LAPORAN CASE-BASED 1
PEMBELAJARAN MESIN

Disusun oleh

Nama : Anyelir Belia Azzahra
NIM : 1301200048
Kelas : IF-44-08 [DDR]

PROGRAM STUDI S1 INFORMATIKA
FAKULTAS INFORMATIKA
UNIVERSITAS TELKOM
2022/2023

DAFTAR ISI

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Laporan ini dibuat untuk memenuhi Tugas Case-Based 1 Pembelajaran Mesin untuk menganalisis data. Data set yang saya pakai sesuai aturan NIM Genap yaitu Data set Arrhythmia. Algoritma supervised yang saya pilih adalah algoritma ANN (Artificial Neural Network) . ANN merupakan model penalaran yang didasarkan pada otak manusia. ANN terdiri dari sejumlah prosesor sangat sederhana dan saling berhubungan yang disebut neuron. Neuron yang terhubung dengan pembobotan (weight) melewati sinyal dari neuron satu ke neuron yang lain.

1.2 Rumusan Masalah

Rumusan masalah pada laporan kali ini adalah sebagai berikut.

a. Apa algoritma supervised yang digunakan?
b. Bagaimana ikhtisar kumpulan data yang dipilih?
c. Bagaimana ringkasan pra pemrosesan data yang diusulkan?
d. Bagaimana implementasi dari algoritma supervised yang dipilih?
e. Bagaimana hasil yang diperoleh menggunakan algoritma supervised yang dipilih?

1.3 Tujuan

Tujuan dari dibuatnya laporan ini adalah sebagai berikut.

a. Menjelaskan algoritma supervised yang dipilih (ANN/MLP/RNN/LSTM/CNN)
b. Menjelaskan ikhtisar kumpulan data yang digunakan
c. Menjelaskan ringkasan pra pemrosesan data yang diusulkan
d. Menjelaskan implementasi dari algoritma supervised yang dipilih (ANN/MLP/RNN/LSTM/CNN)
e. Menjelaskan hasil yang diperoleh menggunakan algoritma supervised yang dipilih (ANN/MLP/RNN/LSTM/CNN)

BAB 2
PEMBAHASAN

2.1 Ikhtisar Data

Berikut ini merupakan ikhtisar data yang dipilih berdasarkan data dari aturan NIM Genap yaitu data set Arrhythmia dengan menggunakan algoritma ANN (Artificial Neural Network). Pada ikhtisar data ini terdapat tabel dan plot yang terstruktur mengenai data Arrhythmia.

2.1.1 Data yang dipilih

Berikut ini merupakan code untuk menampilkan data yang dipilih dari dataset Arrhythmia yang terdiri dari 450 rows dan 280 columns.

Menampilkan Data yang dipilih

```
#menampilkan data

data = pd.read_csv('arrhythmia_csv.csv')
data
```

	age	sex	height	weight	QRSduration	PRinterval	Q-Tinterval	Tinterval	Pinterval	QRS	...	chV6_QwaveAmp	chV6_RwaveAmp	chV6_SwaveAmp	chV6_RPwaveAmp	chV6_SPwaveAmp	chV6_Pwave
0	75	0	190	80	91	193	371	174	121	-16	...	0.0	9.0	-0.9	0.0	0.0	
1	56	1	165	64	81	174	401	149	39	25	...	0.0	8.5	0.0	0.0	0.0	
2	54	0	172	95	138	163	386	185	102	96	...	0.0	9.5	-2.4	0.0	0.0	
3	55	0	175	94	100	202	380	179	143	28	...	0.0	12.2	-2.2	0.0	0.0	
4	75	0	190	80	88	181	360	177	103	-16	...	0.0	13.1	-3.6	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
447	53	1	160	70	80	199	382	154	117	-37	...	0.0	4.3	-5.0	0.0	0.0	
448	37	0	190	85	100	137	361	201	73	86	...	0.0	15.6	-1.6	0.0	0.0	
449	36	0	166	68	108	176	365	194	116	-85	...	0.0	16.3	-28.6	0.0	0.0	
450	32	1	155	55	93	106	386	218	63	54	...	-0.4	12.0	-0.7	0.0	0.0	
451	78	1	160	70	79	127	364	138	78	28	...	0.0	10.4	-1.8	0.0	0.0	

452 rows × 280 columns

Menampilkan deskripsi data Arrhythmia Data Set. Deskripsi ini berisi count, min, max, mean, std, dll dari setiap kategori

```
data.describe()
```

	age	sex	height	weight	QRSduration	PRinterval	Q-Tinterval	Tinterval	Pinterval	QRS	...	chV6_QwaveAmp	chV6_RwaveAmp	chV6_SwaveAmp	chV6_RPwaveAmp	chV6_SPwaveAmp	c
count	452.000000	452.000000	452.000000	452.000000	452.000000	452.000000	452.000000	452.000000	452.000000	452.000000	...	452.000000	452.000000	452.000000	452.000000	452.0	
mean	46.471239	0.550885	166.188053	68.170354	88.920354	155.152655	367.207965	169.949115	90.004425	33.676991	...	-0.278982	9.048009	-1.457301	0.003982	0.0	
std	16.466631	0.497955	37.170340	16.590803	15.364394	44.842283	33.385421	35.633072	25.826643	45.431434	...	0.548876	3.472862	2.002430	0.050118	0.0	
min	0.000000	0.000000	105.000000	6.000000	55.000000	0.000000	232.000000	108.000000	0.000000	-172.000000	...	-4.100000	0.000000	-28.600000	0.000000	0.0	
25%	36.000000	0.000000	160.000000	59.000000	80.000000	142.000000	350.000000	148.000000	79.000000	3.750000	...	-0.425000	6.600000	-2.100000	0.000000	0.0	
50%	47.000000	1.000000	164.000000	68.000000	86.000000	157.000000	367.000000	162.000000	91.000000	40.000000	...	0.000000	8.800000	-1.100000	0.000000	0.0	
75%	58.000000	1.000000	170.000000	79.000000	94.000000	175.000000	384.000000	179.000000	102.000000	66.000000	...	0.000000	11.200000	0.000000	0.000000	0.0	
max	83.000000	1.000000	780.000000	176.000000	188.000000	524.000000	509.000000	381.000000	205.000000	169.000000	...	0.000000	23.600000	0.000000	0.800000	0.0	

8 rows × 280 columns

2.1.2 Mengambil Data Target

Proses ini digunakan untuk mengambil data yang dipilih untuk diolah dari dictionary class

Data Target

```
[4] target = data['class']
    target
```

```
0      8
1      6
2     10
3      1
4      7
      ..
447    1
448   10
449    2
450    1
451    1
Name: class, Length: 452, dtype: int64
```

```
[5] target -= 1
    target
```

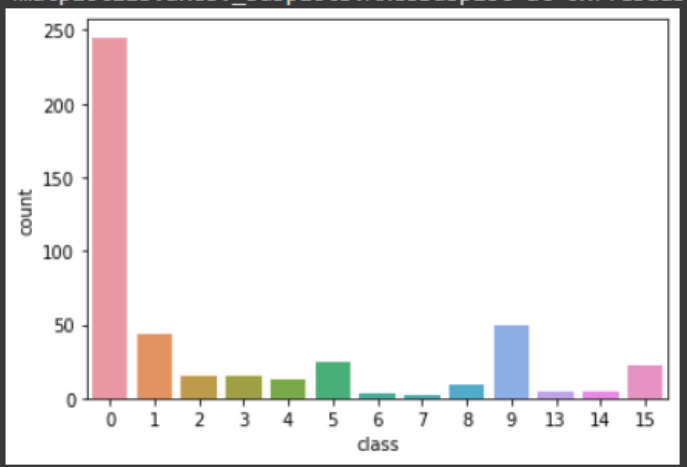
```
0      7
1      5
2      9
3      0
4      6
      ..
447    0
448    9
449    1
450    0
451    0
Name: class, Length: 452, dtype: int64
```

2.1.3 Jumlah Data dalam Target

Untuk menampilkan jumlah data dalam target, dapat menggunakan plot berikut ini.

```
[31] sns.countplot(target, label = "Jumlah")
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7febdabca450>
```



Pada plot di atas didapatkan jumlah target yang paling banyak adalah 0 dengan jumlah hampir 250.

2.2 Ringkasan pra-pemrosesan data

2.2.1 Menghapus Kolom

Menghapus kolom setelah 20, digunakan untuk menghilangkan data yang tidak ada atau sedikit korelasi nya dengan target. Data tersebut tidak memiliki fungsi sehingga dihiraukan

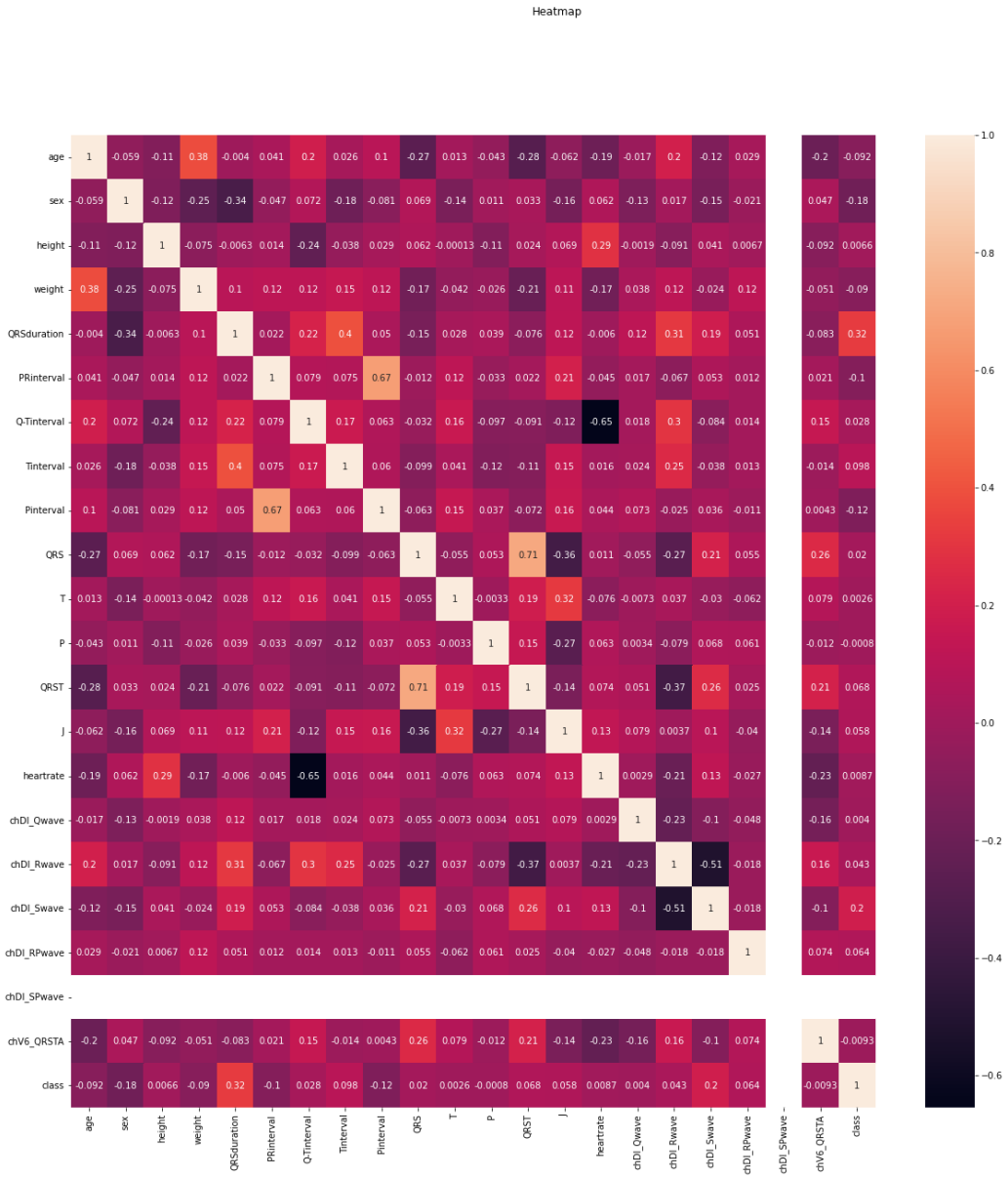
```
#untuk menghapus kolom 20 ke bawah

data.drop(data.columns[20:-2],axis=1, inplace=True)
data
```

2.2.2 Membuat heatmap atau Mencari Korelasi

Mengeksplorasi data dengan menampilkan korelasi pada data set Arrhythmia menggunakan heatmap. Korelasi digunakan untuk melihat skala korelasi data dari yang paling besar dan paling kecil. Korelasi yang tidak ada atau kecil bisa di drop karena tidak diperlukan.

```
#membuat heatmap/korelasi
corr = data.corr()
top_corr_features = corr.index
plt.figure(figsize = (20,20))
plt.suptitle("Heatmap")

#plot heat map
g = sns.heatmap(data[top_corr_features].corr(), annot=True)
```

Sehingga dihasilkan heatmap Korelasi dataset Arrhythmia sebagai berikut.



Saya menggunakan kolom ‘class’ sebagai target yang digunakan

2.2.3 Menampilkan info data setelah diubah

Info data ini digunakan untuk menampilkan info data setelah diubah dan penghapusan kolom.

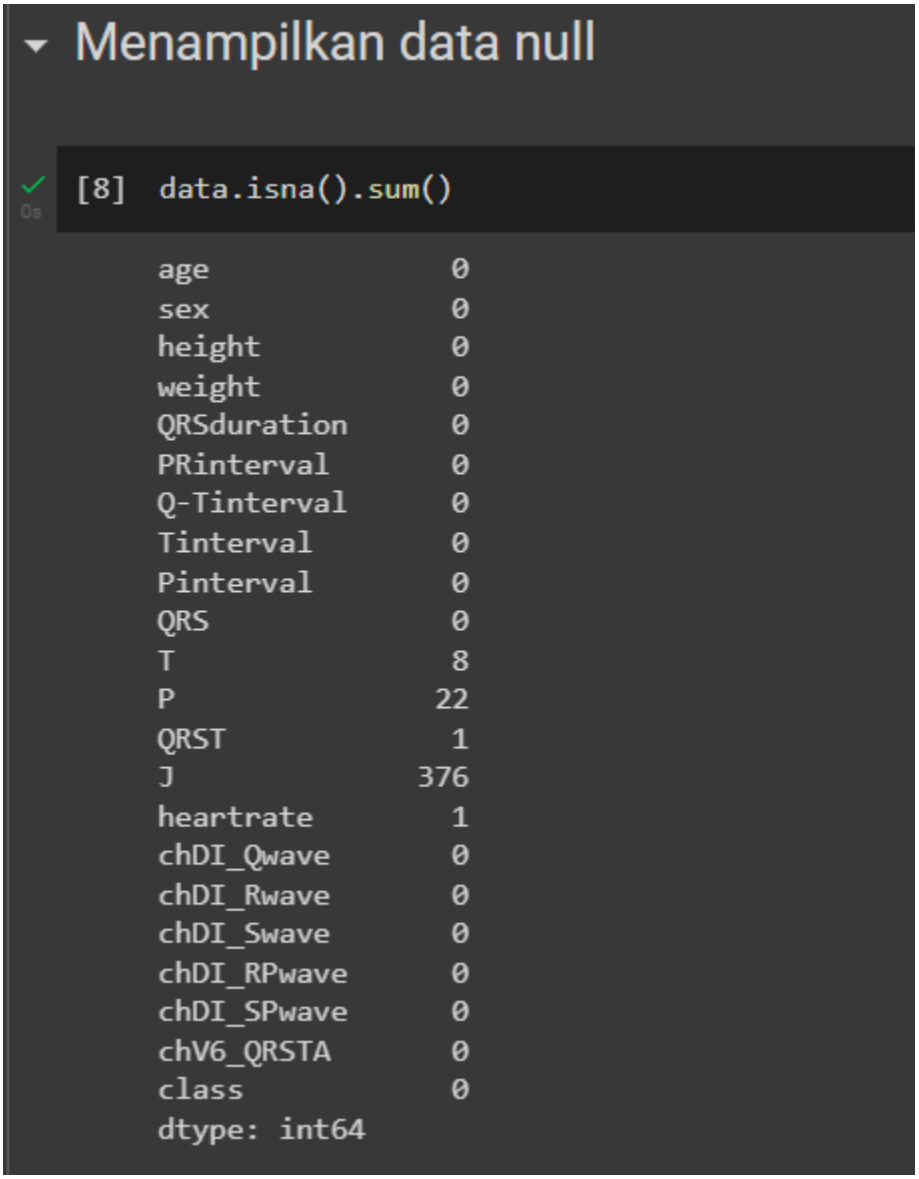
Info data

```
#untuk menampilkan info data

data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 452 entries, 0 to 451
Data columns (total 22 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   age          452 non-null    int64  
 1   sex          452 non-null    int64  
 2   height       452 non-null    int64  
 3   weight       452 non-null    int64  
 4   QRSduration  452 non-null    int64  
 5   PRinterval   452 non-null    int64  
 6   Q-Tinterval  452 non-null    int64  
 7   Tinterval    452 non-null    int64  
 8   Pinterval    452 non-null    int64  
 9   QRS          452 non-null    int64  
 10  T            444 non-null    float64
 11  P            430 non-null    float64
 12  QRST         451 non-null    float64
 13  J            76 non-null     float64
 14  heartrate    451 non-null    float64
 15  chDI_Qwave   452 non-null    int64  
 16  chDI_Rwave   452 non-null    int64  
 17  chDI_Swave   452 non-null    int64  
 18  chDI_RPwave  452 non-null    int64  
 19  chDI_SPwave  452 non-null    int64  
 20  chV6_QRSTA   452 non-null    float64
 21  class        452 non-null    int64  
dtypes: float64(6), int64(16)
memory usage: 77.8 KB
```

2.2.4 Menampilkan Data bernilai Null

Terlihat pada gambar berikut, data yang ditampilkan sebagian besar bernilai 0, dan ada beberapa yang bukan 0. Yaitu T, P, QRST, J, dan heartrate.



2.2.5 Menghapus/Drop Data Selain Null

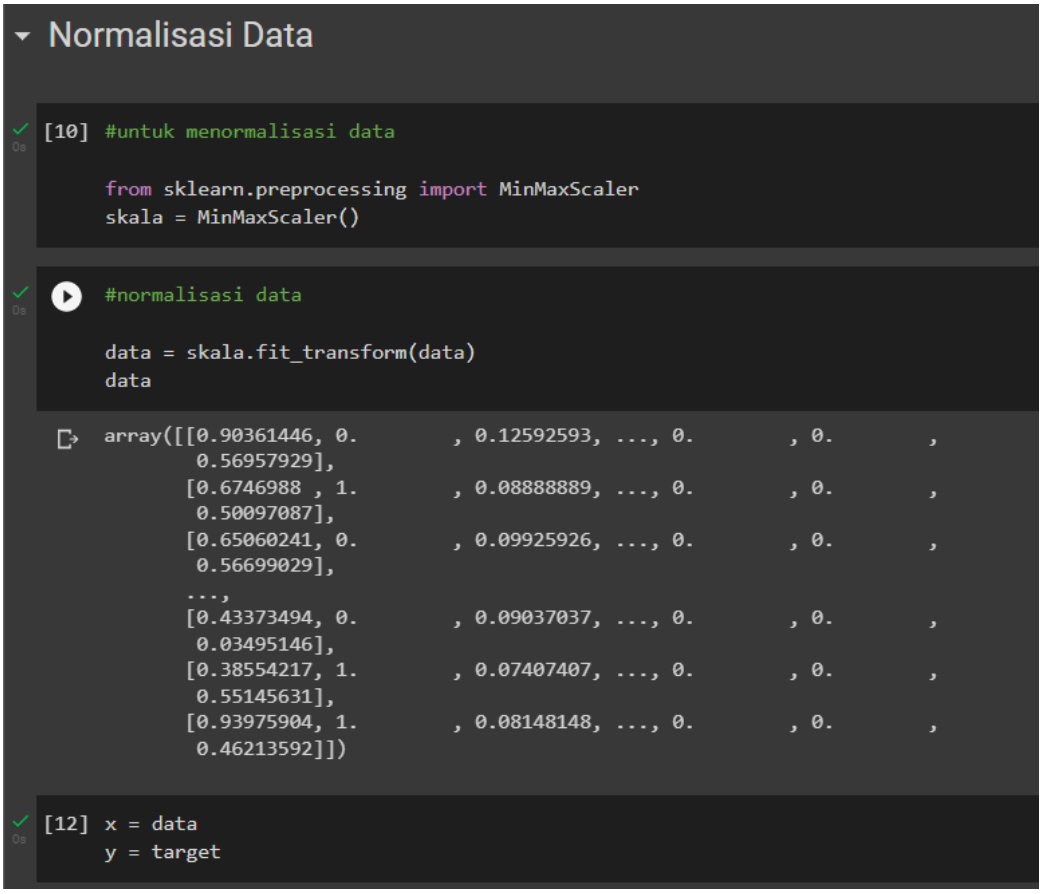
Karena tidak semua data bernilai null atau 0, maka data selain bernilai 0 di drop menggunakan code berikut. Setelah di drop, maka data yang ditampilkan sudah bernilai 0 semua.

```
#drop data yang bukan null

data['P'].fillna(data['P'].mean(), inplace = True)
data['T'].fillna(data['T'].mean(), inplace = True)
data['QRST'].fillna(data['QRST'].mean(), inplace = True)
data['heartrate'].fillna(data['heartrate'].mean(), inplace = True)

data.drop(columns=['J'], inplace=True)
data.drop(columns=['class'], inplace=True)
data.isna().sum()
```

```
age            0
sex            0
height         0
weight         0
QRSduration    0
PRinterval     0
Q-Tinterval    0
Tinterval      0
Pinterval      0
QRS            0
T              0
P              0
QRST           0
heartrate      0
chDI_Qwave     0
chDI_Rwave     0
chDI_Swave     0
chDI_RPwave    0
chDI_SPwave    0
chV6_QRSTA     0
dtype: int64
```

2.2.6 Normalisasi Data

Normalisasi data digunakan untuk pra-processing untuk split data.



Normalisasi Data

```
[10] #untuk menormalisasi data

     from sklearn.preprocessing import MinMaxScaler
     skala = MinMaxScaler()
```

```
     #normalisasi data

     data = skala.fit_transform(data)
     data
```

```
array([[0.90361446, 0.        , 0.12592593, ..., 0.        , 0.        ,
        0.56957929],
       [0.6746988 , 1.        , 0.08888889, ..., 0.        , 0.        ,
        0.50097087],
       [0.65060241, 0.        , 0.09925926, ..., 0.        , 0.        ,
        0.56699029],
       ...,
       [0.43373494, 0.        , 0.09037037, ..., 0.        , 0.        ,
        0.03495146],
       [0.38554217, 1.        , 0.07407407, ..., 0.        , 0.        ,
        0.55145631],
       [0.93975904, 1.        , 0.08148148, ..., 0.        , 0.        ,
        0.46213592]])
```

```
[12] x = data
     y = target
```

BAB 3
PENERAPAN ALGORITMA ANN

Algoritma supervised learning yang digunakan dalam tugas ini adalah ANN (Artificial Neural Network). Saya menggunakan algoritma ini karena tidak terlalu kompleks dan mudah dipahami. Algoritma ini juga selalu menghasilkan output.

Pertama dilakukan split data menjadi x_train, y_train, x_test, dan y_test menggunakan StandardScaler.

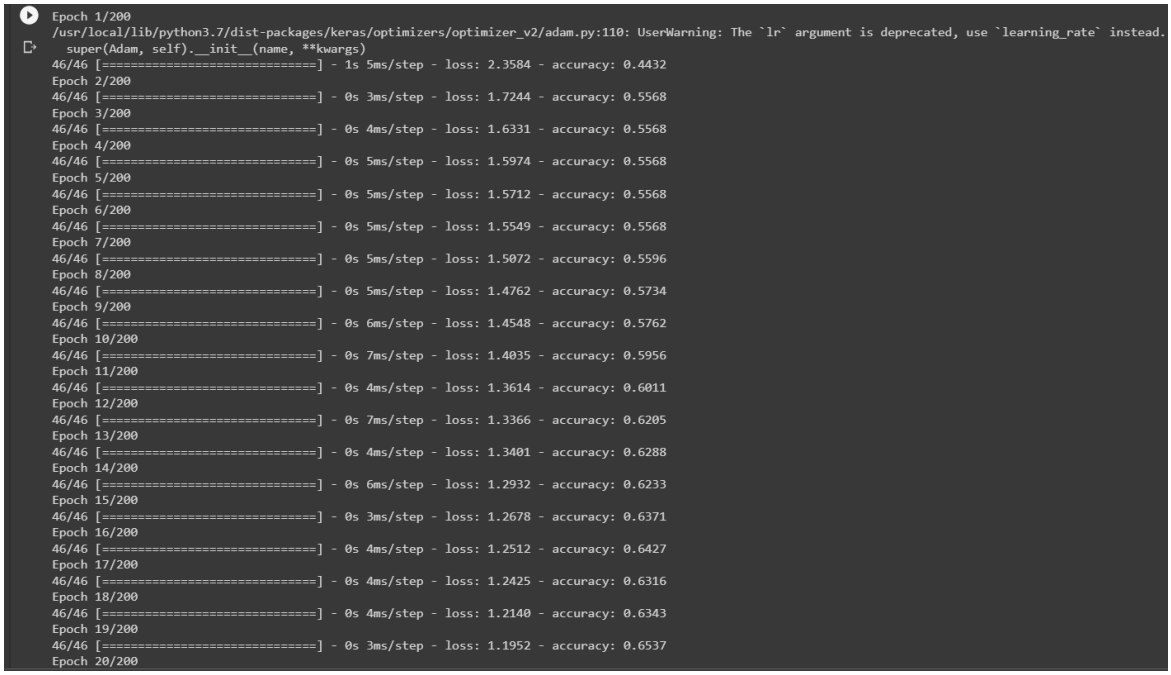
```
#import train test split
from sklearn.model_selection import train_test_split

#Split data menjadi train set dan test set
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
```

Implementasi Algoritma Supervised ANN

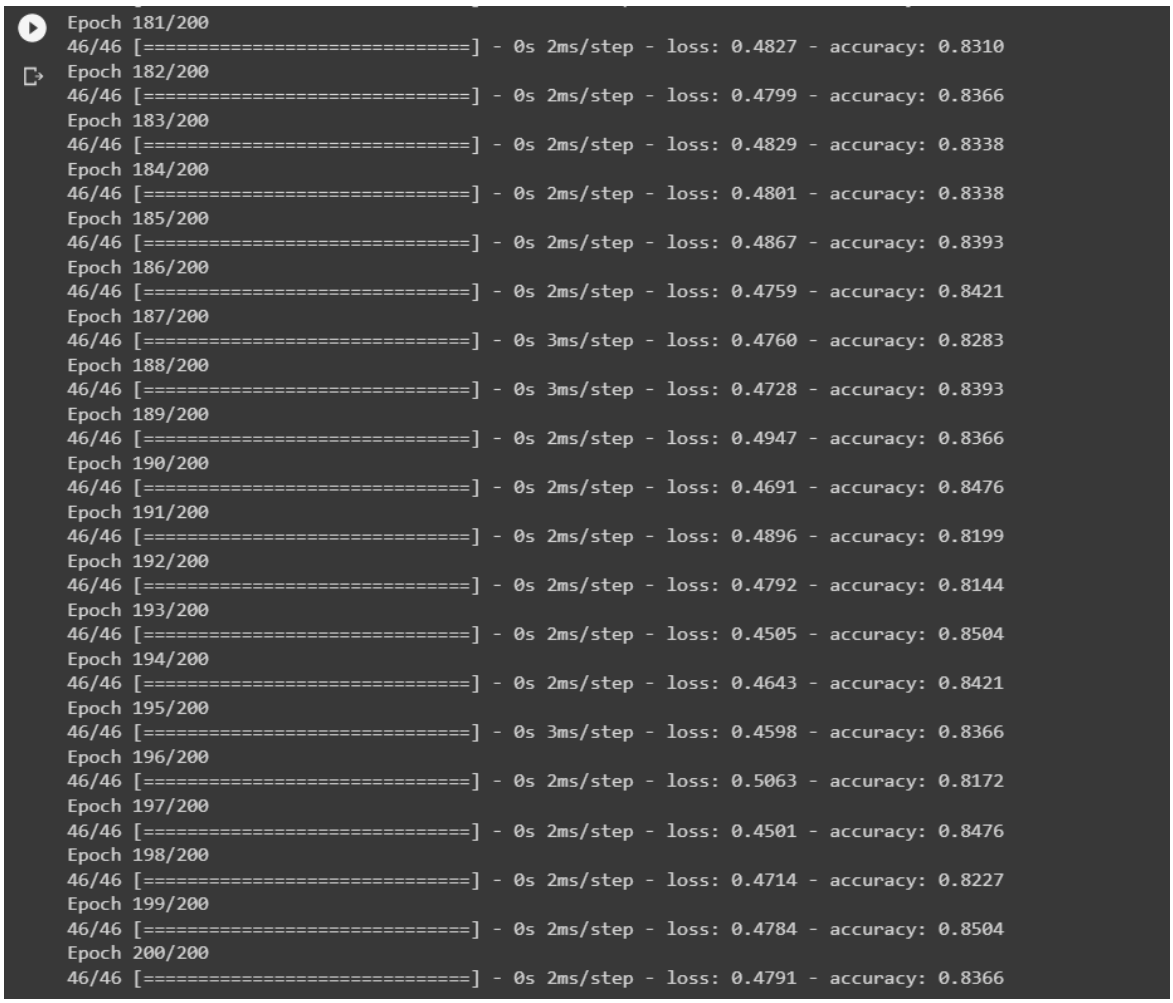
```
buatModel = keras.Sequential([
    keras.layers.Dense(16, input_shape=(20,), activation='relu'),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dense(64, activation='relu'),
    keras.layers.Dense(16, activation='softmax')
])


optimizer = keras.optimizers.Adam(lr= 0.001)
buatModel.compile(optimizer=optimizer,
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

# Constructing the input
x = x_train
y = y_train

# training
training = buatModel.fit(x,y,epochs = 200,batch_size = 8)
```

Pada code tersebut, terdapat variable buatModel untuk menyimpan metode atau proses pembuatan algoritma supervised learning ANN. Kemudian data akan melalui proses training yang dapat dilihat pada code di baris terakhir. Dan juga akan memunculkan nilai akurasi nya.

Hasil dari tahap ini akan memunculkan hasil seperti berikut ini. Dengan epoch sebanyak 200



```
Epoch 1/200
/usr/local/lib/python3.7/dist-packages/keras/optimizers/optimizer_v2/adam.py:110: UserWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  super(Adam, self).__init__(name, **kwargs)
46/46 [==============================] - 1s 5ms/step - loss: 2.3584 - accuracy: 0.4432
Epoch 2/200
46/46 [==============================] - 0s 3ms/step - loss: 1.7244 - accuracy: 0.5568
Epoch 3/200
46/46 [==============================] - 0s 4ms/step - loss: 1.6331 - accuracy: 0.5568
Epoch 4/200
46/46 [==============================] - 0s 5ms/step - loss: 1.5974 - accuracy: 0.5568
Epoch 5/200
46/46 [==============================] - 0s 5ms/step - loss: 1.5712 - accuracy: 0.5568
Epoch 6/200
46/46 [==============================] - 0s 5ms/step - loss: 1.5549 - accuracy: 0.5568
Epoch 7/200
46/46 [==============================] - 0s 5ms/step - loss: 1.5072 - accuracy: 0.5596
Epoch 8/200
46/46 [==============================] - 0s 5ms/step - loss: 1.4762 - accuracy: 0.5734
Epoch 9/200
46/46 [==============================] - 0s 6ms/step - loss: 1.4548 - accuracy: 0.5762
Epoch 10/200
46/46 [==============================] - 0s 7ms/step - loss: 1.4035 - accuracy: 0.5956
Epoch 11/200
46/46 [==============================] - 0s 4ms/step - loss: 1.3614 - accuracy: 0.6011
Epoch 12/200
46/46 [==============================] - 0s 7ms/step - loss: 1.3366 - accuracy: 0.6205
Epoch 13/200
46/46 [==============================] - 0s 4ms/step - loss: 1.3401 - accuracy: 0.6288
Epoch 14/200
46/46 [==============================] - 0s 6ms/step - loss: 1.2932 - accuracy: 0.6233
Epoch 15/200
46/46 [==============================] - 0s 3ms/step - loss: 1.2678 - accuracy: 0.6371
Epoch 16/200
46/46 [==============================] - 0s 4ms/step - loss: 1.2512 - accuracy: 0.6427
Epoch 17/200
46/46 [==============================] - 0s 4ms/step - loss: 1.2425 - accuracy: 0.6316
Epoch 18/200
46/46 [==============================] - 0s 4ms/step - loss: 1.2140 - accuracy: 0.6343
Epoch 19/200
46/46 [==============================] - 0s 3ms/step - loss: 1.1952 - accuracy: 0.6537
Epoch 20/200
```



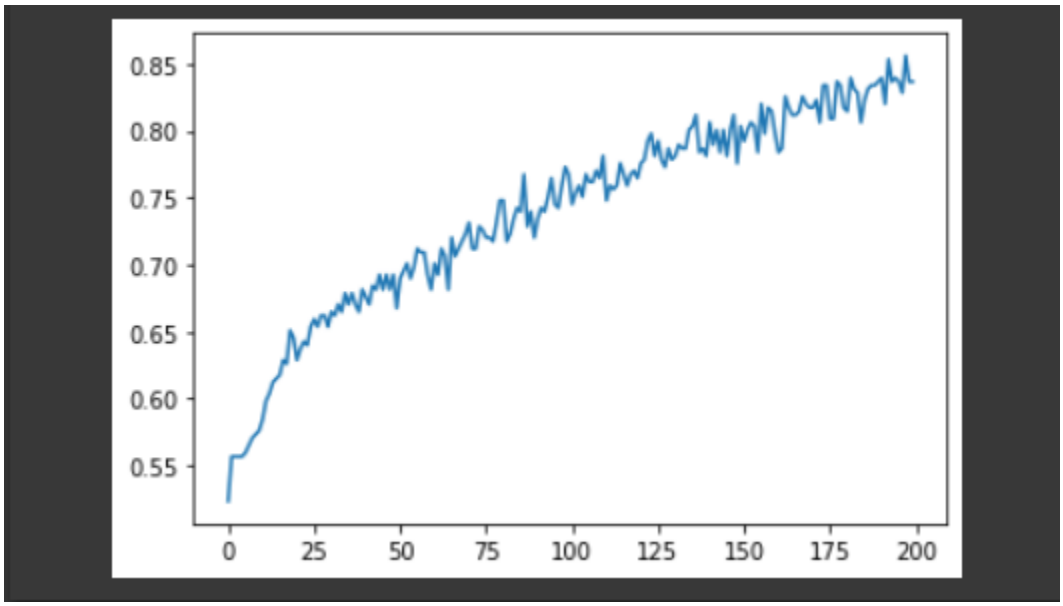
```
Epoch 181/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4827 - accuracy: 0.8310
Epoch 182/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4799 - accuracy: 0.8366
Epoch 183/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4829 - accuracy: 0.8338
Epoch 184/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4801 - accuracy: 0.8338
Epoch 185/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4867 - accuracy: 0.8393
Epoch 186/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4759 - accuracy: 0.8421
Epoch 187/200
46/46 [==============================] - 0s 3ms/step - loss: 0.4760 - accuracy: 0.8283
Epoch 188/200
46/46 [==============================] - 0s 3ms/step - loss: 0.4728 - accuracy: 0.8393
Epoch 189/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4947 - accuracy: 0.8366
Epoch 190/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4691 - accuracy: 0.8476
Epoch 191/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4896 - accuracy: 0.8199
Epoch 192/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4792 - accuracy: 0.8144
Epoch 193/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4505 - accuracy: 0.8504
Epoch 194/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4643 - accuracy: 0.8421
Epoch 195/200
46/46 [==============================] - 0s 3ms/step - loss: 0.4598 - accuracy: 0.8366
Epoch 196/200
46/46 [==============================] - 0s 2ms/step - loss: 0.5063 - accuracy: 0.8172
Epoch 197/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4501 - accuracy: 0.8476
Epoch 198/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4714 - accuracy: 0.8227
Epoch 199/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4784 - accuracy: 0.8504
Epoch 200/200
46/46 [==============================] - 0s 2ms/step - loss: 0.4791 - accuracy: 0.8366
```

BAB 4
EVALUASI MODEL

Code di bawah ini digunakan untuk memunculkan plot akurasi dari data set Arrhythmia. Apabila nilai Epoch semakin tinggi, maka akurasinya akan semakin tinggi pula.

```
plt.plot(training.history['accuracy'], linestyle = 'solid')
plt.show()
```

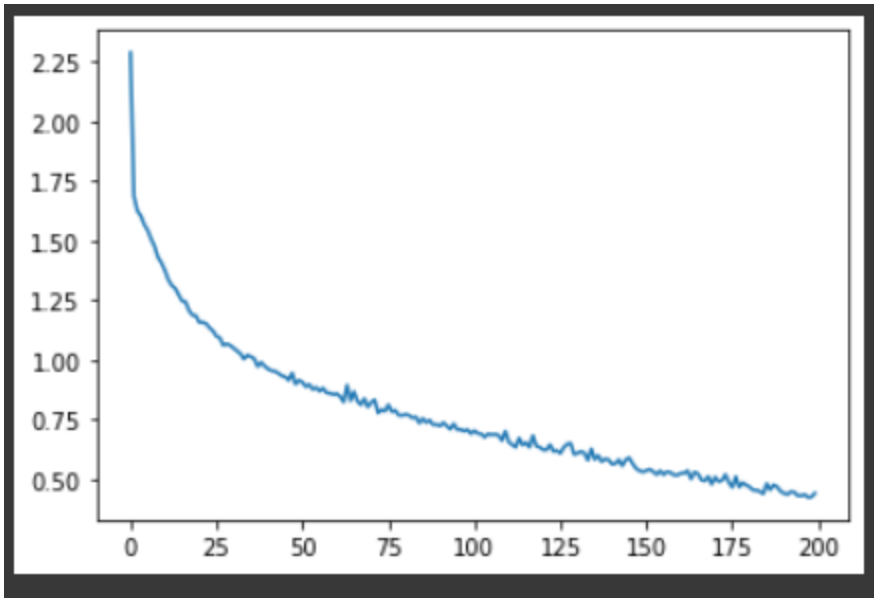
Dihasilkan plot seperti berikut. Terlihat pada plot ini, pada nilai Epoch 200, nilai akurasinya adalah 85% ke atas atau semakin tinggi.



Code di bawah ini menunjukkan selisih antara prediksi dengan data aslinya. Dan semakin banyak proses training maka lost nya akan makin kecil.

```
plt.plot(training.history['loss'], linestyle = 'solid')
plt.show()
```

Plot di bawah ini terbukti bahwa lost semakin kecil yaitu berada di bawah 50% saat Epoch nya semakin besar. Apabila lost nya semakin kecil, maka selisih antara prediksi dan data aslinya semakin kecil (akurat)



Untuk menampilkan akurasinya menggunakan kode berikut

```
b, s = 0, 0
prediksi = buatModel.predict(x_test)
for p, y in zip(prediksi, y_test):
  pr = np.argmax(p)
  print("Prediksi:", pr, "Y Test:", y)
  if(pr == y) :
    b = b + 1
  else :
    s = s + 1

print("Benar:", b)
print("Salah:", s)
akur = b/(b+s)*100
print("Akurasi:", akur)
```

Dengan code tersebut didapatkan akurasi antara prediksi dengan data sebenarnya. Didapatkan hasil seperti berikut. Hasil Pemrosesan tersebut dikatakan benar apabila prediksi dengan test data berhasil,

```
Prediksi: 0 Y Test: 9
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 2 Y Test: 9
Prediksi: 0 Y Test: 0
Prediksi: 5 Y Test: 5
Prediksi: 0 Y Test: 0
Prediksi: 1 Y Test: 14
Prediksi: 9 Y Test: 9
Prediksi: 5 Y Test: 5
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 15
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 1
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 8 Y Test: 9
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 9
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 9 Y Test: 0
Prediksi: 15 Y Test: 9
Prediksi: 0 Y Test: 7
Prediksi: 0 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 15 Y Test: 9
Prediksi: 14 Y Test: 0
Prediksi: 1 Y Test: 0
Prediksi: 0 Y Test: 0
Prediksi: 9 Y Test: 9
Prediksi: 0 Y Test: 0
Prediksi: 15 Y Test: 2
Prediksi: 0 Y Test: 0
Prediksi: 9 Y Test: 15
Prediksi: 0 Y Test: 0
Prediksi: 1 Y Test: 1
Prediksi: 15 Y Test: 1
Prediksi: 0 Y Test: 0
```

Dengan akurasi lebih dari 50% dengan prediksi benar = 47, dan salah = 44.

```
Benar: 47
Salah: 44
Akurasi: 51.648351648351664
```

BAB 4
LAMPIRAN

4.1 Lampiran Video

https://youtu.be/lw6O01qBatY

4.2 Lampiran Code Program

https://colab.research.google.com/drive/1Fxf5Ygidovkh0qMmUdGk57Xp32YZ48XO?usp=sharing

4.3 Lampiran Slide

https://www.canva.com/design/DAFRdsgceXE/8oFDwv0ng5-AzQTbZpufow/view?utm_content=DAFRdsgceXE&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton