

1. Mô tả dữ liệu bài toán:

- **Bộ dữ liệu dự báo sinh non dùng cho Học viên Sau đại học của ĐHYD TPHCM (môn thống kê y học và tin học ứng dụng dùng cho Chuyên khoa cấp II):** tên file: **ivf2_(2).csv (641 thể hiện)**

Hộp thư đến (1) - tongint21028@

Colab Notebooks - Google Drive

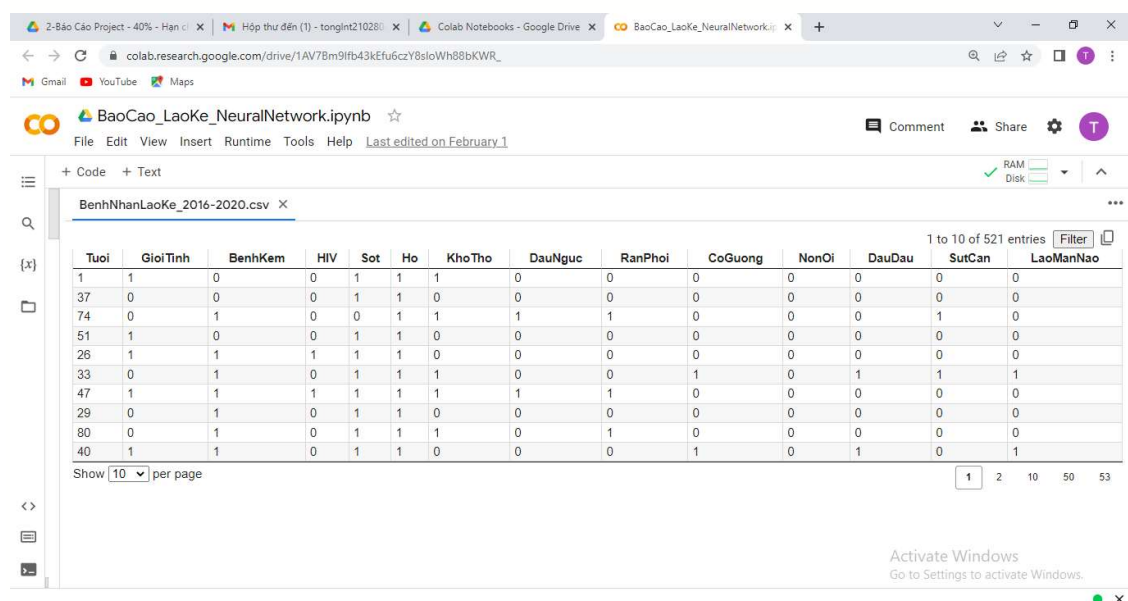
BaoCao_ivf2_NeuralNetwork.ipynb

BaoCao_LaoKe_XetNghiemAFB

<

Gồm các thuộc tính đặc trưng: **tuoime**, **tang_ha**(0: không tăng; 1: có tăng); **tuoithai**(đơn vị tính là tuần, cách tính theo kỹ thuật ivf); **ngheghiep**(chia làm 3 nhóm: **tudo**, **congphan** và **vienchuc**); **sinhnon**(1: sinh non; 0: không sinh non)

- **Bộ dữ liệu chẩn đoán bệnh lao màng não:** Dữ liệu được thu thập từ năm 2016 đến năm 2020 tại Bệnh viện LPPNT; tên file: **BenhNhanLaoKe_2016-2020.csv (521 thể hiện)**



| Tuoi | GiớiTinh | BenhKem | HIV | Sot | Ho | KhoTho | DauNguc | RanPhoi | CoGuong | NonOi | DauDau | SutCan | LaoManNao |
|------|----------|---------|-----|-----|----|--------|---------|---------|---------|-------|--------|--------|-----------|
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 74 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 51 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 26 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 33 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| 47 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 29 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 80 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 40 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |

Dựa vào các đặc trưng là: **Tuoi**, **GioiTinh** (1: nam; 0: nữ), **BenhKem** (1: có; 0: không có), **HIV** (1: có; 0: không có) và các triệu chứng lâm sàng **Sot** (1: có; 0: không có), **Ho** (1: có; 0: không có), **KhoTho** (1: có; 0: không có), **DauNguc** (1: có; 0: không có), **RanPhoi** (1: có; 0: không có), **CoGuong** (1: có; 0: không có), **NonOi** (1: có; 0: không có), **DauDau** (1: có; 0: không có), **SutCan** (1: có; 0: không có)

2. Công cụ khai phá dữ liệu: Viết đoạn code chương trình ứng dụng kỹ thuật DeepLearning sử dụng mạng nơ-ron nhân tạo (ANN: **Artificial Neural Network**) trong dự báo sinh non và chẩn đoán bệnh lao màng não (**Chương trình chạy trên GoogleColab**)

3. Đặc tả bài toán:

- **Dự báo khả năng sinh non của các bà mẹ:**

Dựa vào các đặc trưng: **tuoime**, **tuoithai**, **tang_ha**, **ngheNghiep** (công chức, viên chức hoặc nghề tự do)

```
# Import required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn
from sklearn.neural_network import MLPClassifier
from sklearn.neural_network import MLPRegressor

# Import necessary modules
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.metrics import r2_score

df = pd.read_csv("/content/drive/MyDrive/ColabFolder2022/ivf2_(2).csv")
print(df.shape)
df.describe().transpose()
```

Kết quả chạy đoạn code trên:

(641, 7) // 641 mẫu, 7 thuộc tính đặc trưng

| | count | mean | std | min | 25% | 50% | 75% | max |
|----------|-------|-----------|----------|-------|-------|-------|-------|-------|
| tuoime | 641.0 | 33.971919 | 3.870460 | 23.00 | 31.00 | 34.00 | 37.00 | 43.00 |
| tang_ha | 641.0 | 0.138846 | 0.346055 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 |
| tuoithai | 641.0 | 38.687254 | 2.329931 | 24.69 | 38.01 | 39.15 | 40.15 | 42.35 |
| tudo | 641.0 | 0.162246 | 0.368965 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 |
| congnhan | 641.0 | 0.371295 | 0.483528 | 0.00 | 0.00 | 0.00 | 1.00 | 1.00 |
| vienchuc | 641.0 | 0.466459 | 0.499263 | 0.00 | 0.00 | 0.00 | 1.00 | 1.00 |
| sinhnon | 641.0 | 0.138846 | 0.346055 | 0.00 | 0.00 | 0.00 | 0.00 | 1.00 |

target_column = ['sinhnon'] # **Dán nhãn phân loại 1: sinh non; 0: không sinh non**

predictors = list(set(list(df.columns))-set(target_column))

df[predictors] = df[predictors]/df[predictors].max()

df.describe().transpose()

df.head()

Kết quả chạy đoạn code trên:

| | tuoime | tang_ha | tuoithai | tudo | congnhan | vienchuc | sinhnon |
|---|----------|---------|----------|------|----------|----------|---------|
| 0 | 0.767442 | 0.0 | 0.891145 | 0.0 | 0.0 | 1.0 | 0 |
| 1 | 0.790698 | 0.0 | 0.924439 | 0.0 | 1.0 | 0.0 | 0 |
| 2 | 0.790698 | 0.0 | 0.843447 | 0.0 | 1.0 | 0.0 | 1 |
| 3 | 0.697674 | 0.0 | 0.927745 | 0.0 | 0.0 | 1.0 | 0 |
| 4 | 0.813953 | 0.0 | 0.906257 | 0.0 | 1.0 | 0.0 | 0 |

X = df[predictors].values

y = df[target_column].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=16)

#train_test_split là kỹ thuật đánh giá hiệu suất của thuật toán học máy có giám sát

#dữ liệu: 70% huấn luyện; 30% test

print(X_train.shape); print(X_test.shape)

Kết quả chạy đoạn code trên:

(448, 6)

(193, 6)

```

from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_train, predict_train))
print(classification_report(y_train, predict_train))
print(confusion_matrix(y_test, predict_test))
print(classification_report(y_test, predict_test))

```

Kết quả chạy đoạn code trên sau khi huấn luyện mô hình:

```

[[169  0]
 [  3 21]]
      precision    recall  f1-score   support

     0       0.98      1.00      0.99      169
     1       1.00      0.88      0.93       24

 accuracy          0.98      193
 macro avg       0.99      0.94      0.96      193
 weighted avg    0.98      0.98      0.98      193

```

Để giải thích bảng confusion matrix trên và các giá trị dự báo trong bảng, ta cần nắm lý thuyết sau đây trong lĩnh vực y học (dịch tễ học)

| | Test (-) | Test (+) | Quan sát |
|----------|---------------|---------------|--------------|
| Bệnh (-) | a (TN) | b (FP) | $n1 = a + b$ |
| Bệnh (+) | c (FN) | d (TP) | $n2 = c + d$ |
| | $a + c$ | $b + d$ | |

TN: True Negative

TP: True Positive

FN: False Negative

FP: False Positive

Mục đích confusion_matrix: xem xét sự chính xác của xét nghiệm mới và khả năng chẩn đoán bệnh dựa vào kết quả xét nghiệm mới như thế nào?

recall: độ chính xác của xét nghiệm

$$\text{Độ nhạy} = P(T+|B+) = d/n2$$

$$\text{Độ đặc hiệu} = P(T-|B-) = a/n1$$

precision (giá trị tiên đoán): xem xét khả năng bệnh hay không bệnh khi kết quả Xét nghiệm đã có dương tính hay âm tính

$$\text{Tiên đoán dương} = P(B+|T+) = d/(b + d)$$

$$\text{Tiên đoán âm} = P(B-|T-) = a/(a + c)$$

Như vậy: Trở lại giải thích kết quả đoạn chương trình dự báo trên qua **bảng confusion matrix**

```
[[169  0]
 [ 3 21]]
      precision  recall    f1-score  support
0      0.98      1.00      0.99      169
1      1.00      0.88      0.93      24

accuracy              0.98      193
macro avg      0.99      0.94      0.96      193
weighted avg   0.98      0.98      0.98      193
```

| | Test (-) | Test (+) | Quan sát |
|----------|----------|----------|----------|
| Bệnh (-) | 169 | 0 | 169 |
| Bệnh (+) | 3 | 21 | 24 |
| | 172 | 21 | |

recall:

$$\text{Độ nhạy} = P(T+|B+) = 21/24 = 0.88 \text{ (88\%)}$$

$$\text{Độ đặc hiệu} = P(T-|B-) = 169/169 = 1 \text{ (100\%)}$$

Precision:

Tiên đoán dương = $P(B+|T+) = 21/21 = 1$ (100%)

Tiên đoán âm = $P(B-|T-) = 169/172 = 0.98$ (98%)

Nhận xét kết quả trên:

Độ chính xác của mô hình accuracy khá tốt: 0.98 (193 dữ liệu test)

$$accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

Đo lường độ chính xác của tất cả các dự đoán (tích cực và tiêu cực).

Độ chính xác càng cao càng tốt.

$$accuracy = (21 + 169)/(21 + 169 + 3 + 0) = 0.98$$

accuracy gọi là độ chính xác của mô hình (nhưng chưa nói lên được điều gì mà phải giải thích thêm dựa vào giá trị **recall** và **precision**)

recall:

Độ nhạy = 0.88 (88%)

(Nếu như người mẹ sinh non, thì với những đặc trưng được đưa vào mô hình đã huấn luyện kiểm tra thì kết quả mô hình dự báo là sinh non đạt xác suất 88%)

Độ đặc hiệu = 1 (100%)

(Nếu như người mẹ không sinh non, thì với những đặc trưng được đưa vào mô hình đã huấn luyện kiểm tra thì kết quả mô hình dự báo là không sinh non đạt xác suất 100%)

Precision:

Tiên đoán dương = 1 (100%)

(Nếu như đưa bộ dữ liệu mới của người mẹ vào test mà mô hình dự báo là có sinh non thì khả năng người mẹ sinh non là chắc chắn 100%).

Tiên đoán âm = 0.98 (98%)

(Nếu như đưa bộ dữ liệu mới của người mẹ vào test mà mô hình dự báo là không sinh non thì khả năng người mẹ không sinh non là chắc chắn 98% cũng khá cao).

$$f1 - score = \frac{2*recall*precision}{recall+precision}$$

Còn gọi là thước đo giữa hai giá trị precision và recall bằng cách tìm giá trị trung bình hài hòa của hai giá trị

Save model và loaded model

```
import pickle

# save the model to disk
filename = 'ivf2_finalized_model.sav'
pickle.dump(mlp, open(filename, 'wb'))

# load the model from disk
loaded_model = pickle.load(open(filename, 'rb'))
result = loaded_model.score(X_test, y_test)
print(result)
```

Kết quả test model đạt 0.98 %

```
0.9844559585492227
```

● Chẩn đoán bệnh lao màng não:

Dựa vào các đặc trưng là: **Tuoi**, **GioiTinh**, **BenhKem**, **HIV** và các triệu chứng lâm sàng: **Sot**, **Ho**, **KhoTho**, **DauNguc**, **RanPhoi**, **CoGuong**, **NonOi**, **DauDau**, **SutCan**

| Tuoi | GioiTinh | Benh Kem | HIV | Sot | Ho | KhoTho | DauNguc | RanPhoi | CoGuong | NonOi | DauDau | SutCan |
|------|----------|-------------|-----|-----|----|--------|---------|---------|---------|-------|--------|--------|
|------|----------|-------------|-----|-----|----|--------|---------|---------|---------|-------|--------|--------|

Import required libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import sklearn
from sklearn.neural_network import MLPClassifier
from sklearn.neural_network import MLPRegressor
```

Import necessary modules

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
from math import sqrt
from sklearn.metrics import r2_score
```

```
df = pd.read_csv("/content/drive/MyDrive/ColabFolder2022/BenhNhanLaoKe_2016-2020_(1).csv")
print(df.shape)
df.describe().transpose()
```

Kết quả chạy đoạn chương trình trên

(521, 14) # 521 mẫu và 14 đặc trưng

| | count | mean | std | min | 25% | 50% | 75% | max |
|-----------|-------|-----------|-----------|-----|------|------|------|------|
| Tuoi | 521.0 | 42.113244 | 23.794566 | 1.0 | 26.0 | 39.0 | 62.0 | 96.0 |
| GioiTinh | 521.0 | 0.598848 | 0.490603 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| BenhKem | 521.0 | 0.702495 | 0.457600 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 |
| HIV | 521.0 | 0.143954 | 0.351380 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| Sot | 521.0 | 0.859885 | 0.347440 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| Ho | 521.0 | 0.779271 | 0.415137 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| KhoTho | 521.0 | 0.399232 | 0.490211 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 |
| DauNguc | 521.0 | 0.120921 | 0.326349 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| RanPhoi | 521.0 | 0.245681 | 0.430904 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| CoGuong | 521.0 | 0.030710 | 0.172697 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| NonOi | 521.0 | 0.026871 | 0.161863 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| DauDau | 521.0 | 0.117083 | 0.321828 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| SutCan | 521.0 | 0.232246 | 0.422671 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |
| LaoManNao | 521.0 | 0.182342 | 0.386497 | 0.0 | 0.0 | 0.0 | 0.0 | 1.0 |

```
target_column = ['LaoManNao'] # Dán nhãn
predictors = list(set(list(df.columns))-set(target_column))
df[predictors] = df[predictors]/df[predictors].max()
df.describe().transpose()
df.head()
```


| | Tuoi | GioiTinh | BenhKem | HIV | Sot | Ho | KhoTh | DauNg | RanPh | CoGuo | NonOi | DauDa | SutCan | LaoMa |
|---|--------------|----------|---------|-----|-----|-----|-------|-------|-------|-------|-------|-------|--------|-------|
| | | h | em | | | | o | uc | oi | ng | | u | | nNao |
| 0 | 0.0104 17 | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0 |
| 1 | 0.3854 17 | 0.0 | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0 |
| 2 | 0.7708 33 | 0.0 | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 1.0 | 0 |
| 3 | 0.5312 50 | 1.0 | 0.0 | 0.0 | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0 |
| 4 | 0.2708 33 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0 |

```
X = df[predictors].values
```

```
y = df[target_column].values
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=16)
```

#train_test_split là kỹ thuật đánh giá hiệu suất của thuật toán học máy có giám sát

#dữ liệu: 70% huấn luyện; 30% test

```
print(X_train.shape); print(X_test.shape)
```

Kết quả chạy đoạn chương trình trên:

```
(364, 13)
```

```
(157, 13)
```

```
from sklearn.neural_network import MLPClassifier
```

```
mlp = MLPClassifier(hidden_layer_sizes=(10,10,10), activation='relu', solver='adam', max_iter=1000)
```

MLPClassifier: Thuật toán học máy có giám sát thuộc mạng Neural nhân tạo

```
mlp.fit(X_train,y_train)
```

```
predict_train = mlp.predict(X_train)
```

```
predict_test = mlp.predict(X_test)
```

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

```
print(confusion_matrix(y_train,predict_train))
```

```
print(classification_report(y_train,predict_train))
```

```
print(confusion_matrix(y_test,predict_test))
print(classification_report(y_test,predict_test))
```

Kết quả chạy đoạn chương trình trên:

```
[[125  4]
 [ 13 15]]
      precision  recall f1-score  support
0      0.91      0.97      0.94         129
1      0.79      0.54      0.64          28

accuracy              0.89      157
macro avg      0.85      0.75      0.79      157
weighted avg    0.89      0.89      0.88      157
```

Nhận xét kết quả bảng confusion matrix:

accuracy = 0.89 (89% chính xác trên 157 mẫu test)

recall:

Độ nhạy = 0.54 (54%)

(Nếu như người bị bệnh lao màng não, thì với những đặc trưng được đưa vào mô hình đã huấn luyện kiểm tra thì kết quả mô hình dự báo là lao màng não đạt xác suất 54%) → **rất nguy hiểm (chẩn đoán bệnh sai, khi đó bệnh nhân về nhà rất nguy hiểm đến tính mạng vì không được điều trị kịp thời)**. Như vậy cần bổ sung thêm các đặc trưng khác trong các kết quả xét nghiệm cận lâm sàng (*Acid Fast Bacillus*, *MGIT*, *GENXPERT*, *PNEUCYST*) đưa vào mô hình huấn luyện thêm để tăng xác suất dự báo chính xác. **Accuracy = 0.89** cũng chưa nói lên mô hình đạt độ tin cậy cao vì như trường hợp này thì một người có bệnh nhưng tỷ lệ mô hình chẩn đoán sai khá cao (**46%**).

Độ đặc hiệu = 0.97 (97%)

(Nếu như người không bị bệnh lao màng não, thì với những đặc trưng được đưa vào mô hình đã huấn luyện kiểm tra thì kết quả mô hình dự báo là không lao màng não đạt xác suất 97%). (**người được mô hình dự báo yên tâm**)

precision:

Tiên đoán dương = 0.79 (79%)

(Nếu như đưa bộ dữ liệu mới của người chưa biết mình có bệnh hay không có bệnh vào test mà mô hình dự báo là có lao màng não thì khả năng bị lao màng não thực sự là 79%). Vậy cũng còn 21% mô hình chẩn đoán sai (nên cũng còn hy vọng là mình không bị bệnh)

Tiên đoán âm = 0.91 (91%)

(Nếu như đưa bộ dữ liệu mới của người chưa biết mình có bệnh hay không có bệnh vào test mà mô hình dự báo là không lao màng não thì khả năng không bị lao màng não thực sự là 91%). (người được mô hình dự báo yên tâm)

```
import pickle
```

```
# save the model to disk
```

```
filename = 'LaoKe_finalized_model.sav'
```

```
pickle.dump(mlp, open(filename, 'wb'))
```

```
# load the model from disk
```

```
loaded_model = pickle.load(open(filename, 'rb'))
```

```
result = loaded_model.score(X_test, y_test)
```

```
print(result)
```

Kết quả chạy đoạn chương trình trên: độ chính xác của mô hình cũng tương đối tốt

0.89171974522293

Tài liệu tham khảo

1. Tài liệu môn học Data Mining của thầy TS. Nguyễn Tiến Đạt.
2. Nguyễn Thanh Tuấn, “Deep Learning Cơ bản” tái bản 2, 2020.
3. Machine Learning cơ bản của Vũ Hữu Tiệp.
4. Tổng quan về Artificial Neural Network. (n.d.). Viblo. Retrieved May 10, 2022, from <https://viblo.asia/p/tong-quan-ve-artificial-neural-network-1VgZvwYrlAw>.
5. Gavin Hackeling, “Mastering Machine Learning with Scikit-learn”, 2014.