In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
data = pd.read_csv('dataANN.csv')
```

In [3]:
```
data.shape
```

Out[3]:
```
(70000, 14)
```

In [4]:
```
data
```

Out[4]:

	id	age_days	age_year	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smok
0	0	18393	50.391781	2	168	62.0	110	80	1	1	
1	1	20228	55.419178	1	156	85.0	140	90	3	1	
2	2	18857	51.663014	1	165	64.0	130	70	3	1	
3	3	17623	48.282192	2	169	82.0	150	100	1	1	
4	4	17474	47.873973	1	156	56.0	100	60	1	1	
...	...	...	...	...	...	...	...	...	...	...	.
69995	99993	19240	52.712329	2	168	76.0	120	80	1	1	
69996	99995	22601	61.920548	1	158	126.0	140	90	2	2	
69997	99996	19066	52.235616	2	183	105.0	180	90	3	1	
69998	99998	22431	61.454795	1	163	72.0	135	80	1	2	
69999	99999	20540	56.273973	1	170	72.0	120	80	2	1	

70000 rows × 14 columns

In [5]:
```
X = data.iloc[:, 1:13].values
y = data.iloc[:, 13].values
```

In [6]:
```
X
```

Out[6]:
```
array([[1.83930000e+04, 5.03917808e+01, 2.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 1.00000000e+00],
       [2.02280000e+04, 5.54191781e+01, 1.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 1.00000000e+00],
       [1.88570000e+04, 5.16630137e+01, 1.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       ...,
       [1.90660000e+04, 5.22356164e+01, 2.00000000e+00, ...,
        0.00000000e+00, 1.00000000e+00, 0.00000000e+00],
       [2.24310000e+04, 6.14547945e+01, 1.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 0.00000000e+00],
       [2.05400000e+04, 5.62739726e+01, 1.00000000e+00, ...,
        0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])
```

In [7]:
```
y
```

Out[7]:
```
array([0, 1, 1, ..., 1, 1, 0], dtype=int64)
```

In [8]:
```
from sklearn.preprocessing import LabelEncoder
labelencoder_X_1 = LabelEncoder()
y = labelencoder_X_1.fit_transform(y)
```

In [9]:
```
y
```

Out[9]:
```
array([0, 1, 1, ..., 1, 1, 0], dtype=int64)
```

In [11]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, rando
```

In [12]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [13]:
```
X_train
```

Out[13]:
```
array([[ 1.5985301 ,  1.5985301 , -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [ 0.01419727,  0.01419727, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [ 0.56581583,  0.56581583, -0.73535519, ..., -0.31159702,
        -0.23933665, -2.02362986],
       ...,
       [ 1.17782468,  1.17782468,  1.35988704, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [-0.46933025, -0.46933025, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [ 0.38302379,  0.38302379,  1.35988704, ..., -0.31159702,
        -0.23933665,  0.49416152]])
```

In [14]:
```
X_test
```

Out[14]:
```
array([[-1.72779877, -1.72779877, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [ 0.13619373,  0.13619373, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       [ 0.43936103,  0.43936103, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152],
       ...,
       [ 1.21713916,  1.21713916,  1.35988704, ..., -0.31159702,
        -0.23933665, -2.02362986],
       [-1.96652275, -1.96652275,  1.35988704, ...,  3.20927329,
        -0.23933665,  0.49416152],
       [-0.22209489, -0.22209489, -0.73535519, ..., -0.31159702,
        -0.23933665,  0.49416152]])
```

In [15]:
```
X_train.shape
```

Out[15]:
```
(56000, 12)
```

In [16]:
```
X_test.shape
```

Out[16]:
```
(14000, 12)
```

In [17]:
```
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras import models
from keras import layers
```

In [18]:
```
classifier = models.Sequential()
```

In [19]:
```
classifier.add(layers.Dense(12, activation='relu' ,use_bias=True,name = 'input'))
classifier.add(layers.Dropout(rate=0.1))
```

In [20]:
```
classifier.add(layers.Dense(12, activation='relu',use_bias=True,name = 'hidden1'))
classifier.add(layers.Dropout(rate=0.1))
```

In [21]:
```
classifier.add(layers.Dense(1,  activation='sigmoid',use_bias=True,name = 'output
```

In [29]:
```
classifier.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accura
```

In [30]:
```
history = classifier.fit(X_train, y_train, batch_size=100, epochs=100,validation_
```

```
acy: 0.7341 - val_loss: 0.5624 - val_accuracy: 0.7223
Epoch 79/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5461 - accur
acy: 0.7334 - val_loss: 0.5613 - val_accuracy: 0.7223
Epoch 80/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5460 - accur
acy: 0.7337 - val_loss: 0.5618 - val_accuracy: 0.7196
Epoch 81/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5459 - accur
acy: 0.7328 - val_loss: 0.5627 - val_accuracy: 0.7223
Epoch 82/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5456 - accur
acy: 0.7342 - val_loss: 0.5620 - val_accuracy: 0.7170
Epoch 83/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5453 - accur
acy: 0.7340 - val_loss: 0.5607 - val_accuracy: 0.7214
Epoch 84/100
549/549 [==============================] - 1s 1ms/step - loss: 0.5457 - accur
acy: 0.7339 - val_loss: 0.5619 - val_accuracy: 0.7250
Epoch 85/100
```

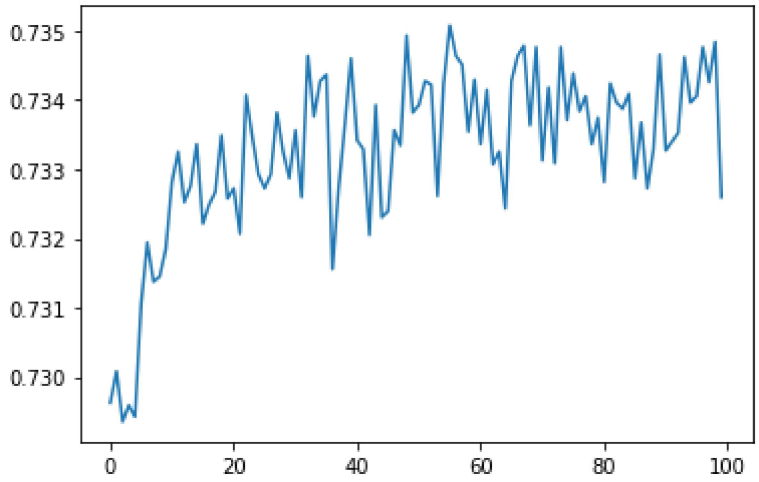
In [31]:
```
classifier.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input (Dense)                (None, 12)                156
_________________________________________________________________
dropout (Dropout)            (None, 12)                0
_________________________________________________________________
hidden1 (Dense)              (None, 12)                156
_________________________________________________________________
dropout_1 (Dropout)          (None, 12)                0
_________________________________________________________________
output (Dense)               (None, 1)                 13
=================================================================
Total params: 325
Trainable params: 325
Non-trainable params: 0
_________________________________________________________________
```

In [32]:
```
classifier.weights
```

Out[32]:
```
[<tf.Variable 'input/kernel:0' shape=(12, 12) dtype=float32, numpy=
 array([[ 3.83899808e-01,  1.69875354e-01,  1.31449610e-01,
         -5.87779820e-01,  2.34277979e-01, -1.80614993e-01,
         -4.30697612e-02, -4.36798662e-01,  3.78667116e-01,
          2.19583318e-01,  1.91744030e-01, -1.89905941e-01],
        [ 5.99754393e-01, -8.85015503e-02,  3.74571681e-01,
          2.10306183e-01, -2.19680190e-01,  1.69994906e-01,
          1.52537206e-04,  4.21699017e-01,  2.50457585e-01,
         -2.23337755e-01, -2.62025326e-01, -1.05706677e-01],
        [-1.71571448e-01,  1.20864145e-03,  2.18342021e-01,
         -2.10637208e-02, -7.51293777e-03,  1.07831676e-02,
          3.61952223e-02, -2.58291513e-02,  2.91432198e-02,
          7.97408447e-02,  1.52941672e-02, -9.51172411e-03],
        [ 9.13591310e-02,  1.28161669e-01, -1.54117823e-01,
         -5.98947778e-02, -1.02040470e-02, -1.29963206e-02,
          4.14380021e-02, -1.83649827e-02, -2.07286794e-02,
          6.34928867e-02,  1.15355207e-02,  2.87403492e-03],
        [ 1.41857406e-02, -3.10321122e-01,  5.23836732e-01,
         -2.32308537e-01,  4.24790680e-02,  4.41529416e-02,
```

In [33]:
```
plt.plot(history.history['accuracy'])
plt.show()
```



In [34]:
```
y_pred = classifier.predict(X_test)
y_pred = (y_pred > 0.5)
```

In [35]:
```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[35]:
```
array([[5428, 1641],
       [2085, 4846]], dtype=int64)
```

In [36]:
```
print("Our accuracy is {}%".format(((cm[0][0] + cm[1][1])/14000)*100))
```

```
Our accuracy is 73.38571428571429%
```