Klasifikasi Jenis Kanker Payudara Menggunakan Pendekatan Model pada *Machine Learning*

Makalah Ini Disusun Sebagai Proyek UAS Kelompok Sains Data

Fasilitator: Bevina Desjwiandra Handari

Disusun oleh:

Deanarani Kharisma (2106726125)
Dimas Satrio Adjie (2106651276)
Roswita Bellinda (2106722505)
Salsabila H. Sastro (2106722726)

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS INDONESIA

DEPOK

2023

Abstrak

Makalah ini akan membahas salah satu aplikasi *Neural Network* yang digunakan untuk mengklasifikasikan jenis kanker payudara. *Artificial Neural Network* adalah jaringan dari sekelompok unit pemroses kecil (*neural*/neuron) yang dimodelkan dan ditiru berdasarkan seperti jaringan saraf manusia, yaitu neuron. *Artificial Neural Network* merupakan suatu sistem yang dapat beradaptasi dengan mengubah struktur-strukturnya dan akan menyelesaikan suatu masalah berdasarkan informasi-informasi yang mengalir baik itu eksternal maupun internal.

Kata kunci: Data, Klasifikasi, *Machine Learning*, *Neural Network*, *Artificial Neural Network*.

1. PENDAHULUAN

1.1 Latar Belakang

Kanker payudara adalah kanker yang paling sering terjadi pada wanita, berdampak pada 2,1 juta wanita setiap tahun, dan juga menyebabkan kematian terkait kanker pada wanita. Diperkirakan 627.000 wanita meninggal karena kanker payudara, yaitu sekitar 15% dari semua kematian akibat kanker di kalangan wanita.

Sel kanker payudara yang pertama dapat tumbuh menjadi tumor sebesar 1 cm pada waktu 8–12 tahun. Sel kanker tersebut diam pada kelenjar payudara. Sel-sel kanker payudara ini dapat menyebar melalui aliran darah ke seluruh tubuh. Sel kanker payudara dapat bersembunyi di dalam tubuh kita selama bertahun-tahun tanpa kita ketahui, dan tiba-tiba aktif menjadi tumor ganas atau kanker.

Secara umum, terdapat dua jenis tumor payudara, yaitu tumor payudara ganas dan tumor payudara jinak. Keduanya tentu tidak sama, perbedaan paling mendasar antara tumor payudara ganas dan jinak adalah perkembangannya. Secara umum, tumor ganas berkembang menjadi sel kanker, sedangkan tumor jinak adalah non-kanker dan dapat sembuh dengan sendirinya, tetapi mungkin memerlukan perawatan khusus.

Berdasarkan uraian di atas, dalam pemodelan masalah ini, kami berencana untuk membuat model *neural network* sebagai alternatif penyelesaian masalah klasifikasi kanker payudara. Model *neural network* yang dibuat memanfaatkan data dari 32 fitur, terdiri atas *radius*, *texture*, *perimeter*, *area*, *smoothness*, *compactness*, dan fitur-fitur lainnya yang akan disebutkan rinci di bagian selanjutnya. Diagnosis nantinya akan dipilih sebagai variabel output dan komponen lainnya sebagai variabel prediktor. Pada akhir pemodelan ini, besar harapan kami akan memperoleh model *neural network* yang dapat mengestimasi dan mengklasifikasikan jenis kanker payudara pada keadaan tertentu, yang mana modelnya sendiri juga harus memenuhi standar kelayakan model yang ada.

1.2 Rumusan Masalah

Rumusan masalah dalam penelitian kami sebagai berikut.

1. Bagaimana akurasi klasifikasi model *neural network* berdasarkan data yang tersedia?
2. Bagaimana hubungan antara diagnosis dengan fitur-fitur lainnya yang digunakan dalam pemodelan?
3. Bagaimana hasil analisis data terhadap model *neural network* yang diperoleh?

1.3 Tujuan Penelitian

Berdasarkan rumusan masalah di atas, tujuan penelitian kami adalah sebagai berikut.

1. Mengetahui akurasi klasifikasi model *neural network* berdasarkan data yang tersedia.
2. Mengetahui hubungan antara diagnosis dengan fitur-fitur lainnya yang digunakan dalam pemodelan.
3. Mengetahui hasil analisis data terhadap model *neural network* yang diperoleh.

2. ISI

2.1 Data dan Metode

Model *neural network* untuk masalah klasifikasi jenis kanker payudara yang dibuat memanfaatkan *dataset* Breast Cancer Wisconsin (sumber: https://archive.ics.uci.edu/) yang terdiri dari 569 baris dan 32 fitur. Namun, yang digunakan hanya menjadi 28 fitur kontinu yang memiliki korelasi yang paling baik dengan variabel target. Variabel target yang dimaksud jenis tumor kanker payudaranya, apakah tumor ganas (malignant) atau tumor jinak (benign). Inilah yang nantinya akan menjadi output atau tujuan dari model *neural network* yang kami bangun. Berikut fitur-fitur beserta penjelasannya pada *dataset* kami:

a. ID: Nomor pasien
b. Diagnosis: Apakah tumornya ganas atau jinak
c. Radius: Jarak titik tengah dengan lingkaran payudaranya
d. Texture: Standar deviasi dari nilai skala abu-abu
e. Perimeter: Besarnya tumor
f. Area: Daerah terkenanya tumor
g. Smoothness: Variasi lokal dengan panjang radius
h. Compactness = (perimeter^2) / (area - 1.0)
i. Concavity: Kerasnya bagian cekung dari garis bentuknya
j. Concave points: Banyaknya bagian cekung dari garis bentuknya
k. Symmetry: Kesimetrisan
l. Fractal dimension = aproksimasi coastline - 1
m. Mean: Rata-rata

n. Se: Standard error
o. Worst: Rata-rata terbesar

Adapun metode yang kami gunakan dalam pemodelan ini adalah pengolahan, analisis, dan visualisasi data numerik dengan menggunakan Google Colaboratory dengan bahasa pemrograman Python pada tautan berikut project-sainsdata-kelompok1. Proses pemodelan secara garis besar terdiri atas *import* data, *data cleaning* dan *data preprocessing*, membangun model dengan Keras dan TensorFlow, *train* dan melihat akurasi model, dan terakhir *test* model dengan sebuah contoh input.

2.2 Implementasi

2.2.1 *Importing the Dependencies*

Mengimpor beberapa fungsi dan *libraries* seperti *numpy, pandas, matplotlib.pyplot, sklearn datasets,* dan *train test* untuk bisa menjalankan program ini.

```
[ ]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import sklearn.datasets
     from sklearn.model_selection import train_test_split
     import matplotlib.pyplot as plt
     import seaborn as sns
     %matplotlib inline
```

2.2.2 *Data Importation*

- Dalam penelitian ini, kami menggunakan data sebanyak 569 baris dengan 31 fitur, yang semuanya adalah data numerik.

```
[ ]  # print jumlah baris dan kolom
     data_frame.shape

     (569, 31)
```

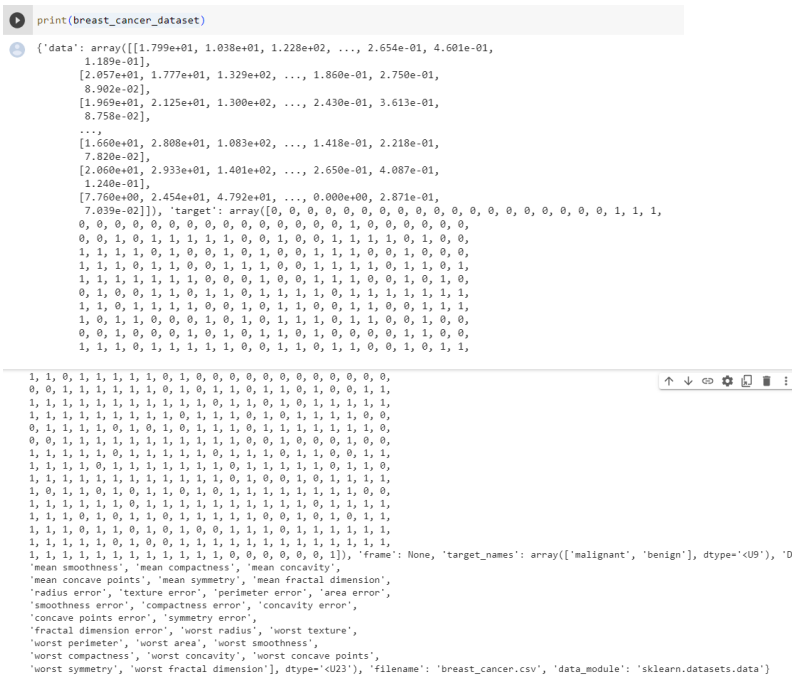
```
print(breast_cancer_dataset)

{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
         1.189e-01],
        [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
         8.902e-02],
        [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
         8.758e-02],
        ...,
        [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
         7.820e-02],
        [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
         1.240e-01],
        [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
```

2.2.3 *Data Collection & Processing*

- Menggunakan fungsi berikut untuk memanggil *dataset* kanker payudara.

```
[ ]  # input data dari sklearn
     breast_cancer_dataset = sklearn.datasets.load_breast_cancer()
```

- Berikut hasilnya dapat dilihat menampilkan nilai fitur dan elemen. Serta target dimana 0 mewakili tumor ganas dan 1 mewakili tumor jinak.



```
print(breast_cancer_dataset)
```

```
{'data': array([[1.799e+01, 1.038e+01, 1.228e+02, ..., 2.654e-01, 4.601e-01,
        1.189e-01],
       [2.057e+01, 1.777e+01, 1.329e+02, ..., 1.860e-01, 2.750e-01,
        8.902e-02],
       [1.969e+01, 2.125e+01, 1.300e+02, ..., 2.430e-01, 3.613e-01,
        8.758e-02],
       ...,
       [1.660e+01, 2.808e+01, 1.083e+02, ..., 1.418e-01, 2.218e-01,
        7.820e-02],
       [2.060e+01, 2.933e+01, 1.401e+02, ..., 2.650e-01, 4.087e-01,
        1.240e-01],
       [7.760e+00, 2.454e+01, 4.792e+01, ..., 0.000e+00, 2.871e-01,
        7.039e-02]]), 'target': array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 0,
1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0,
1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1,
1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0,
0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1,
1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1,
1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1,
1, 1, 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, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1]), 'frame': None, 'target_names': array(['malignant', 'benign'], dtype='<U9'), 'D
'mean smoothness', 'mean compactness', 'mean concavity',
'mean concave points', 'mean symmetry', 'mean fractal dimension',
'radius error', 'texture error', 'perimeter error', 'area error',
'smoothness error', 'compactness error', 'concavity error',
'concave points error', 'symmetry error',
'fractal dimension error', 'worst radius', 'worst texture',
'worst perimeter', 'worst area', 'worst smoothness',
'worst compactness', 'worst concavity', 'worst concave points',
'worst symmetry', 'worst fractal dimension'], dtype='<U23'), 'filename': 'breast_cancer.csv', 'data_module': 'sklearn.datasets.data'}
```

2.2.4 Membuat Data Frame

- Menggunakan pandas library untuk membuat data frame yang berisi kolom berupa fitur-fitur dari *dataset*.

```
[ ]  # input data ke data frame
     data_frame = pd.DataFrame(breast_cancer_dataset.data, columns = breast_cancer_dataset.feature_names)
```

- Membuat kolom baru untuk fitur target yang diberi nama “label”.

```
[ ]  # menambahkan target pada data
     data_frame['label'] = breast_cancer_dataset.target
```

- Mencari informasi data menggunakan fungsi berikut untuk melihat berapa banyak null dan non-null pada data, sehingga dapat dibersihkan, serta statistik data.

```
[ ] # memeriksa missing values
    data_frame.isnull().sum()
```

```
mean radius                0
mean texture               0
mean perimeter             0
mean area                  0
mean smoothness            0
mean compactness           0
mean concavity             0
mean concave points        0
mean symmetry              0
mean fractal dimension     0
radius error               0
texture error              0
perimeter error            0
area error                 0
smoothness error           0
compactness error          0
concavity error            0
concave points error       0
symmetry error             0
fractal dimension error    0
worst radius               0
worst texture              0
worst perimeter            0
worst area                 0
worst smoothness           0
worst compactness          0
worst concavity            0
worst concave points       0
worst symmetry             0
worst fractal dimension    0
label                      0
dtype: int64
```

```
# melihat gambaran awal dari data
data_frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   mean radius              569 non-null    float64
 1   mean texture             569 non-null    float64
 2   mean perimeter           569 non-null    float64
 3   mean area                569 non-null    float64
 4   mean smoothness          569 non-null    float64
 5   mean compactness         569 non-null    float64
 6   mean concavity           569 non-null    float64
 7   mean concave points      569 non-null    float64
 8   mean symmetry            569 non-null    float64
 9   mean fractal dimension   569 non-null    float64
 10  radius error             569 non-null    float64
 11  texture error            569 non-null    float64
 12  perimeter error          569 non-null    float64
 13  area error               569 non-null    float64
 14  smoothness error         569 non-null    float64
 15  compactness error        569 non-null    float64
 16  concavity error          569 non-null    float64
 17  concave points error     569 non-null    float64
 18  symmetry error           569 non-null    float64
 19  fractal dimension error  569 non-null    float64
 20  worst radius             569 non-null    float64
 21  worst texture            569 non-null    float64
 22  worst perimeter          569 non-null    float64
 23  worst area               569 non-null    float64
 24  worst smoothness         569 non-null    float64
 25  worst compactness        569 non-null    float64
 26  worst concavity          569 non-null    float64
 27  worst concave points     569 non-null    float64
 28  worst symmetry           569 non-null    float64
 29  worst fractal dimension  569 non-null    float64
 30  label                    569 non-null    int64
dtypes: float64(30), int64(1)
```

```
[ ] # melihat deskripsi statistik
    data_frame.describe()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200

8 rows × 31 columns

- Melakukan pengelompokkan data dan mencari rata-ratanya untuk setiap label.

```
[ ] data_frame.groupby('label').mean()
```

label	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	me symmet
0	17.462830	21.604906	115.365377	978.376415	0.102898	0.145188	0.160775	0.087990	0.1929
1	12.146524	17.914762	78.075406	462.790196	0.092478	0.080085	0.046058	0.025717	0.1741

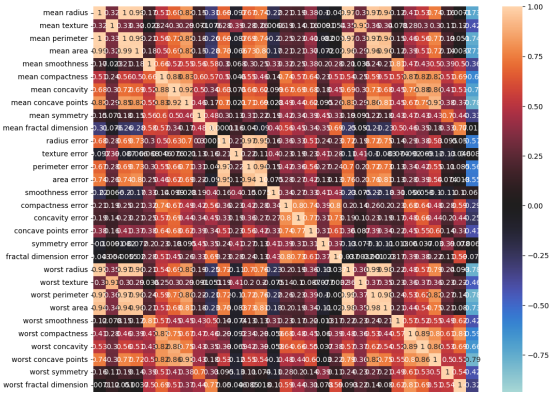
2 rows × 30 columns

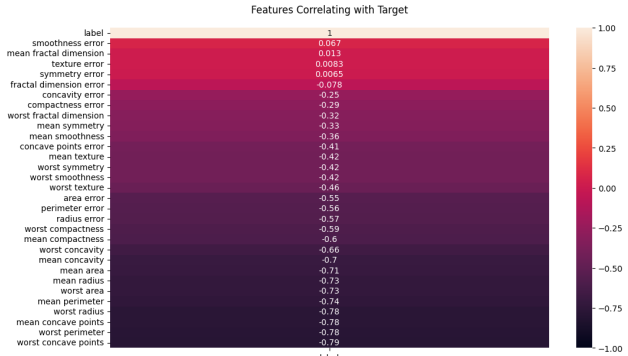
2.2.5 *Exploratory Data Analysis*

- Mencari *Pearson Correlation* antarfitur dan antara fitur serta target.

```
#Pearson Correlation
corr = data_frame.corr()
plt.subplots(figsize=(12,10))
sns.heatmap(corr, vmin=-1, center=0, vmax=1, annot=True)
plt.show()
```

```
[ ] plt.figure (figsize = (12, 7), dpi = 100)
    heatmap = sns.heatmap (data_frame.corr()[['label']].sort_values (by = 'label', ascending = False), vmin = -1, vmax = 1, annot = True)
    heatmap.set_title ('Features Correlating with Target', fontdict = {'fontsize':12}, pad = 18);
```





- *Drop* fitur dengan korelasi rendah, yaitu "*texture error*" dan "*symmetry error*".

```
[ ]  #drop fitur dengan korelasi rendah

     data = data_frame.drop(columns='texture error', axis=1)
     data = data.drop(columns='symmetry error', axis=1)
     print(data)
```

2.2.6 Memisahkan fitur dengan target

Dalam *machine learning* ataupun *neural network,* fitur dan target haruslah dipisah, oleh karena itu kami melakukan beberapa hal berikut.

- Membuat variabel baru, yaitu X dan Y

```
[ ]  X = data.drop(columns='label', axis=1)
     Y = data['label']
```

Dimana X berisi semua kolom kecuali label dan Y berisi kolom label.

2.2.7 Membagi data menjadi *data test* dan *data train*

Sebelum membuat *neural network*, pastikan mengambil beberapa persen dari data untuk diuji.

```
[ ]  X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

- X_train berisi data pelatihan dari fitur dan label yang sesuai akan disimpan di Y_train.
- X_test berisi fitur data uji dan semua label yang akan disimpan di Y_test.
- Dengan menggunakan *test_size* = 0,2 karena kami menggunakan 20% data menjadi data uji dengan *random_state* = 2.

2.2.8 Standarisasi Data

Untuk meningkatkan tingkat akurasi kebenaran data, sangat penting untuk melakukan standarisasi data.

```
[ ] from sklearn.preprocessing import StandardScaler
```

```
[ ] scaler = StandardScaler()

    X_train_std = scaler.fit_transform(X_train)

    X_test_std = scaler.transform(X_test)
```

2.2.9 *Deep Neural Network*

Bagian terpenting adalah membangun *neural network*, berikut langkah yang kami gunakan.

- Import tensorflow dan keras.

```
[ ] # import tensorflow and Keras
    import tensorflow as tf
    tf.random.set_seed(3)
    from tensorflow import keras
```

Tensorflow adalah perpustakaan pembelajaran mendalam yang dibuat oleh Google sehingga memudahkan kita untuk membuat *neural network,* sedangkan Keras adalah pembungkus tensorflow. Random seed digunakan untuk mendapatkan nilai akurasi yang sama berapa kali kita menjalankan codenya.

- Membuat layer dan *neural network*.

```
[ ] # set layers untuk Neural Network

    model = keras.Sequential([
                        keras.layers.Flatten(input_shape=(28,)), #flatten input menjadi single dime
                        keras.layers.Dense(20, activation='relu'), #hidden layers
                        keras.layers.Dense(2, activation='relu') #output
    ])
```

```
[ ] # compile Neural Network

    model.compile(optimizer='adam',
                  loss='sparse_categorical_crossentropy',
                  metrics=['accuracy'])
```

- Melakukan pelatihan *neural network*.

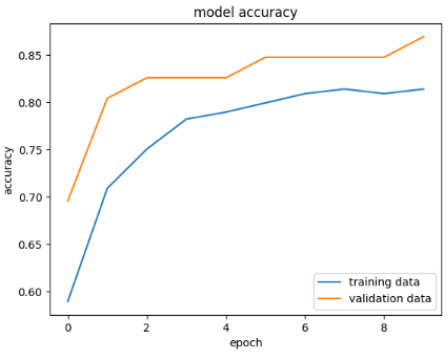
```
[ ] # train Neural Network

    history = model.fit(X_train_std, Y_train, validation_split=0.1, epochs=10)
```

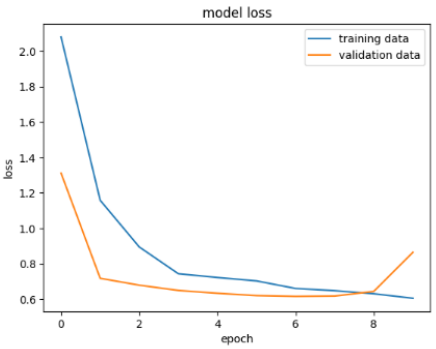
2.2.10 *Visualizing Accuracy and Loss*

Untuk melihat tingkat keakuratan model dan kesalahan maka dibuat visualisasi dari variabel ‘history’ yang berisi *training data* agar mudah dimengerti menggunakan fungsi berikut.

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])

plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')

plt.legend(['training data', 'validation data'], loc = 'lower right')
```

Lalu menghasilkan grafik akurasi model dari *train data* dan *validation data* serta tingkat *loss* sebagai berikut, dengan *validation data* adalah data yang sebagian besar digunakan untuk mendeskripsikan evaluasi dari model saat *hyperparameter tuning* dan *data prepration*, sedangkan *test data* adalah data yang digunakan untuk mendeskripsikan evaluasi dari model final untuk dibandingkan dengan model final lainnya.





2.2.11 Mengecek keakuratan mode pada *data test*

- Bisa dilihat bahwa tingkat akurasi model adalah 93,85%

```
loss, accuracy = model.evaluate(X_test_std, Y_test)
print(accuracy)

4/4 [==============================] - 0s 5ms/step - loss: 0.1693 - accuracy: 0.9386
0.9385964870452881
```

- Membangun model.predict() untuk memberikan nilai probabilitas kelas untuk *data point*.

```
[ ]  Y_pred = model.predict(X_test_std)
```

- Memanggil fungsi *argmax* untuk pengklasifikasian.

```
[ ]  #  argmax function

     my_list = [0.25, 0.56]

     index_of_max_value = np.argmax(my_list)
```

- Melakukan tes ulang untuk model.predict() dan fungsi *argmax* untuk memprediksi probabilitas label.

```
[ ]  # converting the prediction probability to class labels

     Y_pred_labels = [np.argmax(i) for i in Y_pred]
     print(Y_pred_labels)

     [0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0,
```

2.2.12 *Building the predictive system*

Tahap terakhir adalah membangun sistem prediksi, berikut algoritma yang kami gunakan.

- Memasukkan data dari 30 fitur yang dipakai.

```
input_data = (11.76,21.6,74.72,
```

- Mengubah data yang sudah diinput menjadi *numpy array*.

```
# Mengubah data yang sudah diinput menjadi numpy array
input_data_as_numpy_array = np.asarray(input_data)
```

- Membentuk kembali *numpy array* seperti yang kami perkirakan untuk satu *data point*.

```
# Membentuk kembali numpy array seperti yang kami perkirakan untuk satu data point.
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)
```

- Melakukan standarisasi data agar tingkat akurasi kebenaran tinggi.

```
# Melakukan standarisasi data agar tingkat akurasi kebenaran tinggi
input_data_std = scaler.transform(input_data_reshaped)
```

- Membuat model.prediction() dari data yang sudah diinput.

```
prediction = model.predict(input_data_std)
print(prediction)
```

- Memasukkan fungsi *argmax* agar dapat diklasifikasikan ke label atau target.

```
prediction_label = [np.argmax(prediction)]
print(prediction_label)
```

- Mencetak hasil, apabila *prediction_label* sama dengan 0 maka tumor tersebut ganas, selain nilai 0 maka tumor tersebut jinak.

```
if(prediction_label[0] == 0):
  print('The tumor is Malignant')

else:
  print('The tumor is Benign')
```

- Berikut contoh hasil yang ditunjukkan dengan menggunakan salah satu data pada data set.

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py
  warnings.warn(
1/1 [==============================] - 0s 144ms/step
[[0.        1.0687064]]
[1]
The tumor is Benign
```

2.3 Analisis Data

Inferensi dari EDA (*Exploratory Data Analysis*) dan *Data Test*:

1. Tidak ada nilai yang hilang.
2. Semua nilainya adalah nilai numerik kontinu.
3. Mean sedikit lebih tinggi daripada median untuk sebagian besar fitur. Oleh karena itu, data tersebut condong ke kanan (*right skewed*).

4. Terdapat sedikit ketidakseimbangan *dataset*, yaitu kasus Benign (yang diwakilkan dengan 1) lebih banyak daripada kasus Malignant (yang diwakilkan dengan 0).
5. Mean sebagian besar fitur jelas lebih besar untuk kasus Malignant dibandingkan dengan kasus Benign (*Groupby*).
6. Sebagian besar fitur memiliki pencilan (*outliers*).
7. Matriks korelasi mengungkapkan bahwa sebagian besar fitur memiliki korelasi yang tinggi. Oleh karena itu, kita dapat menghapus beberapa fitur tertentu selama Seleksi Fitur.

3. PENUTUP

3.1 Kesimpulan

1. Penerapan model *Artificial Neural Network* yang digunakan untuk mengklasifikasikan jenis kanker payudara memberikan hasil yang relatif baik pada proses pengolahan data dengan tingkat akurasi model 93,85%.
2. Model *neural network* yang digunakan dengan memanfaatkan *dataset* Breast Cancer Wisconsin dalam masalah klasifikasi jenis kanker payudara terdiri dari 569 baris dan 32 fitur. Namun, hanya 28 fitur kontinu yang memiliki korelasi yang paling baik dengan variabel target.

3.2 Saran

Untuk mendapatkan hasil yang lebih akurat, perlu diterapkannya *hyperparameter tuning* (proses mencari nilai optimal dari *hyperparameter* suatu model *machine learning* untuk memperbaiki performa model *machine learnin*g), yaitu suatu nilai atau parameter yang diatur secara manual dan berperan di dalam optimasi kinerja suatu model.

4. DAFTAR PUSTAKA

[1] Haykin, Simon, Neural Networks and Learning Machines, 4th ed. Canada: Pearson, 2016.

[2] Chollet, François, Deep Learning with Python, 2nd ed. Manning, 2021.

[3] Nielse, Michael A., Neural Networks and Deep Learning. Determination Press, 2015.