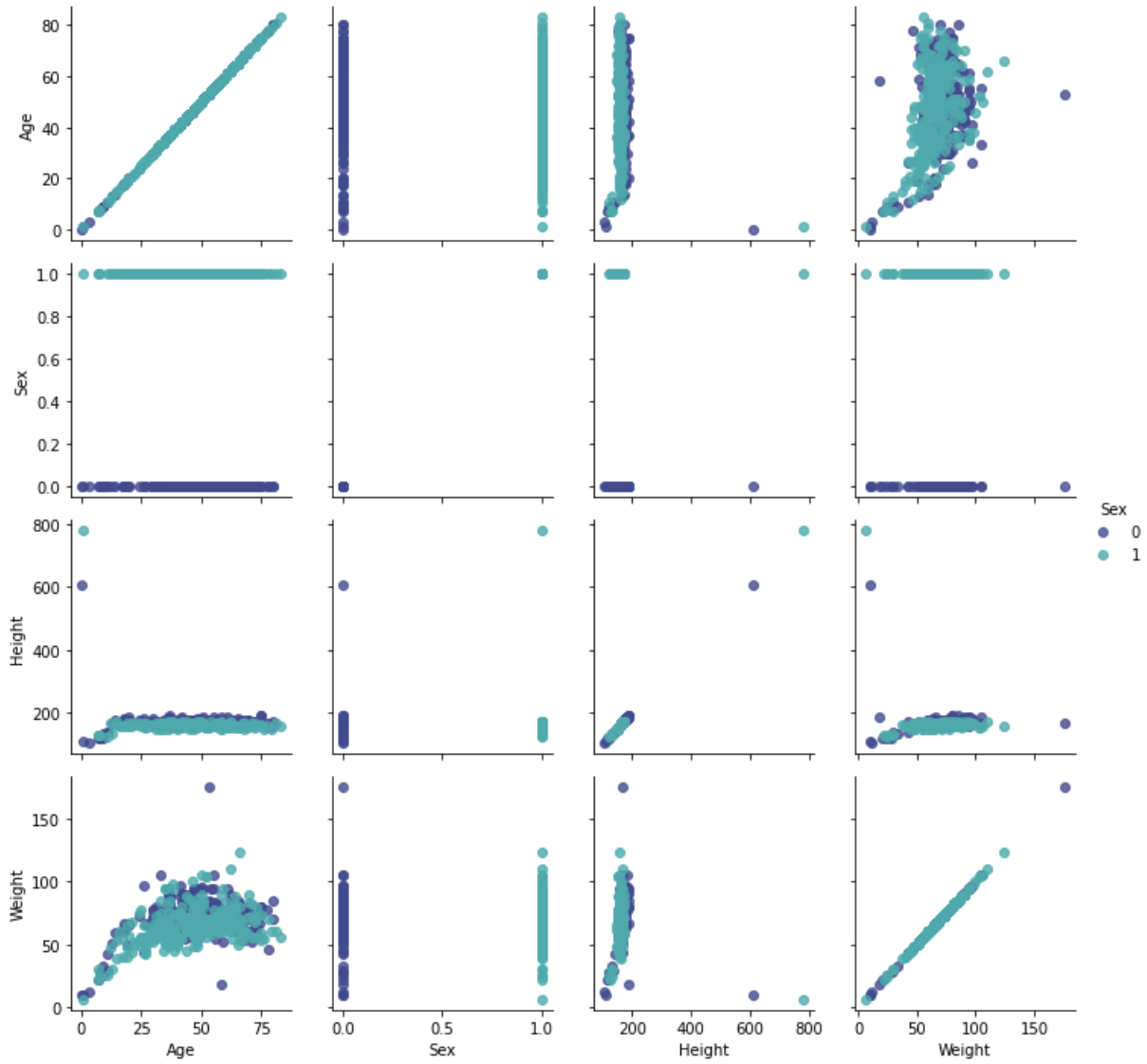
Terdapat 408 jumlah missing value atau value dengan kategori “?” yang terdapat di dalam dataset dengan persebaran hanya 5 kolom atau 5 features saja yang memiliki missing value tersebut.

Grafik Persebaran Missing Values



Persebaran missing value berikut dapat disimpulkan bahwa terdapat banyaknya jumlah missing value terbesar terdapat pada features 13 dan disusul dengan features 11 sebagai features dengan missing value terbanyak kedua.

Grafik Value Features



Terdapat outlier value yang dapat dilihat pada grafik height dengan adanya 2 nilai diatas value 200, 2 value tersebut dapat dikatakan sebagai outlier value yang dimiliki oleh dataset tersebut.

PRA-PEMROSESAN DATA

Sebelum masuk ke dalam tahapan pra – pemrosesan data, dataset tersebut harus diubah menjadi sebuah dataset yang layak dan dapat untuk di proses di dalam algoritma yang dipilih. Kekurangan dari dataset tersebut terdapat pada value dari features yang tersedia, terdapat 2 permasalahan value yang terdapat pada dataset tersebut. Dataset tersebut memiliki missing value yang harus dihilangkan dengan melakukan dropping feature pada feature yang memiliki missing value terbanyak. Setelah dihilangkan, jika terdapat beberapa missing value lainnya yang tersebar di beberapa feature, perlu diisikan value tersebut dengan menambahkan value modus, mean, atau median.

Program Menghilangkan Missing Values

```
[88] df.drop(columns = 13, inplace=True)
```

Mengubah missing value menjadi nilai modus

```
[89] df[10].replace([np.nan], df[10].mode()[0], inplace=True)
     df[11].replace([np.nan], df[11].mode()[0], inplace=True)
     df[12].replace([np.nan], df[12].mode()[0], inplace=True)
     df[14].replace([np.nan], df[14].mode()[0], inplace=True)
```

```
[90] nanValue = 0
     for i in range(0, 452):
       for j in range(0, 279):
         if (df.iloc[i, j] == np.nan):
           nanValue += 1
     print("Missing value yang terdapat pada dataset tersebut adalah:", nanValue)
```

```
Missing value yang terdapat pada dataset tersebut adalah: 0
```

Karena pada feature 13 terdapat missing value dengan jumlah terbesar, maka feature 13 dapat dihilangkan dengan melakukan dropping feature. Setelah itu terdapat beberapa missing value yang tersebar pada feature 10, 11, 12, dan 14 yang tidak akan dilakukan dropping feature, karena beberapa value feature tersebut dapat digunakan dalam pemrosesan data ke dalam algoritma. Features 10, 11, 12, dan 14 untuk missing value saya ubah valuenya menjadi value modus dari featurenya sendiri, dengan hasil bahwa missing value dataset tersebut sudah berjumlah 0.

Program Menghilangkan Outlier Values

Mengubah outlier value menjadi value yang normal

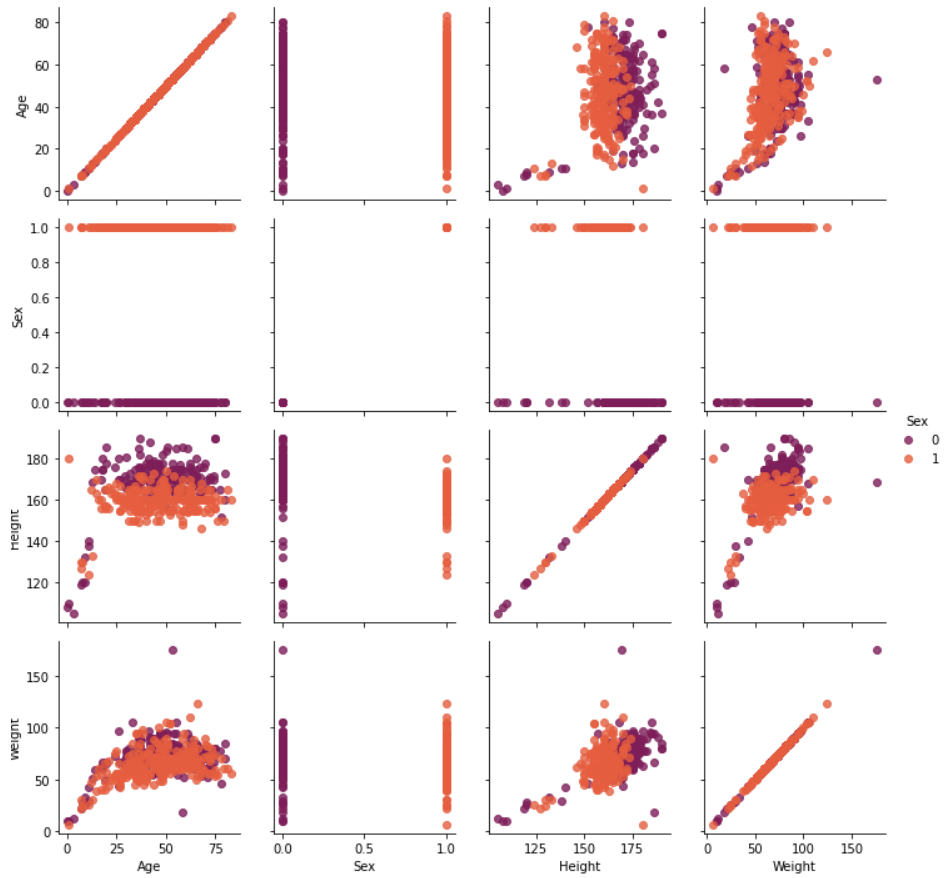
```
[95] sorted(x['Height'], reverse=True)[:10]

     [780, 608, 190, 190, 190, 188, 186, 186, 186, 185]
```

```
[96] x['Height'] = x['Height'].replace(608,108)
     x['Height'] = x['Height'].replace(780,180)
```

Setelah melakukan pengubahan missing value, selanjutnya adalah melakukan pengubahan outlier value yang dimana outlier value adalah suatu value dari feature yang memiliki nilai tidak wajar. Dari grafik value features dapat diketahui bahwa terdapat 1 feature bernama "Height" yang memiliki 2 value bernilai tidak wajar. Outlier values tersebut saya ubah dengan mengganti value tersebut menjadi normal value. Terdapat 1 feature lagi bernama "Weight" yang memiliki 1 outlier value, tetapi outlier value tersebut dapat dikategorikan menjadi normal value dari segi fakta.

Grafik Value Features Tanpa Outlier Value



Alat analisis data yang saya gunakan adalah Python yang dimana sebagai salah satu alat analisis data terlengkap yang memiliki library yang dibutuhkan dan dapat digunakan di tugas berikut. Beberapa library seperti matplotlib dan seaborn memiliki fungsi dalam menampilkan isi dataset dan menampilkan beberapa features dataset dalam bentuk plot atau grafik. Algoritma pembelajaran mesin juga dapat digunakan sebagai library dalam python yang dapat membantu saya dari pemrosesan data sampai perhitungan prediksi menggunakan algoritma pembelajaran mesin.

Program Scalling Data

Membuat variabel x train, x test, y train, dan y test yang akan digunakan di algoritma neural network

```
[ ]  x_train, x_test, y_train, y_test = ms.train_test_split(x, y, test_size=0.20, random_state = 10)
     print("Jumlah data training : ", len(x_train))
     print("Jumlah data testing  : ", len(x_test))
```

```
Jumlah data training :  361
Jumlah data testing  :  91
```

Melakukan proses scalling terhadap variabel x train dan x test

```
[ ]  scaler = MinMaxScaler()
     x_train = scaler.fit_transform(x_train)
     x_test = scaler.transform(x_test)
```

Selanjutnya, teknik pra – pemrosesan data yang saya gunakan adalah melakukan reduksi berdasarkan transformasi, dengan pengubahan nilai value berdasarkan range tertentu. Teknik tersebut dinamakan proses scalling yang dimana saya menggunakan proses tersebut untuk mengubah bentuk data x train dan x test supaya dapat digunakan di dalam fungsi algoritma nanti. Sebelumnya saya juga mencoba melakukan teknik pra – pemrosesan data yang dimana melakukan reduksi dimensi, tetapi data tersebut tidak bisa digunakan setelah reduksi dimensi karena memiliki jumlah value yang bertambah dengan hasil jumlah value data x train dan x test tidak sama dengan data y train dan y test. Teknik pra – pemrosesan data reduksi berdasarkan transformasi yaitu scalling, lebih layak digunakan karena hanya mengubah value dari data tersebut yang nantinya dapat digunakan di algoritma pembelajaran mesin nanti, dengan scalling yang saya gunakan adalah scalling berdasarkan Min – Max Normalization dengan rumus berikut:

Rumus Min – Max Normalization

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

ALGORITMA SUPERVISED LEARNING

Program Neural Network MLP (Logistic Sigmoid)

Menggunakan perhitungan prediksi data menggunakan algoritma neural network MLP

Fungsi Logistic Sigmoid

$$S(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} = 1 - S(-x).$$

```
[ ]  mlp = ann.MLPClassifier(activation='logistic', random_state=10, hidden_layer_sizes=(250,), max_iter=2500)
     mlp.fit(x_train, np.ravel(y_train))
```

```
MLPClassifier(activation='logistic', hidden_layer_sizes=(250,), max_iter=2500,
              random_state=10)
```

```
[ ]  prediksi = mlp.predict(x_test)
     print(prediksi)
```

```
[ 3  1  2  1  1  1  2  1  4  2  1  1  1 10 10  1  1  3  1 10  6  4  1  1
  1  1  1 16  1  1  1  1  1  1  1 10  1  6  2  1  6 10  2  1  1  6  1  1
  4  3  1  1  1  1  1  1  1 10  1  1  1  1  1  1 16  1 10  1 10  2 10 10
  2 10  1  1  1  1  1  1  2  1  1  1  1  1  2  9  1  1  1]
```

Saya menggunakan algoritma MLP sebagai algoritma yang akan menghitung hasil prediksi dari dataset test yang diberikan. Menurut saya, algoritma MLP atau algoritma Multi Layer Perceptron adalah algoritma supervised learning yang menggunakan hidden layer atau perhitungan prediksi yang tersembunyi. Hidden layer tersebut akan menerima inputan user berupa dataset yang diolah lalu akan dikembangkan berdasarkan cara kerja jaringan saraf dan hidden layer tersebut akan mengoutputkan hasil dari cara kerja jaringan saraf menjadi output prediksi dataset. Penggunaan algoritma MLP dapat menghasilkan nilai akurasi yang tinggi atau output dataset yang cukup sama dengan actual output pada dataset.

Saya menggunakan 3 fungsi utama MLP dalam pencarian fungsi apa yang akan menghasilkan nilai akurasi tertinggi. Fungsi yang pertama adalah Fungsi Logistic Sigmoid yang dimana menggunakan bilangan euler dalam pencarian output prediksi. Hasil nilai akurasi yang saya dapatkan dengan parameter hidden layer adalah 250 dan max iteration adalah 2500 dengan tambahan random state adalah 10 mendapat nilai akurasi sebesar 78.02%.

Penggunaan parameter untuk setiap fungsi akan disamakan dan hasil yang akan dijadikan sebagai pertimbangan penggunaan fungsi akan berfokus terhadap output nilai akurasinya saja.

Program Neural Network MLP (Rectified Linear Unit)

Fungsi Rectified Linear Unit (relu)

$$f(x) = x^+ = \max(0, x),$$

```
[ ]  mlp = ann.MLPClassifier(activation='relu', random_state=10, hidden_layer_sizes=(250,), max_iter=2500)
     mlp.fit(x_train, np.ravel(y_train))
```

```
MLPClassifier(hidden_layer_sizes=(250,), max_iter=2500, random_state=10)
```

```
[ ]  prediksi = mlp.predict(x_test)
     print(prediksi)
```

```
[ 3  1  2  1  1  1  2  1  4  2  1  1  1 10 10  1  1  3  1 10  6  4  1  1
  1  1  1 16  1  1  1  1  1  1  1 10  1  6  2  1  6 16  2  1  1  6  1  1
  4  3  1  1  1  1  1  1  1 10  1  1  1  1  1  1 16  1 10  1 10  2 10 10
  2 10  1  1  1  1  1  1  2  1  1  1  1  1  2  9  1  1  1]
```

Fungsi kedua yang saya gunakan adalah fungsi Rectified Linear Unit yang menggunakan nilai maksimum dalam perhitungan prediksi output dataset. Dengan parameter yang sama, hasil nilai akurasi yang saya dapatkan adalah 79.12%. Untuk sementara fungsi Rectified Linear Unit memiliki nilai akurasi yang lebih baik dibandingkan dengan fungsi Logistic Sigmoid dengan penggunaan nilai parameter yang sama.

Program Neural Network MLP (Tangen Hyperbolik)

Fungsi Tangen Hyperbolik (tanh)

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}.$$

```
[ ]  mlp = ann.MLPClassifier(activation='tanh', random_state=10, hidden_layer_sizes=(250,), max_iter=2500)
     mlp.fit(x_train, np.ravel(y_train))
```

```
MLPClassifier(activation='tanh', hidden_layer_sizes=(250,), max_iter=2500,
              random_state=10)
```

```
prediksi = mlp.predict(x_test)
print(prediksi)
```

```
[ 3  1  2  1  1  1  2  1  4  2  1  1  1 10 10  1  1  3  1 10  6  4  1  1
  1  4  1 16  1  1  1  1  1  1  1 10  1  6  2  1  6 16  2  1  1  6  1  1
  4  3  1  1  1  1  1  4  1 10  1  1  1  1  1  1 16  1 10  1 10  2 10 10
  2 10  1  1  1  1  1  1  2  1  1  1  1  1  2  9  1  1  1]
```

Fungsi terakhir yang saya gunakan adalah fungsi Tangen Hyperbolik yang menggunakan nilai tan dalam perhitungan prediksi output dataset. Dengan parameter yang sama antara 2 fungsi, hasil nilai akurasi yang saya dapatkan adalah 81.32%. Fungsi tangen hyperbolik memiliki nilai akurasi yang paling tinggi dan memiliki nilai akurasi yang cukup bagus karena rentang nilai akurasi tersebut diatas 80%.

Program Neural Network MLP dengan Parameter yang Berbeda

Menggunakan parameter yang berbeda

Fungsi Tangen Hyperbolik (tanh)

```
[32] mlp = ann.MLPClassifier(activation='tanh', random_state=10, hidden_layer_sizes=(500,), max_iter=5000)
     mlp.fit(x_train, np.ravel(y_train))
```

```
MLPClassifier(activation='tanh', hidden_layer_sizes=(500,), max_iter=5000,
              random_state=10)
```

```
[33] prediksi = mlp.predict(x_test)
     print(prediksi)
```

```
[ 3  1  2  1  1  1  2  1  4  2  1  1  1 10 10  1  1  3  1 10  6  4  1  1
  1  4  1 16  1  1  1  1  1  1  1 10  1  6  2  1  6 16  2  1  1  6  1  1
  4  3  1  1  1  1  1  4  1 10  1  1  1  1  1  1 16  1 10  1 10  2 10 10
  2 10  1  1  1  1  1  1  2  1  1  1  1  1  2  9  1  1  1]
```

Tetapi, bagaimana jika penggunaan setiap fungsi kembali diulang tetapi menggunakan parameter yang berbeda apakah akan menghasilkan nilai akurasi yang sama? Saya akan mencoba melakukan perhitungan output nilai akurasi dengan parameter yang berbeda berdasarkan rincian berikut:

Tabel Parameter

Random State	Hidden Layer	Max Iter
10	50	500
10	100	1000
10	250	2500
10	500	5000
10	750	7500

Parameter semula yang digunakan dalam setiap fungsi adalah:

Random State = 10

Hidden Layer = 250

Max Iter = 2500

Terdapat beberapa perbedaan nilai akurasi berdasarkan parameter dan penggunaan fungsi tersebut yang dimana nilai akurasi dapat meningkat atau nilai akurasi juga bisa dapat turun yang nanti akan dibahas pada bab selanjutnya.

EVALUASI

Tabel Akurasi

Fungsi	Random State	Hidden Layer	Max Iter	Akurasi
Logistic Sigmoid	10	250	2500	78.02%
Rectified Linear Unit	10	250	2500	79.12%
Tangen Hyperbolik	10	250	2500	81.32%
Logistic Sigmoid	10	500	5000	79.12%
Rectified Linear Unit	10	500	5000	79.12%
Tangen Hyperbolik	10	500	5000	81.32%
Logistic Sigmoid	10	100	1000	80.22%
Rectified Linear Unit	10	100	1000	78.02%
Tangen Hyperbolik	10	100	1000	81.32%
Logistic Sigmoid	10	50	500	75.82%
Rectified Linear Unit	10	50	500	75.82%
Tangen Hyperbolik	10	50	500	76.92%
Logistic Sigmoid	10	750	7500	79.12%
Rectified Linear Unit	10	750	7500	79.12%
Tangen Hyperbolik	10	750	7500	80.22%

Dengan menggunakan parameter yang berbeda, semua fungsi memiliki tingkatan nilai akurasi yang dapat berubah dan dapat juga tidak berubah. Tetapi dari semua fungsi, fungsi Tangen Hyperbolik memiliki nilai tingkatan akurasi tertinggi dengan beberapa test case parameter yang dibuat.

Program Menampilkan Performansi Algoritma MLP

Menampilkan nilai akurasi model

```
[30] count = 0
     for i, output in enumerate(np.ravel(y_test)):
       if output == prediksi[i]:
         count += 1
     print("Akurasi model: {}%".format(round(count/len(y_test)*100, 2)))
```

```
Akurasi model: 81.32%
```

```
import sklearn.metrics as met
print(met.classification_report(y_test, prediksi))
```

```
              precision    recall  f1-score   support

           1       0.89      0.96      0.93        52
           2       0.33      0.43      0.38         7
           3       1.00      0.75      0.86         4
           4       0.80      1.00      0.89         4
           5       0.00      0.00      0.00         1
           6       0.50      0.50      0.50         4
           9       1.00      0.33      0.50         3
          10       0.90      1.00      0.95         9
          14       0.00      0.00      0.00         2
          15       0.00      0.00      0.00         1
          16       0.67      0.50      0.57         4

    accuracy                           0.81        91
   macro avg       0.55      0.50      0.51        91
weighted avg       0.79      0.81      0.79        91
```

Saya menampilkan beberapa performansi lain selain akurasi sebagai nilai acuan untuk menghitung dan menyimpulkan fungsi dengan parameter apa yang terbaik dalam mengolah dataset tersebut. Dapat saya simpulkan bahwa fungsi Tangen Hyperbolik dengan parameter berikut menjadi fungsi dengan parameter yang memiliki performansi terbaik dalam pengolahan dataset tersebut:

Fungsi = Tangen Hyperbolik

Random State = 10

Hidden Layer = 250

Max Iter = 2500

Output Berdasarkan Distribusi Kelas yang Sebenarnya

Class Distribution: Database: Arrhythmia

Class code :	Class :	Number of instances:
01	Normal	245
02	Ischemic changes (Coronary Artery Disease)	44
03	Old Anterior Myocardial Infarction	15
04	Old Inferior Myocardial Infarction	15
05	Sinus tachycardy	13
06	Sinus bradycardy	25
07	Ventricular Premature Contraction (PVC)	3
08	Supraventricular Premature Contraction	2
09	Left bundle branch block	9
10	Right bundle branch block	50
11	1. degree AtrioVentricular block	0
12	2. degree AV block	0
13	3. degree AV block	0
14	Left ventricule hypertrophy	4
15	Atrial Fibrillation or Flutter	5
16	Others	22

Kesimpulan yang dapat saya berikan dari penggunaan algoritma MLP berikut dengan menggunakan 3 fungsi yang berbeda adalah terdapat perbedaan tingkat akurasi berdasarkan penggunaan 3 fungsi tersebut yang dimana fungsi tangen hyperbolic dapat menghasilkan akurasi yang besar jika memperbesar jumlah test size dan jumlah random state dan memperkecil jumlah max iter dan jumlah hidden layer size dengan ukuran yang secukupnya, parameter tersebut menjadi kunci utama pengolahan dataset dalam algoritma MLP. Hasil akurasi terbukti bahwa fungsi tangen hyperbolic memiliki akurasi tertinggi, dan semua akurasi memiliki tingkatan diatas 75% yang dimana dapat disimpulkan sebagai algoritma yang cukup mampu dalam mengolah dataset tersebut.

Penugasan Case Based 1 berikut menjadi sebuah salah satu kesempatan bagi saya dalam mempelajari salah satu algoritma supervised learning dan juga kesempatan dalam mempelajari bagaimana cara melakukan teknik pra – pemrosesan data terhadap dataset yang masih raw atau tidak layak untuk digunakan. Maka dari itu, saya atas nama Muhammad Rafi Andeo Praja mengucapkan banyak rasa syukur dan rasa terima kasih kepada Bapak Bedy Purnama selaku dosen pengampu mata kuliah Pembelajaran Mesin dan seluruh Tim Dosen Pembelajaran Mesin S1 Informatika tahun ajaran 2022/2023. Saya sangat terbuka dalam menerima segala saran dan kritikan terhadap penugasan Case Based 1 ini, dan saya tau bahwa penugasan Case Based 1 ini

dari segi laporan dan kode program dapat dikatakan masih sangat jauh dari kata sempurna. Saya berharap bahwa untuk penugasan berikutnya dalam mata kuliah yang sama atau mata kuliah yang berbeda, saya dapat melatih diri saya kembali dalam pembuatan laporan yang baik dan pembuatan kode program yang baik.

Link Kode Program:

Github: https://github.com/RafiAndeo/Multi_Layer_Perceptron_Algorithm

Google Colab: https://colab.research.google.com/drive/1Y-MIg80swqR6RueTYcQPgAXMYHkAfjJJ?usp=sharing

Link Video Presentasi:

https://drive.google.com/file/d/1TB0lTENBVxGh4Ctij2Mi4FHHp-n901_s/view?usp=share_link

Link Slide Presentasi:

https://docs.google.com/presentation/d/1_BwqWbyjdbNHzAhcbeS2g9nLMmgaWvUG/edit?usp=share_link&ouid=116705627032248410275&rtpof=true&sd=true

DAFTAR PUSTAKA

Kurniawan, Dios. 2020. *Pengenalan Machine Learning dengan Python*. Jakarta: PT Elex Media Komputindo