Bộ giáo dục và đào tạo

Trường Đại học Ngoại ngữ - Tin học TP.HCM

Đề tài

[DỰ ĐOÁN BỆNH TIỂU ĐƯỜNG]

GVHD: Th.s Vũ Đình Ái

HVTH: 21DH113218 Bùi Tuấn Đạt

Tháng 11 Năm 2023

BẢNG PHÂN CÔNG NHIỆM VỤ

Nhiệm vụ	Tên thành viên	Đánh giá
Thu thập dữ liệu	Bùi Tuấn Đạt	
Phân tích dữ liệu	Bùi Tuấn Đạt	
Xây dựng mô hình	Bùi Tuấn Đạt	
Cài đặt và demo	Bùi Tuấn Đạt	

Mục lục

Tóm tắt

Bệnh tiểu đường là một căn bệnh mà cơ thể không sản xuất hoặc không sử dụng insulin một cách hiệu quả, dẫn đến sự tăng glucose trong máu. Bệnh này có thể gây nên nhiều biến chứng sức khỏe, bao gồm các vấn đề tim mạch, thị lực, thần kinh, và thận. Vì vậy, việc dự đoán bệnh tiểu đường có ý nghĩa quan trọng hỗ trợ chẩn đoán và quản lý bệnh tiểu đường một cách hiệu quả. Báo cáo này sử dụng các thuật toán máy học như hồi quy logistic, cây quyết định và Support vector machine trên tập dữ liệu được thu thập tại trang Kaggle, và kết quả cho thấy với độ chính xác là 78% mà thuật toán hồi quy logistic mang lại là độ chính xác cao nhất.

Từ khoá: Hồi quy Logistic, Cây Quyết định, Support Vector Machine, Bệnh Tiểu đường

I. Giới thiệu

Bệnh tiểu đường là một loại bệnh mà cơ thể không thể sản xuất hoặc sử dụng insulin một cách hiệu quả, dẫn đến việc tăng đường huyết. Bệnh tiểu đường có thể gây ra các biến chứng sức khỏe nghiêm trọng, như bệnh tim, đột quy, và suy thận. Mặc dù không có liệu pháp chữa trị vĩnh viễn cho bệnh tiểu đường, việc đưa ra dự đoán chính xác và sớm có thể giúp điều chỉnh lối sống và điều trị để kiểm soát bệnh tốt hơn.

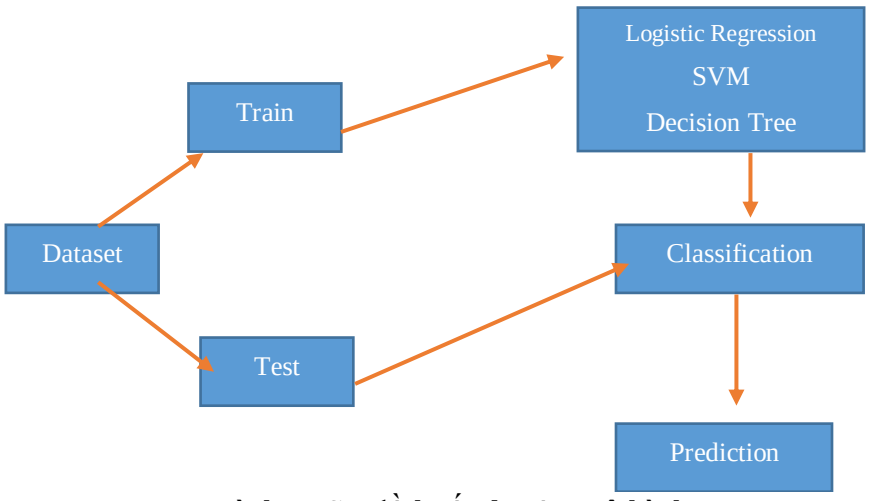
Theo Sở Y tế Thế giới, vào năm 2019, khoảng 463 triệu người trên toàn cầu mắc bệnh tiểu đường và dự kiến con số này sẽ tăng lên 700 triệu vào năm 2045. Việc dự đoán bệnh tiểu đường từ dữ liệu lâm sàng và y học hình ảnh đã trở thành một lĩnh vực nghiên cứu sôi động trong thập kỷ qua. Các phương pháp dự đoán dựa trên máy học và trí tuệ nhân tạo có tiềm năng lớn để cải thiện khả năng chẩn đoán và dự đoán bệnh tiểu đường.

Tính đến thời điểm này, việc áp dụng các phương pháp máy học và dữ liệu lâm sàng để dự đoán bệnh tiểu đường đang là một hướng nghiên cứu và ứng dụng tiềm năng, giúp tạo ra các công cụ hỗ trợ chẩn đoán và quản lý bệnh tiểu đường một cách hiệu quả và chính xác.

Và trong báo cáo này em sẽ áp dụng các thuật toán của máy học như hồi quy logistic , cây quyết định và support vector machine để tiến hành dự đoán bệnh tiểu đường thông qua tập dữ liệu được thu thập tại trang Kaggle.

II. Xây dựng mô hình

Dữ liệu được thập từ trang Kaggle và lưu thành file CSV, sau đó sẽ được chia thành 2 tập dữ liệu có tỉ lệ 80% cho tập huấn luyện và 20% cho tập kiểm tra , kế tiếp chương trình sẽ áp dụng các thuật toán như Hồi quy logistic, Cây quyết định, SVM tiến hành phân lớp thông qua tập huấn luyện. Cuối cùng sẽ dự đoán và đánh giá mô hình thông qua tập kiểm tra.



Hình 1. Sơ đồ huấn luyện mô hình

III. Phân tích dữ liệu

III.1. Giới thiệu về dữ liệu

-Phần trình bày giới thiệu về thông tin của dữ liệu

- Dữ liệu được thu thập tại trang Kaggle, **Cơ sở dữ liệu về bệnh tiểu đường của người da đỏ Pima,** dự đoán sự khởi phát của bệnh tiểu đường dựa trên các biện pháp chẩn đoán.
- Các trường thông tin của dữ liệu diễn đạt thông tin:

  1. Pregnancies: Số lần mang thai
  2. Glucose: Nồng độ đường huyết sau kiểm tra bụng đói
  3. BloodPressure: Huyết áp (mm Hg)
  4. SkinThickness: Độ dày da gập (mm)
  5. Insulin: Insulin huyết thanh đo lường (mu U/ml)
  6. BMI: Chỉ số khối cơ thể (BMI)
  7. DiabetesPedigreeFunction: Hệ số dẫn gen tiểu đường
  8. Age: Tuổi của người được kiểm tra

- Các đoạn code liên quan đến dữ liệu được trích ra từ codelab:

  Đọc dữ liệu:

```
[ ]  data=pd.read_csv('/content/mydrive/MyDrive/Colab Notebooks/diabetes.csv')
```

  Kiểm tra dữ liệu có thiếu không

```
data.isna().sum()
```

  Hiển thị thông tin tổng quan về dữ liệu

```
data.info()
```

  Tạo biểu đồ cột biểu thị số lượng mẫu cho 2 loại kết quả trong cột 'Outcome'

```
data['Outcome'].value_counts().plot(kind='bar',figsize=(7,4))
plt.xlabel('Outcome')
plt.ylabel('Count')
plt.title('Outcome Counts')
plt.show()
```

  Hiển thị thông tin thống kê cho từng cột

```
[ ]  data.describe().transpose().round(2)
```

- Trực quan hoá dữ liệu thông qua matplotlib

  Tạo các biểu đồ histogram cho tất cả các cột

```
# Histogram of data
data.hist(figsize=(20,20))
plt.show()
```

  Tạo biểu đồ cột đếm số lần xuất hiện của từng nhóm tuổi

```
[ ] # Kiểm tra nhóm tuổi có số lượng bệnh nhân tiểu đường cao nhất
    plt.figure(figsize=(12, 6))
    sns.countplot(x='Age', data=data[data['Outcome'] == 1], order=data[data['Outcome'] == 1]['Age'].value_counts().index)
    plt.title('Age Group vs. Outcome=1')
    plt.xlabel('Age Group')
    plt.ylabel('Count of Outcome=1')
    plt.xticks(rotation=90)  # Xoay nhãn trục x để dễ đọc hơn
    plt.show()
```

  Hiển thị các hệ số tương quan giữa từng cặp cột

```
correlation_matrix = data.corr().round(2)
correlation_matrix
```

```
# Create a heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()
```

  Trực quan hóa phân phối của dữ liệu

```
sns.boxplot(data, orient='h', flierprops={'markerfacecolor': 'blue', 'marker': 'o'})
plt.figure(figsize=(10,6))
```

  => loại bỏ các ngoại lệ từ DataFrame đầu vào chúng có thể làm sai lệch kết quả phân tích và dẫn đến những kết luận không chính xác

```
def remove_outlier (dataFrame):
  for column_name in dataFrame.columns:
    Q1 = data[column_name].quantile(0.25)
    Q3 = data[column_name].quantile(0.75)
    IQR = Q3 - Q1
    lower_limit = Q1 - 1.5*IQR
    upper_limit = Q3 + 1.5*IQR
    print(f"{column_name} >> Lower limit: {lower_limit} \n Upper limit: {upper_limit}")
    dataFrame = dataFrame[(dataFrame[column_name] > lower_limit)|(dataFrame[column_name] < upper_limit)]

  return dataFrame
```

```
[6] data = remove_outlier(data)
```

  Đánh giá mức độ chính xác của mô hình phân loại qua ma trận nhầm lẫn (confusion matrix)

```
from sklearn.metrics import accuracy_score, precision_score, classification_report, confusion_matrix
cm = confusion_matrix(y_test, lr_pred)
cm
```

```
array([[87, 12],
       [22, 33]])
```

```
sns.heatmap(cm,annot=True,fmt="d")
```

```
[ ] cm_svm = confusion_matrix(y_test, sv_pred)
    cm_svm

    array([[85, 14],
           [21, 34]])
```

```
[ ] sns.heatmap(cm_svm,annot=True,fmt="d")
```

```
[ ] cm_rf = confusion_matrix(y_test, rf_pred)
    cm_rf

    array([[79, 20],
           [17, 38]])
```

```
[ ] sns.heatmap(cm_rf,annot=True,fmt="d")
```

III.2. Trích chọn đặc trưng

-Phần trình bày giới thiệu về các thông tin dữ liệu đưa vào tập huấn luyện.

- Các đoạn code liên quan đế dữ liệu được trích ra từ codelab

  Làm sạch dữ liệu

  Có thể bệnh nhân không mang thai nên cột Mang thai có thể bằng 0. Do đó loại trừ cột mang thai. Tuy nhiên Glucose, BloodPressure, SkinThickness, Insulin & BMI có thể bằng 0.

  Quy trình thay thế giá trị 0 để làm sạch dữ liệu

  1 Tính giá trị trung bình, loại trừ giá trị 0

  2 Thay thế giá trị 0 bằng giá trị trung bình

```
[13] columns_to_replace_Zero = ['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']

     for column in columns_to_replace_Zero:
         mean_value = data[data[column] != 0][column].mean()
         data[column] = data[column].replace(0, mean_value)
```

```
# Kiểm tra dữ liệu sau khi thay thế giá trị 0
data.describe().transpose().round(2) #Updated values
```

  Loại bỏ các ngoại lệ

```
[19] def remove_outlier (dataFrame):
       for column_name in dataFrame.columns:
         Q1 = data[column_name].quantile(0.25)
         Q3 = data[column_name].quantile(0.75)
         IQR = Q3 - Q1
         lower_limit = Q1 - 1.5*IQR
         upper_limit = Q3 + 1.5*IQR
         print(f"{column_name} >> Lower limit: {lower_limit} \n Upper limit: {upper_limit}")
         dataFrame = dataFrame[(dataFrame[column_name] > lower_limit)|(dataFrame[column_name] < upper_limit)]

       return dataFrame
```

```
data = remove_outlier(data)
```

III.3. Chuẩn hoá dữ liệu

-Các đoạn code liên quan đế dữ liệu được trích ra từ codelab

```
# Tách dữ liệu trong các tính năng và cột đầu ra.
X = data[['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']].values
y = data['Outcome'].values
```

```
# Chia tỷ lệ các tính năng đầu vào thành bộ chia tiêu chuẩn
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X = scaler.fit_transform(X)
Y = data['Outcome']
```

III.4. Phân chia dữ liệu

-Các đoạn code liên quan đế dữ liệu được trích ra từ codelab

```
[ ] from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.8, random_state = 42)
```

```
[ ] X_train.shape
```

```
(614, 8)
```

```
X_test.shape
```

```
(154, 8)
```

IV. Huấn luyện mô hình

-Các đoạn code liên quan đế dữ liệu được trích ra từ codelab

```
[ ] from sklearn.linear_model import LogisticRegression
    logReg = LogisticRegression()
    logReg.fit(X_train, y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

```
[ ] from sklearn import svm
    classifier = svm.SVC(kernel='linear')
    classifier.fit(X_train, y_train)
```

```
▾        SVC
SVC(kernel='linear')
```

```
[ ] from sklearn.ensemble import RandomForestClassifier
    model=RandomForestClassifier(n_estimators=110, random_state=42,criterion='entropy')
    model.fit(X_train,y_train)
```

```
▾                              RandomForestClassifier
RandomForestClassifier(criterion='entropy', n_estimators=110, random_state=42)
```

V. Đánh giá mô hình

-Các đoạn code liên quan đế dữ liệu được trích ra từ codelab

Train score & Test score

```
[35] # Train score & Test score of Logistic Regression
     from sklearn.metrics import accuracy_score
     print("Train accuracy of Logistic Regression", logReg.score(X_train,y_train)*100)
     print("Accuracy (Test) score of Logistic Regression", logReg.score(X_test, y_test)*100)
     print("Accuracy (Test) score of Logistic Regression", accuracy_score(y_test, lr_pred)*100)
```

```
Train accuracy of Logistic Regression 76.2214983713355
Accuracy (Test) score of Logistic Regression 77.92207792207793
Accuracy (Test) score of Logistic Regression 77.92207792207793
```

```
[36] # Train score & Test score of SVM
     print("Train accuracy of SVM", classifier.score(X_train,y_train)*100)
     print("Accuracy (Test) score of SVM", classifier.score(X_test, y_test)*100)
     print("Accuracy score of SVM", accuracy_score(y_test, sv_pred)*100)
```

```
Train accuracy of SVM 76.54723127035831
Accuracy (Test) score of SVM 77.27272727272727
Accuracy score of SVM 77.27272727272727
```

```
[37] # Train score & Test score of Random Forest
     print("Train accuracy of Random Forest", model.score(X_train,y_train)*100)
     print("Accuracy (Test) score of Random Forest", model.score(X_test, y_test)*100)
     print("Accuracy score of Random Forest", accuracy_score(y_test, rf_pred)*100)
```

```
Train accuracy of Random Forest 100.0
Accuracy (Test) score of Random Forest 75.97402597402598
Accuracy score of Random Forest 75.97402597402598
```

Đánh giá mô hình phân loại qua ma trận nhầm lẫn (confusion matrix)

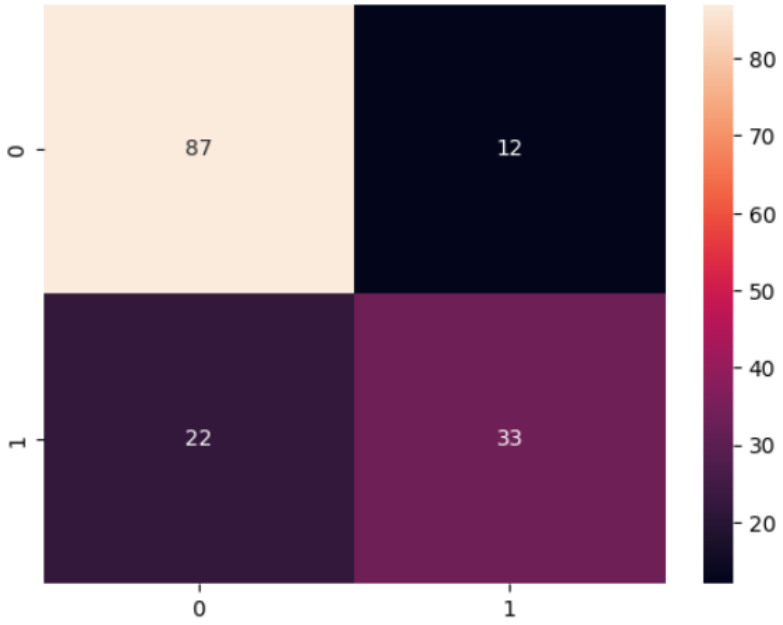
Confusion matrix of Logistic Regression

```
from sklearn.metrics import accuracy_score, precision_score, classification_report, confusion_matrix
cm = confusion_matrix(y_test, lr_pred)
cm
```

```
array([[87, 12],
       [22, 33]])
```

```
[39] sns.heatmap(cm,annot=True,fmt="d")
```

```
<Axes: >
```



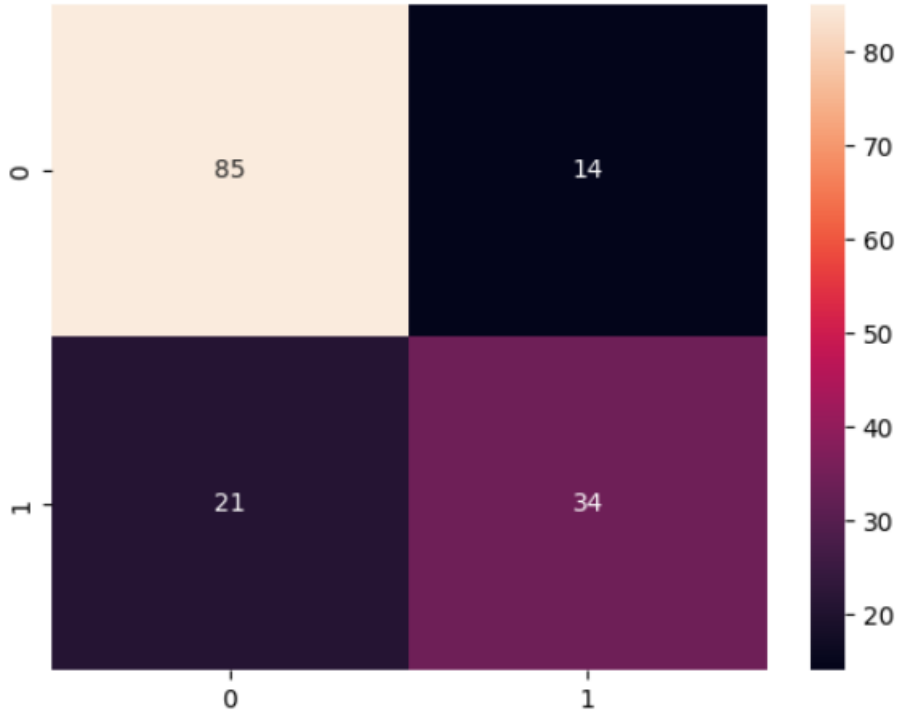
Confusion matrix of SVM

```
cm_svm = confusion_matrix(y_test, sv_pred)
cm_svm
```

```
array([[85, 14],
       [21, 34]])
```

```
[43] sns.heatmap(cm_svm,annot=True,fmt="d")
```

```
<Axes: >
```



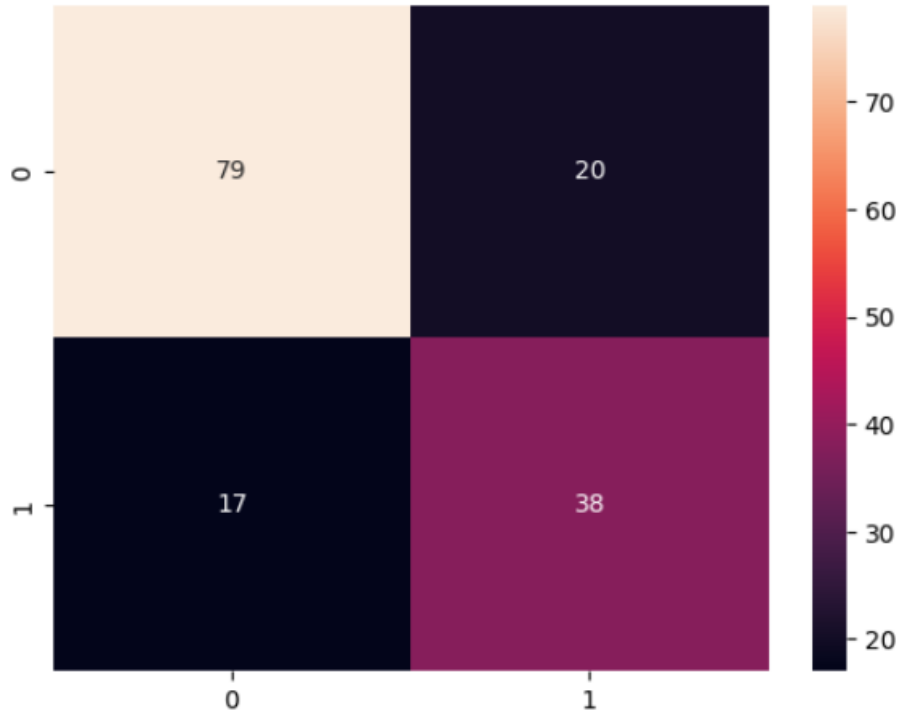
Confusion matrix of Random Forest

```
[46] cm_rf = confusion_matrix(y_test, rf_pred)
     cm_rf

     array([[79, 20],
            [17, 38]])
```

```
[47] sns.heatmap(cm_rf,annot=True,fmt="d")

     <Axes: >
```



In ra báo cáo đánh giá hiệu suất của mô hình dự đoán

```
print(classification_report(y_test, lr_pred ))
```

```
              precision    recall  f1-score   support

           0       0.80      0.88      0.84        99
           1       0.73      0.60      0.66        55

    accuracy                           0.78       154
   macro avg       0.77      0.74      0.75       154
weighted avg       0.78      0.78      0.77       154
```

Điểm chính xác là 0,73 cho thấy rằng, trong số các mẫu được dự đoán là dương tính (bệnh tiểu đường), 73% trong số đó là dương tính thực sự. Độ chính xác là thước đo mức độ mô hình xác định các trường hợp dương tính đồng thời giảm thiểu các trường hợp dương tính giả. Tóm lại, độ chính xác là 0,78, cùng với điểm chính xác là 0,73.

```
print(classification_report(y_test, sv_pred ))
```

```
              precision    recall  f1-score   support

           0       0.80      0.86      0.83        99
           1       0.71      0.62      0.66        55

    accuracy                           0.77       154
   macro avg       0.76      0.74      0.74       154
weighted avg       0.77      0.77      0.77       154
```

Điểm chính xác là 0,71 cho thấy rằng, trong số các mẫu được dự đoán là dương tính (bệnh tiểu đường), 71% trong số đó là dương tính thực sự. Độ chính xác là 0,77, cùng với điểm chính xác là 0,71.

```
[48] print(classification_report(y_test, rf_pred ))
```

```
              precision    recall  f1-score   support

           0       0.82      0.80      0.81        99
           1       0.66      0.69      0.67        55

    accuracy                           0.76       154
   macro avg       0.74      0.74      0.74       154
weighted avg       0.76      0.76      0.76       154
```

Điểm chính xác là 0,66 cho thấy rằng, trong số các mẫu được dự đoán là dương tính (bệnh tiểu đường), 66% trong số đó là dương tính thực sự. Độ chính xác là 0,76, cùng với điểm chính xác là 0,76.

VI. Kết luận

Phần trình bày kết luận về báo cáo gồm các nội dung

- Các kết quả đã triển khai

Em đã triển khai được các mô hình (Hồi quy logistic, Cây quyết định, SVM) tiến hành phân lớp thông qua tập huấn luyện và dự đoán, đánh giá mô hình thông qua tập kiểm tra.

- Hướng phát triển của đề tài

Cần áp dụng vào nhiều nguồn dữ liệu khác nhau, tìm hiểu thêm nhiều mô hình hơn và áp dụng vào 1 trang web hay 1 app cụ thể để chuẩn đoán thông qua các số liệu thu thập.

Tài liệu tham khảo (IEEE)

[1] https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/data

[2] https://www.kaggle.com/code/nsujinsurendran/pima-indians-diabetes-database

[3] https://www.youtube.com/watch?v=ZAbiKPeIUxU

[4] https://www.youtube.com/watch?v=YkkgEk_fZ9A&t=758s

[5] https://www.youtube.com/watch?v=xUE7SjVx9bQ