Cancer Detect

by Affa Alfiandy

In [2]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from tensorflow.keras.callbacks import EarlyStopping
```

In [3]:

```
data_cancer = pd.read_csv('C:/Users/affaa/OneDrive/Dokumen/Latihan programming/TF_2_Note
```

In [4]:

```
data_cancer
```

Out[4]:

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mea symmet
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.24
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.18
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.20
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.25
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.18
...	...	...	...	...	...	...	...	...	
564	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.17
565	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.17
566	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.15
567	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.23
568	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.15

569 rows × 31 columns

In [5]:

```
data_cancer.isnull().sum()
```

Out[5]:

```
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
benign_0__mal_1            0
dtype: int64
```

In [6]:

```
data_cancer.info
```

Out[6]:

```
<bound method DataFrame.info of      mean radius  mean texture  mean perim
eter  mean area  mean smoothness  \
0          17.99         10.38          122.80     1001.0          0.11840
1          20.57         17.77          132.90     1326.0          0.08474
2          19.69         21.25          130.00     1203.0          0.10960
3          11.42         20.38           77.58      386.1          0.14250
4          20.29         14.34          135.10     1297.0          0.10030
..           ...           ...             ...        ...              ...
564        21.56         22.39          142.00     1479.0          0.11100
565        20.13         28.25          131.20     1261.0          0.09780
566        16.60         28.08          108.30      858.1          0.08455
567        20.60         29.33          140.10     1265.0          0.11780
568         7.76         24.54           47.92      181.0          0.05263

     mean compactness  mean concavity  mean concave points  mean symmetry
\
0             0.27760         0.30010              0.14710         0.2419
1             0.07864         0.08690              0.07017         0.1812
2             0.15990         0.19740              0.12790         0.2069
3             0.28390         0.24140              0.10520         0.2597
4             0.13280         0.19800              0.10430         0.1809
..                ...             ...                  ...            ...
564           0.11590         0.24390              0.13890         0.1726
565           0.10340         0.14400              0.09791         0.1752
566           0.10230         0.09251              0.05302         0.1590
567           0.27700         0.35140              0.15200         0.2397
568           0.04362         0.00000              0.00000         0.1587

     mean fractal dimension  ...  worst texture  worst perimeter  worst ar
ea  \
0                   0.07871  ...          17.33           184.60      201
9.0
1                   0.05667  ...          23.41           158.80      195
6.0
2                   0.05999  ...          25.53           152.50      170
9.0
3                   0.09744  ...          26.50            98.87       56
7.7
4                   0.05883  ...          16.67           152.20      157
5.0
..                      ...  ...            ...              ...
...
564                 0.05623  ...          26.40           166.10      202
7.0
565                 0.05533  ...          38.25           155.00      173
1.0
566                 0.05648  ...          34.12           126.70      112
4.0
567                 0.07016  ...          39.42           184.60      182
1.0
568                 0.05884  ...          30.37            59.16       26
8.6

     worst smoothness  worst compactness  worst concavity  \
0             0.16220            0.66560           0.7119
```

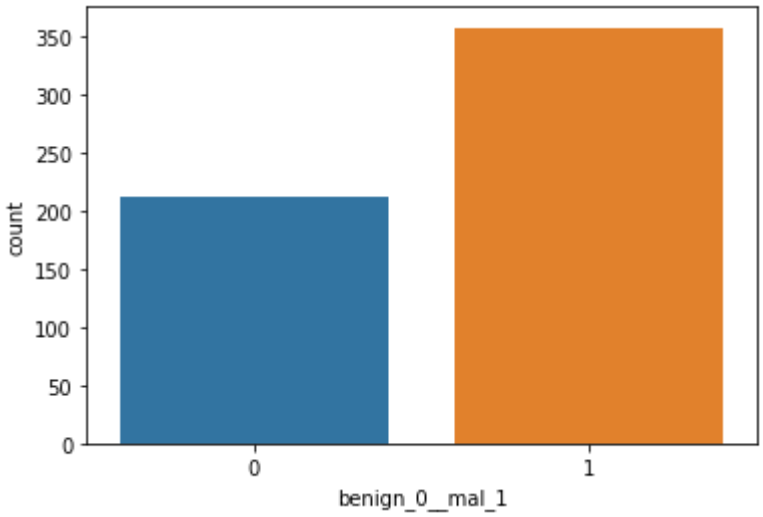
```
1             0.12380           0.18660           0.2416
2             0.14440           0.42450           0.4504
3             0.20980           0.86630           0.6869
4             0.13740           0.20500           0.4000
..                ...               ...              ...
564           0.14100           0.21130           0.4107
565           0.11660           0.19220           0.3215
566           0.11390           0.30940           0.3403
567           0.16500           0.86810           0.9387
568           0.08996           0.06444           0.0000

     worst concave points  worst symmetry  worst fractal dimension  \
0                  0.2654          0.4601                  0.11890
1                  0.1860          0.2750                  0.08902
2                  0.2430          0.3613                  0.08758
3                  0.2575          0.6638                  0.17300
4                  0.1625          0.2364                  0.07678
..                    ...             ...                      ...
564                0.2216          0.2060                  0.07115
565                0.1628          0.2572                  0.06637
566                0.1418          0.2218                  0.07820
567                0.2650          0.4087                  0.12400
568                0.0000          0.2871                  0.07039

     benign_0__mal_1
0                  0
1                  0
2                  0
3                  0
4                  0
..               ...
564                0
565                0
566                0
567                0
568                1

[569 rows x 31 columns]>
```

In [7]:

```
sns.countplot(x='benign_0__mal_1',data=data_cancer)
```

Out[7]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x23438651508>
```



In [8]:

```
data_cancer.corr()['benign_0__mal_1'].sort_values()
```

Out[8]:

```
worst concave points      -0.793566
worst perimeter           -0.782914
mean concave points       -0.776614
worst radius              -0.776454
mean perimeter            -0.742636
worst area                -0.733825
mean radius               -0.730029
mean area                 -0.708984
mean concavity            -0.696360
worst concavity           -0.659610
mean compactness          -0.596534
worst compactness         -0.590998
radius error              -0.567134
perimeter error           -0.556141
area error                -0.548236
worst texture             -0.456903
worst smoothness          -0.421465
worst symmetry            -0.416294
mean texture              -0.415185
concave points error      -0.408042
mean smoothness           -0.358560
mean symmetry             -0.330499
worst fractal dimension   -0.323872
compactness error         -0.292999
concavity error           -0.253730
fractal dimension error   -0.077972
symmetry error             0.006522
texture error              0.008303
mean fractal dimension     0.012838
smoothness error           0.067016
benign_0__mal_1            1.000000
Name: benign_0__mal_1, dtype: float64
```

In [9]:

```
X = data_cancer.drop('benign_0__mal_1',axis=1).values
y = data_cancer['benign_0__mal_1'].values
```

In [10]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=1
```

In [11]:

```
scaller = MinMaxScaler()
```

In [12]:

```
X_train = scaller.fit_transform(X_train)
X_test = scaller.transform(X_test)
```

In [13]:

```
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense,Dropout
```

In [14]:

```
X_train.shape
```

Out[14]:

```
(426, 30)
```

In [15]:

```
model = Sequential()
```

In [16]:

```
model.add(Dense(30,activation = 'relu'))
model.add(Dense(15,activation = 'relu'))
#Because Binary Classsification Use sigmoid
model.add(Dense(1,activation = 'sigmoid'))

model.compile(optimizer='adam',loss='binary_crossentropy')
```

In [17]:

```
model.fit(x=X_train,y=y_train,epochs=600,validation_data=(X_test,y_test))
```

```
Epoch 596/600
426/426 [==============================] - 0s 166us/sample - loss: 0.002
4 - val_loss: 0.1702
Epoch 597/600
426/426 [==============================] - 0s 165us/sample - loss: 0.002
0 - val_loss: 0.1849
Epoch 598/600
426/426 [==============================] - 0s 168us/sample - loss: 0.002
2 - val_loss: 0.1873
Epoch 599/600
426/426 [==============================] - 0s 180us/sample - loss: 0.002
3 - val_loss: 0.1904
Epoch 600/600
426/426 [==============================] - 0s 179us/sample - loss: 0.002
5 - val_loss: 0.1802
```

Out[17]:

```
<tensorflow.python.keras.callbacks.History at 0x2343a98d608>
```

In [18]:

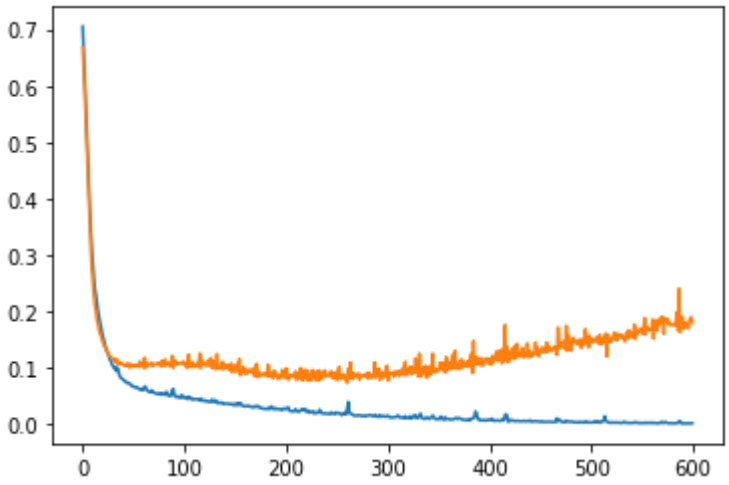
```
losses = pd.DataFrame(model.history.history)
```

In [19]:

```
plt.plot(losses)
```

Out[19]:

```
[<matplotlib.lines.Line2D at 0x2343b04ce08>,
 <matplotlib.lines.Line2D at 0x2343c046248>]
```



In [20]:

```
model.add(Dense(30,activation = 'relu'))
model.add(Dense(15,activation = 'relu'))
#Because Binary Classsification Use sigmoid
model.add(Dense(1,activation = 'sigmoid'))

model.compile(optimizer='adam',loss='binary_crossentropy')
```

In [21]:

```
earlystop = EarlyStopping(monitor ='val_loss',mode='min',verbose=1,patience=25)
```

In [22]:

```
model.fit(x=X_train,y=y_train,epochs=600,validation_data=(X_test,y_test),callbacks=[ear
```

```
Epoch 30/600
426/426 [==============================] - 0s 216us/sample - loss: 0.038
2 - val_loss: 0.1612
Epoch 31/600
426/426 [==============================] - 0s 197us/sample - loss: 0.034
2 - val_loss: 0.1656
Epoch 32/600
426/426 [==============================] - 0s 194us/sample - loss: 0.030
7 - val_loss: 0.1632
Epoch 33/600
426/426 [==============================] - 0s 202us/sample - loss: 0.027
6 - val_loss: 0.1637
Epoch 34/600
426/426 [==============================] - 0s 234us/sample - loss: 0.024
9 - val_loss: 0.1675
Epoch 35/600
426/426 [==============================] - 0s 190us/sample - loss: 0.022
5 - val_loss: 0.1657
Epoch 36/600
426/426 [==============================] - 0s 222us/sample - loss: 0.020
```

In [23]:

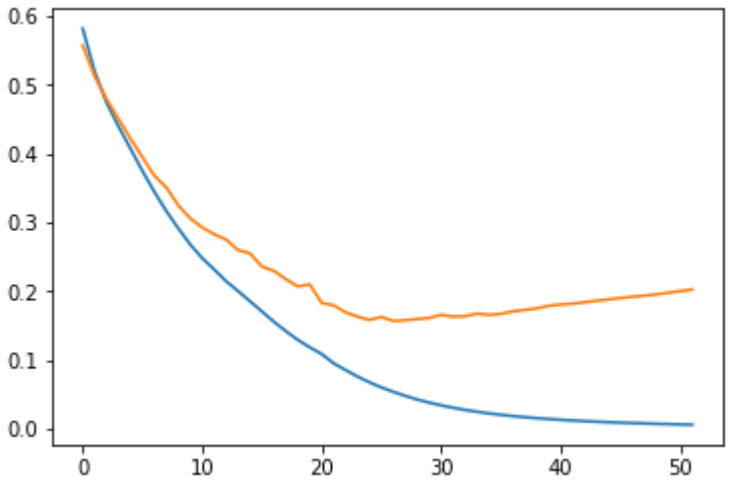
```
model_loss = pd.DataFrame(model.history.history)
```

In [24]:

```
plt.plot(model_loss)
```

Out[24]:

```
[<matplotlib.lines.Line2D at 0x2343c3b9cc8>,
 <matplotlib.lines.Line2D at 0x2343c3b9e88>]
```



In [25]:

```
from tensorflow.keras.layers import Dropout
```

In [26]:

```
model.add(Dense(30,activation = 'relu'))
model.add(Dropout(0.5))
model.add(Dense(15,activation = 'relu'))
model.add(Dropout(0.5))
#Because Binary Classsification Use sigmoid
model.add(Dense(1,activation = 'sigmoid'))

model.compile(optimizer='adam',loss='binary_crossentropy')
```

In [27]:

```
model.fit(x=X_train,y=y_train,epochs=600,validation_data=(X_test,y_test),callbacks=[ear
```

```
Epoch 28/600
426/426 [==============================] - 0s 201us/sample - loss: 0.278
5 - val_loss: 0.3254
Epoch 29/600
426/426 [==============================] - 0s 176us/sample - loss: 0.264
4 - val_loss: 0.3246
Epoch 30/600
426/426 [==============================] - 0s 188us/sample - loss: 0.269
2 - val_loss: 0.3242
Epoch 31/600
426/426 [==============================] - 0s 184us/sample - loss: 0.270
6 - val_loss: 0.3244
Epoch 32/600
426/426 [==============================] - 0s 178us/sample - loss: 0.259
9 - val_loss: 0.3242
Epoch 33/600
426/426 [==============================] - 0s 178us/sample - loss: 0.263
6 - val_loss: 0.3239
Epoch 34/600
426/426 [==============================] - 0s 202us/sample - loss: 0.252
```

In [28]:

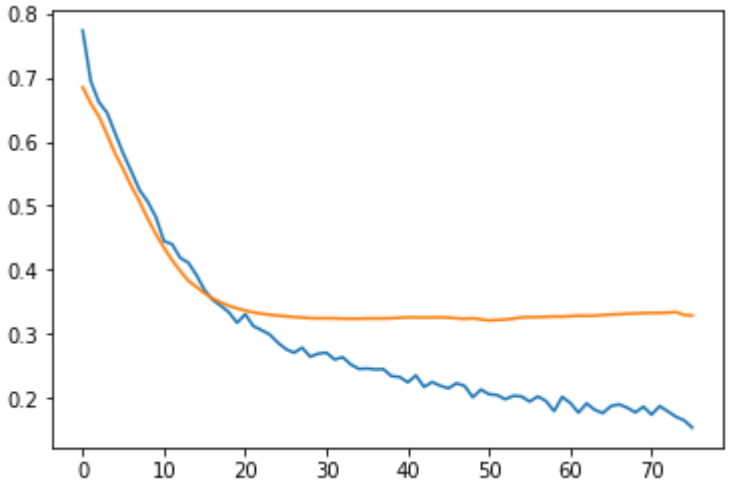
```
model_loss = pd.DataFrame(model.history.history)
```

In [29]:

```
plt.plot(model_loss)
```

Out[29]:

```
[<matplotlib.lines.Line2D at 0x2343c8ad8c8>,
 <matplotlib.lines.Line2D at 0x2343c8b5748>]
```



In [30]:

```
predictions = model.predict(X_test)
```

In [31]:

```
from sklearn.metrics import classification_report,confusion_matrix
```

In [33]:

```
testdata = data_cancer.drop('benign_0__mal_1',axis=1).iloc[0]
```

In [34]:

```
model.predict(testdata)
```

```
WARNING:tensorflow:Falling back from v2 loop because of error: Failed to f
ind data adapter that can handle input: <class 'pandas.core.series.Serie
s'>, <class 'NoneType'>

---------------------------------------------------------------------------
-
InvalidArgumentError                      Traceback (most recent call las
t)
<ipython-input-34-5337eeb452c8> in <module>
----> 1 model.predict(testdata)

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\py
thon\keras\engine\training.py in predict(self, x, batch_size, verbose, ste
ps, callbacks, max_queue_size, workers, use_multiprocessing)
    907         max_queue_size=max_queue_size,
    908         workers=workers,
--> 909         use_multiprocessing=use_multiprocessing)
    910
    911   def reset_metrics(self):

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\py
thon\keras\engine\training_arrays.py in predict(self, model, x, batch_siz
e, verbose, steps, callbacks, **kwargs)
    720         verbose=verbose,
    721         steps=steps,
--> 722         callbacks=callbacks)

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\py
thon\keras\engine\training_arrays.py in model_iteration(model, inputs, tar
gets, sample_weights, batch_size, epochs, verbose, callbacks, val_inputs,
 val_targets, val_sample_weights, shuffle, initial_epoch, steps_per_epoch,
validation_steps, validation_freq, mode, validation_in_fit, prepared_feed_
values_from_dataset, steps_name, **kwargs)
    391
    392         # Get outputs.
--> 393         batch_outs = f(ins_batch)
    394         if not isinstance(batch_outs, list):
    395           batch_outs = [batch_outs]

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\py
thon\keras\backend.py in __call__(self, inputs)
   3738         value = math_ops.cast(value, tensor.dtype)
   3739       converted_inputs.append(value)
-> 3740     outputs = self._graph_fn(*converted_inputs)
   3741
   3742     # EagerTensor.numpy() will often make a copy to ensure memory
 safety.

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\py
thon\eager\function.py in __call__(self, *args, **kwargs)
   1079       TypeError: For invalid positional/keyword argument combinati
ons.
   1080     """
-> 1081     return self._call_impl(args, kwargs)
   1082
   1083   def _call_impl(self, args, kwargs, cancellation_manager=None):
```

```
c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\python\eager\function.py in _call_impl(self, args, kwargs, cancellation_manager)
   1119       raise TypeError("Keyword arguments {} unknown. Expected {}.".format(
   1120           list(kwargs.keys()), list(self._arg_keywords)))
-> 1121     return self._call_flat(args, self.captured_inputs, cancellation_manager)
   1122
   1123   def _filtered_call(self, args, kwargs):

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\python\eager\function.py in _call_flat(self, args, captured_inputs, cancellation_manager)
   1222     if executing_eagerly:
   1223       flat_outputs = forward_function.call(
-> 1224           ctx, args, cancellation_manager=cancellation_manager)
   1225     else:
   1226       gradient_name = self._delayed_rewrite_functions.register()

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\python\eager\function.py in call(self, ctx, args, cancellation_manager)
    509               inputs=args,
    510               attrs=("executor_type", executor_type, "config_proto", config),
--> 511               ctx=ctx)
    512         else:
    513           outputs = execute.execute_with_cancellation(

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\python\eager\execute.py in quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
     65     else:
     66       message = e.message
---> 67     six.raise_from(core._status_to_exception(e.code, message), None)
     68   except TypeError as e:
     69     keras_symbolic_tensors = [

c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\six.py in raise_from(value, from_value)

InvalidArgumentError:  Matrix size-incompatible: In[0]: [30,1], In[1]: [30,30]
         [[node sequential/dense/Relu (defined at c:\users\affaa\anaconda3\envs\mytfenv\lib\site-packages\tensorflow_core\python\framework\ops.py:1751) ]] [Op:__inference_keras_scratch_graph_33731]

Function call stack:
keras_scratch_graph
```

In [ ]: