TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CÁ NHÂN MÔN KHAI THÁC DỮ LIỆU VÀ KHAI PHÁ TRI THỨC**

**TÌM HIỂU VÀ DEMO VỀ LINEAR REGRESSION**

*Người hướng dẫn*: **TS HOÀNG ANH**

*Người thực hiện*: **HUỲNH NGUYỄN TƯỜNG VY – 52000170**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO CÁ NHÂN MÔN KHAI THÁC DỮ LIỆU VÀ KHAI PHÁ TRI THỨC**

**TÌM HIỂU VÀ DEMO VỀ LINEAR REGRESSION**

Người hướng dẫn: **TS HOÀNG ANH**

Người thực hiện: **HUỲNH NGUYỄN TƯỜNG VY**

Lớp **: 20050201**

Khoá  **: 24**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2023**

LỜI CẢM ƠN

Em xin cảm ơn thầy Hoàng Anh đã tận tình chỉ bảo chúng em qua những buổi học tại lớp, thầy đã chỉ em cách thức làm bài, chỉ điểm những chỗ còn sai sót chưa phù hợp. Nếu không có những lời hướng dẫn, dạy bảo của thầy thì bài thu hoạch của em cũng rất khó để hoàn thiện. Một lần nữa em xin chân thành cảm ơn thầy.

Bước đầu đi vào thực tế với nền kiến thức mở rộng, kiến thức củaem còn hạn chế và nhiều bỡ ngỡ. Vì thế, trong quá trình biên soạn khó tránh những sai sót, em rất mong nhận được những ý kiến đóng góp quý báu của thầy và các bạn để bài báo cáo hoàn thiện hơn.

# ĐỒ ÁN ĐƯỢC HOÀN THÀNH

# TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là sản phẩm đồ án của riêng tôi / chúng tôi và được sự hướng dẫn của TS Nguyễn Văn A;. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong đồ án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung đồ án của mình.** Trường đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày tháng năm*

*Tác giả*

*(ký tên và ghi rõ họ tên)*

|  |
| --- |
| *Vy*  *Huỳnh Nguyễn Tường Vy* |

|  |  |  |
| --- | --- | --- |
|  |  |  |

# PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

**Phần xác nhận của GV hướng dẫn**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

**Phần đánh giá của GV chấm bài**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

CHƯƠNG 1 – PHÂN CÔNG CÁ NHÂN

CHƯƠNG 2 – KẾT QUẮ THỰC HIỆN

**MỤC LỤC**

[LỜI CẢM ƠN i](#_Toc155946784)

[ĐỒ ÁN ĐƯỢC HOÀN THÀNH ii](#_Toc155946785)

[TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG ii](#_Toc155946786)

[PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN iii](#_Toc155946787)

[TÓM TẮT v](#_Toc155946788)

[DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT 2](#_Toc155946789)

[DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ 3](#_Toc155946790)

[CHƯƠNG 1 – PHÂN CÔNG CÁ NHÂN 4](#_Toc155946791)

[1.1 Phân công công việc 4](#_Toc155946792)

[CHƯƠNG 2 – KẾT QUẢ CÔNG VIỆC 5](#_Toc155946793)

[2.1 Các phương pháp cải tiến Linear Regression 5](#_Toc155946794)

[2.2 Tìm dataset và thực hiện demo 9](#_Toc155946795)

[TÀI LIỆU THAM KHẢO 29](#_Toc155946796)

[PHỤ LỤC 30](#_Toc155946797)

# DANH MỤC KÍ HIỆU VÀ CHỮ VIẾT TẮT

**CÁC KÝ HIỆU**

**CÁC CHỮ VIẾT TẮT**

DANH MỤC CÁC BẢNG BIỂU, HÌNH VẼ, ĐỒ THỊ

**DANH MỤC HÌNH**

[Hình 1. Dạng tổng quát của Polynomial Regression 8](#_Toc155946808)

[Hình 2. Một số dạng thường gặp của Polynomial Regression 8](#_Toc155946809)

[Hình 3. So sánh Simple Linear Regression và Polymial model 8](#_Toc155946810)

[Hình 4. Định nghĩa Huber Loss 9](#_Toc155946811)

[Hình 5. Trực quan hoá dữ liệu (1) 11](#_Toc155946812)

[Hình 6. Trực quan hoá dữ liệu (2) 11](#_Toc155946813)

[Hình 7. Dữ liệu sau khi chuyển đổi 23](#_Toc155946814)

CHƯƠNG 1 – PHÂN CÔNG CÁ NHÂN

* 1. Phân công công việc
* Tìm hiểu các phương pháp cải tiến Linear Regression
* Tìm dataset & thực hiện demo

CHƯƠNG 2 – KẾT QUẢ CÔNG VIỆC

2.1 Các phương pháp cải tiến Linear Regression

2.1.1. Regularization

Regularization trong linear regression là một kỹ thuật nhằm ngăn chặn hiện tượng overfitting (quá khớp) của mô hình hồi quy tuyến tính. Overfitting xảy ra khi mô hình học quá sát dữ liệu huấn luyện, dẫn đến khả năng tổng quát hóa kém trên dữ liệu mới. Regularization giải quyết vấn đề này bằng cách điều chỉnh hàm chi phí (cost function) để hạn chế độ phức tạp của mô hình và ưu tiên những mô hình đơn giản hơn.

Cách thức hoạt động:

Thêm một số hạng mới vào hàm chi phí:

* L1 Regularization (Lasso Regression): Thêm tổng trị tuyệt đối của các tham số (theta).
  + Nếu lambda bằng 0 thì chúng ta sẽ nhận lại OLS (bình phương tối thiểu thông thường) trong khi một giá trị rất lớn sẽ làm cho các hệ số bằng 0, nghĩa là nó sẽ trở nên không phù hợp.
* L2 Regularization (Ridge Regression): Thêm tổng bình phương của các tham số (theta).
  + Nếu lambda bằng 0 thì trở lại lại OLS. Tuy nhiên, nếu lambda quá lớn thì nó sẽ tăng thêm trọng lượng và dẫn đến trang bị thiếu. Phải nói rằng, cách chúng ta chọn lambda rất quan trọng. Kỹ thuật này hoạt động rất tốt để tránh các vấn đề về overfitting.
* Sự khác biệt chính giữa các kỹ thuật này là lasso thu nhỏ hệ số của tính năng ít quan trọng hơn về 0, do đó loại bỏ hoàn toàn một số tính năng. Nói cách khác, chính quy hóa L1 hoạt động tốt cho việc lựa chọn tính năng trong trường hợp chúng ta có nhiều tính năng (features)
* Ảnh hưởng:
  + Thu nhỏ các tham số (theta) về 0, khiến một số đặc trưng (feature) ít quan trọng hơn.
  + Giảm độ phức tạp của mô hình, giúp giảm thiểu overfitting.
* Các loại Regularization thường gặp:
  + L1 Regularization: - Tốt cho việc chọn lọc đặc trưng (feature selection).Có thể đưa một số tham số về 0, loại bỏ chúng khỏi mô hình.
  + L2 Regularization: - Không loại bỏ hoàn toàn các đặc trưng. Thường được sử dụng phổ biến hơn vì nó tạo ra các mô hình ổn định hơn.
* Lựa chọn tham số lambda:
  + Tham số lambda kiểm soát mức độ ảnh hưởng của regularization.
  + Lambda lớn hơn dẫn đến regularization mạnh hơn và mô hình đơn giản hơn.
  + Lambda tối ưu thường được tìm kiếm thông qua kỹ thuật cross-validation.

2.1.2 Gradient Descent Variants

- Gradient Descent Variants: Gradient Descent là một kỹ thuật tối ưu hóa phổ biến trong ML và Học sâu và nó có thể được sử dụng với hầu hết, các thuật toán học tập. Gradient là độ dốc của một hàm. Nó đo lường mức độ thay đổi của một biến để đáp ứng với những thay đổi của một biến khác. Về mặt toán học, Gradient Descent là một hàm lồi có đầu ra là đạo hàm riêng của một tập các tham số đầu vào của nó. Gradient càng lớn thì độ dốc càng lớn.

- Bắt đầu từ một giá trị ban đầu, Gradient Descent được chạy lặp đi lặp lại để tìm các giá trị tối ưu của các tham số để tìm giá trị nhỏ nhất có thể có của chi phí hàm đã cho.

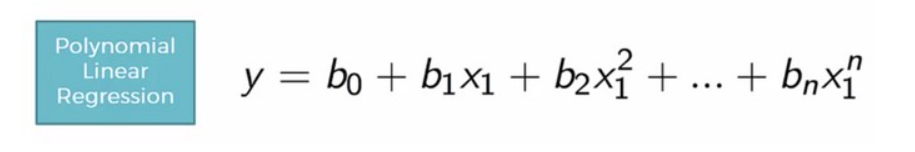
- SDG: Stochastic Gradient Descent (SGD) là một thuật toán tối ưu hóa được sử dụng trong học máy, đặc biệt là trong huấn luyện các mô hình mạng nơ-ron. Khác với Gradient Descent truyền thống, SGD chỉ sử dụng một mẫu duy nhất trong mỗi lần lặp để tính toán gradient. Điều này giúp SGD trở nên hiệu quả hơn về mặt tính toán, đặc biệt là với các tập dữ liệu lớn.

* Ưu điểm của SGD:
* Hiệu quả về mặt tính toán
* Có thể thoát khỏi các điểm cực tiểu cục bộ
* Dễ dàng thực hiện
* Nhược điểm của SGD:
* Độ ổn định có thể kém hơn so với Gradient Descent truyền thống
* Cần điều chỉnh cẩn thận các tham số học tập
* Batch Gradient Descent
* Là một thuật toán tối ưu hóa được sử dụng để tìm điểm tối thiểu của một hàm mất mát.
* Để làm vậy, Batch Gradient Descent sử dụng tất cả các điểm dữ liệu trong tập huấn luyện để tính toán đạo hàm của hàm mất mát tại mỗi lần lặp.
* Ưu điểm:
* Độ ổn định cao
* Có thể đạt được độ chính xác cao
* Nhược điểm:
* Không hiệu quả với các tập dữ liệu lớn
* Không hiệu quả với online learning

2.1.3. Polynomial Regression

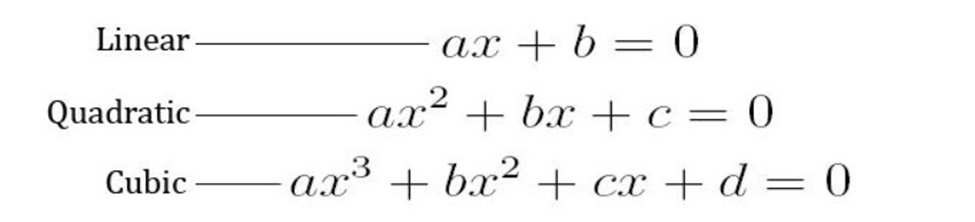
Polynomial Regression là thuật toán hồi quy đa thức, nó giống như thuật toán hồi quy tuyến tính, sử dụng mối quan hệ giữa các biến độc lập x và biến phụ thuộc y được biểu diễn dưới dạng đa thức bậc n, để tìm cách tốt nhất vẽ một đường qua các điểm dữ liệu sao cho tối ưu và phù hợp nhất. Polynomial Regression là một thuật toán trong machine learning, nó được dùng cho các bài toán về dự đoán, dự báo (prediction).

Dạng tổng quát của Polynomial Regression như sau:

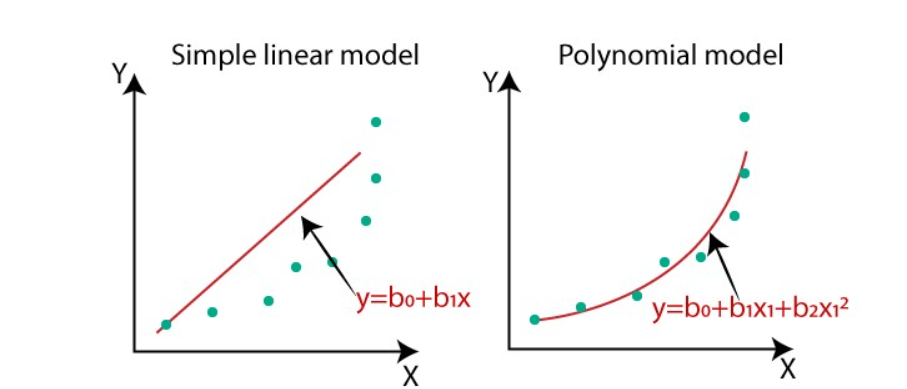


Hình 1. Dạng tổng quát của Polynomial Regression

Một số dạng thường gặp của Polynomial Regression:



Hình 2. Một số dạng thường gặp của Polynomial Regression

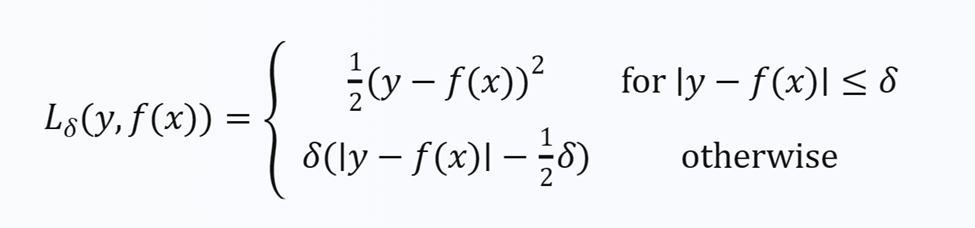


Hình 3. So sánh Simple Linear Regression và Polymial model

2.2.4. Huber Loss

Huber Loss là một hàm mất mát được sử dụng trong linear regression để giảm ảnh hưởng của các điểm ngoại lệ (outliers) so với Mean Squared Error (MSE). Trong mô hình linear regression, MSE có thể là nhạy cảm với các giá trị ngoại lệ và dẫn đến việc mô hình hóa không chính xác.

* Huber Loss được định nghĩa như sau:



Hình 4. Định nghĩa Huber Loss

Trong đó:

* + y là giá trị thực tế
  + f(x) là giá trị dự đoán
  + δ là một hằng số dương.
  + Khi ∣y−f(x)∣≤δ, hàm loss sử dụng bình phương sai số giữa giá trị dự đoán và giá trị thực tế.
  + Khi ∣y−f(x)∣>δ, hàm loss sử dụng hàm tuyến tính để giảm thiểu ảnh hưởng của các giá trị ngoại lai.

2.2 Tìm dataset và thực hiện demo

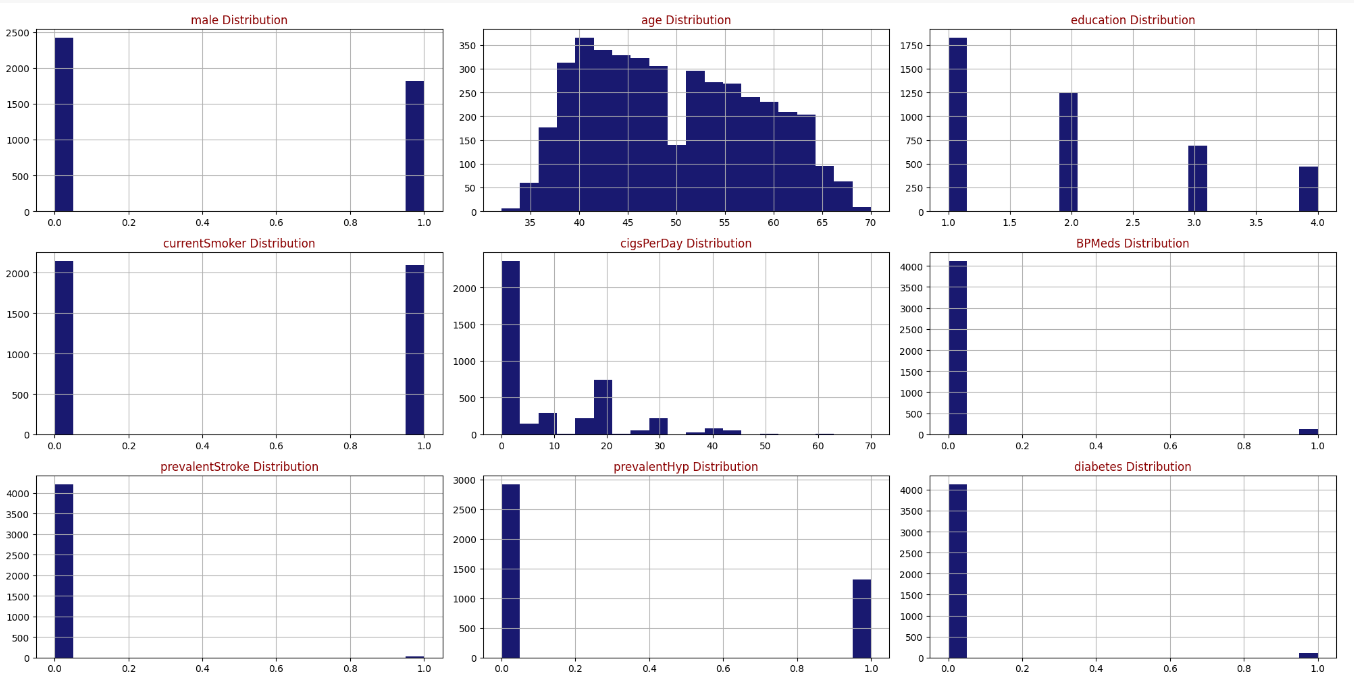
2.2.1 Demo dự đoán một người có mắc bệnh tim hay không dựa trên chỉ số BMI:

2.2.1.1 Dataset

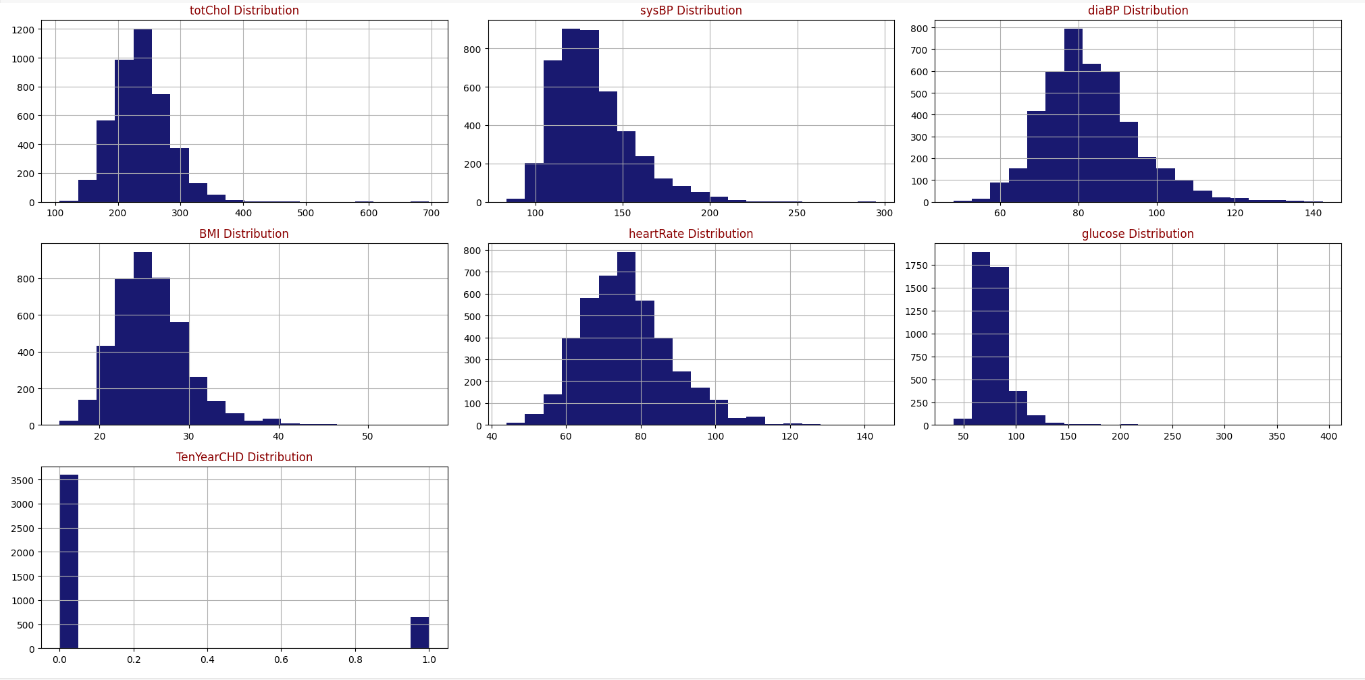
- Thông tin dataset:

* Male: Cho biết cá nhân là nam (1) hay không (0).
* Age: Thể hiện tuổi của cá nhân tính bằng năm.
* Education: Cho biết trình độ học vấn của cá nhân.
* currentSmoker: Cho biết cá nhân có phải là người hút thuốc hiện tại (1) hay không (0).
* cigsPerDay: Thể hiện số điếu thuốc trung bình mỗi ngày mà cá nhân hút.
* BPMeds: Có thể cho biết cá nhân có đang sử dụng thuốc huyết áp (1) hay không (0).
* prevalentStroke: Cho biết cá nhân có tiền sử đột quỵ (1) hay không (0).
* prevalentHyp: Cho biết cá nhân có tiền sử tăng huyết áp (huyết áp cao) (1) hay không (0).
* Diabetes: Cho biết cá nhân mắc bệnh tiểu đường (1) hay không (0).
* totChol: Thể hiện mức cholesterol toàn phần của cá nhân.
* sysBP: Thể hiện huyết áp tâm thu của cá nhân.
* diaBP: Thể hiện huyết áp tâm trương của cá nhân.
* BMI: Thể hiện chỉ số khối cơ thể của cá nhân.
* heartRate: Thể hiện nhịp tim của cá nhân.
* Glucose: Thể hiện mức glucose của cá nhân.
* TenYearCHD: Cho biết cá nhân có phát triển bệnh tim mạch vành (CHD) trong vòng 10 năm quan sát (1) hay không (0).

- Trực quan hoá dữ liệu:



Hình 5. Trực quan hoá dữ liệu (1)



Hình 6. Trực quan hoá dữ liệu (2)

2.2.1.2 Code demo:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.impute import SimpleImputer

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import MinMaxScaler

from sklearn.pipeline import make\_pipeline

from sklearn.dummy import DummyClassifier

from sklearn.linear\_model import LogisticRegression, LinearRegression

from sklearn.metrics import accuracy\_score , ConfusionMatrixDisplay , classification\_report,r2\_score

data= pd.read\_csv("/content/framingham.csv")

data.shape

data.info()

data.describe()

#EDA

#Missing data

data.isnull().sum()

#Handle missing value

data["education"].fillna(data["education"].mode()[0],inplace=True)

data["cigsPerDay"].fillna(data["cigsPerDay"].mode()[0],inplace=True)

data["BPMeds"].fillna(data["BPMeds"].mode()[0],inplace=True)

data["totChol"].fillna(data["totChol"].mode()[0],inplace=True)

data["BMI"].fillna(data["BMI"].mode()[0],inplace=True)

data["heartRate"].fillna(data["heartRate"].mode()[0],inplace=True)

data["glucose"].fillna(data["glucose"].mode()[0],inplace=True)

corr=data.drop(columns='TenYearCHD').corr()

plt.subplots(figsize=(25,20))

sns.heatmap(corr,annot=True)

def draw\_histograms(dataframe, features, rows, cols):

fig=plt.figure(figsize=(20,20))

for i, feature in enumerate(features):

ax=fig.add\_subplot(rows,cols,i+1)

dataframe[feature].hist(bins=20,ax=ax,facecolor='midnightblue')

ax.set\_title(feature+" Distribution",color='DarkRed')

fig.tight\_layout()

plt.show()

draw\_histograms(data,data.columns,6,3)

#Preprocessing

#Define X and Y

X= data.drop('TenYearCHD', axis='columns')

Y=data['TenYearCHD']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,Y,test\_size=0.2 , shuffle=True , random\_state=42)

scaler = MinMaxScaler()

X = scaler.fit\_transform(X)

dummy\_classifier = DummyClassifier(strategy='most\_frequent')

dummy\_classifier.fit(X\_train , y\_train)

y\_pred = dummy\_classifier.predict(X\_test)

accuracy = accuracy\_score(y\_test , y\_pred)

print(f"Baseline Model Accuracy: {accuracy:.4f}")

Logis=make\_pipeline(

SimpleImputer(strategy='mean'),

MinMaxScaler(),

LogisticRegression()

)

Logis.fit(X\_train , y\_train)

Logis.score(X\_train,y\_train)

Logis\_pred=Logis.predict(X\_test)

Logis\_score=accuracy\_score(y\_test,Logis\_pred)

Logis\_score

from sklearn.metrics import mean\_squared\_error, r2\_score

mse = mean\_squared\_error(y\_test, Logis\_pred)

r2 = r2\_score(y\_test, Logis\_pred)

print("Mean squared error:", mse)

print("R-squared:", r2)

#Giá trị R^2 trong logistic regression thường thấp hơn linear regression và có thể mang cả giá trị âm.

# Để đánh giá mô hình Logistic => Sử dụng các chỉ số phù hợp với mục đích phân loại:

#Accuracy: Tỉ lệ dự đoán chính xác.

#Sensitivity: Tỉ lệ phân loại chính xác các trường hợp thực sự thuộc lớp target.

#Specificity: Tỉ lệ phân loại chính xác các trường hợp thực sự không thuộc lớp target.

#F1-score: Chỉ số cân bằng giữa precision (chính xác) và recall (độ bao phủ).

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix

cm = confusion\_matrix(y\_test, Logis\_pred)

accuracy = accuracy\_score(y\_test, Logis\_pred)

precision = precision\_score(y\_test, Logis\_pred)

recall = recall\_score(y\_test, Logis\_pred)

f1 = f1\_score(y\_test, Logis\_pred)

# Extract elements from the confusion matrix

tn, fp, fn, tp = cm.ravel()

# Calculate additional metrics

npv = tn / (tn + fn) # Negative Predictive Value

specificity = tn / (tn + fp) # Specificity

commnet\_acc = f"Mô hình đạt độ chính xác {accuracy:.2%}. Điều này có nghĩa là tỷ lệ dự đoán đúng về rủi ro mắc bệnh "

comment\_pre = f"Tỷ lệ precision là {precision:.2%}. Có nghĩa là {precision:.2%} trong số các trường hợp được dự đoán là gặp rủi ro mắc bệnh thì thực sự là mắc bệnh. "

comment\_recall = f"Tỷ lệ recall là {recall:.2%}. Tức là mô hình đoán đúng {recall:.2%} các trường hợp gặp rủi ro thực sự "

comment\_f1 = f"F1-score đạt {f1:.2f}. Là một độ đo kết hợp giữa precision và recall, là trung bình cộng của precision và recall, được điều chỉnh theo độ nặng của precision và recall."

comment\_npv = f"Tỷ lệ NPV là {npv:.2%}. Điều này cho biết {npv:.2%} trong số các trường hợp được dự đoán là không gặp mắc bệnh thì thực sự không mắc bệnh"

# Bình luận về Specificity

comment\_spe = f"Tỷ lệ specificity là {specificity:.2%}. Điều này là độ đo về khả năng mô hình loại bỏ đúng các trường hợp không mắc bệnh"

print(commnet\_acc+"\n" +comment\_pre+"\n" +comment\_recall+"\n" +comment\_f1+"\n" +comment\_npv+"\n"+comment\_spe+"\n" )

ConfusionMatrixDisplay.from\_estimator(Logis , X\_test , y\_test);

print(classification\_report(y\_test,Logis\_pred))

# 0,61,3,1,30,0,0,1,0,225,150,95,28.58,65,103,1

# Example new data

new\_data = pd.DataFrame({

'male': [1],

'age': [61],

'education': [3],

'currentSmoker': [1],

'cigsPerDay': [35],

'BPMeds': [0],

'prevalentStroke': [0],

'prevalentHyp': [1],

'diabetes': [0],

'totChol': [225],

'sysBP': [150],

'diaBP': [95],

'BMI': [28.58],

'heartRate': [65],

'glucose': [103],

})

new\_predictions = Logis.predict\_proba(new\_data)

print(f'Predicted TenYearCHD: {new\_predictions[0]}')

predictions=Logis.predict(new\_data)

print(predictions[0])

Linear = LinearRegression()

Linear.fit(X\_train,y\_train)

Linear.score(X\_train,y\_train)

Linear\_pred=Linear.predict(X\_test)

from sklearn.metrics import mean\_squared\_error, r2\_score

mse = mean\_squared\_error(y\_test, Linear\_pred)

r2 = r2\_score(y\_test, Linear\_pred)

print("Mean squared error:", mse)

print("R-squared:", r2)

new\_predictions\_1 = Linear.predict(new\_data)

print(f'Predicted TenYearCHD: {new\_predictions\_1[0]}')

from sklearn.linear\_model import Lasso

lasso\_model = Lasso(alpha=0.0000000000000005) # You can adjust the alpha value

# Fit the model to the training data

lasso\_model.fit(X\_train, y\_train)

# Make predictions on the testing data

y\_pred = lasso\_model.predict(X\_test)

# Evaluate the model

r2 = r2\_score(y\_test, y\_pred)

print(f"R-squared (R2) score: {r2}")

from sklearn.linear\_model import Ridge

ridge\_model = Ridge(alpha=0.0000000000000005) # You can adjust the alpha value

# Fit the model to the training data

ridge\_model.fit(X\_train, y\_train)

# Make predictions on the testing data

y\_pred = ridge\_model.predict(X\_test)

# Evaluate the model

r2 = r2\_score(y\_test, y\_pred)

print(f"R-squared (R2) score: {r2}")

from sklearn.linear\_model import SGDRegressor

sgd\_regressor = SGDRegressor(max\_iter=1000, tol=1e-3, alpha=0.01, learning\_rate="invscaling", eta0=0.01)

sgd\_regressor.fit(X\_train, y\_train)

y\_pred = ridge\_model.predict(X\_test)

# Evaluate the model

r2 = r2\_score(y\_test, y\_pred)

print(f"R-squared (R2) score: {r2}")

2.2.2 Demo dự đoán rủi ro tín dụng

2.2.2.1 Dataset:

- Thông tin dataset:

* Age (numeric): Đây là số tuổi của khách hàng, dạng số.
* Sex (text: male, female): Đây là giới tính của khách hàng, dạng chữ, chỉ có giá trị "male" (nam) và "female" (nữ).
* Job (numeric: 0 - unskilled and non-resident, 1 - unskilled and resident, 2 - skilled, 3 - highly skilled): Đây là loại công việc của khách hàng, dạng số nguyên với 4 bậc thang:
* 0: Chưa có kỹ năng và không cư trú
* 1: Chưa có kỹ năng nhưng cư trú
* 2: Có kỹ năng (thợ)
* 3: Kỹ năng cao
* Housing (text: own, rent, or free): Đây là hình thức chỗ ở của khách hàng, dạng chữ với các giá trị:
* "own": Sở hữu
* "rent": Thuê
* "free": Tự do
* Saving accounts (text - little, moderate, quite rich, rich): Đây là mức tiết kiệm của khách hàng, dạng ký tự với 4 bậc:
* "little": Ít tiết kiệm
* "moderate": Tiết kiệm vừa phải
* "quite rich": Khá giàu có
* "rich": Giàu có
* Checking account (numeric, in DM - Deutsch Mark): Đây là số dư tài khoản của khách hàng, dạng chữ với các giá trị: "little": Ít tiết kiệm
* "moderate": Tiết kiệm vừa phải
* "rich": Giàu có
* Credit amount (numeric, in DM): Đây là số tiền tín dụng mà khách hàng đã vay, dạng số với đơn vị tính bằng Mark Đức (DM).
* Duration (numeric, in month): Đây là thời gian vay tính bằng tháng.
* Purpose (text: car, furniture/equipment, radio/TV, domestic appliances, repairs, education, business, vacation/others): Đây là mục đích vay tiền của khách hàng, dạng ký tự với các giá trị:
* "car": Mua xe
* "furniture/equipment": Mua đồ nội thất/dụng cụ
* "radio/TV": Mua radio/TV
* "domestic appliances": Mua đồ gia dụng
* "repairs": Sửa chữa
* "education": Giáo dục
* "business": Kinh doanh
* "vacation/others": Du lịch/Khác
* Risk (text: good,vad): Nguy cơ rủi ro tín dụng của khách hàng, dạng chữ với các giá trị:
* good là không xảy ra khả năng rủi ro tính dụng
* bad là có khả năng xảy ra rủi ro

2.2.2.2 Code BackEnd:

Bước 1: chuyển dữ liệu gốc sang dữ liệu chỉ còn ở dạng số:

Code chuyển đổi:

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import random

data= pd.read\_csv("german\_credit\_data.csv",header = 0, names = ['Index', 'Age', 'Sex', 'Job', 'Housing', 'Saving accounts',

'Checking account', 'Credit amount', 'Duration', 'Purpose', 'Risk'])

data['Saving accounts'].fillna('Unknown', inplace=True)

data['Checking account'].fillna('Unknown', inplace=True)

# Convert categorical features to binary

data['Sex'] = data['Sex'].map({'male': 1, 'female': 0})

data['Housing'] = data['Housing'].map({'own': 1, 'rent': 0, 'free': 2})

data['Saving accounts'] = data['Saving accounts'].map({'little': 1, 'moderate': 2, 'rich': 3,'quite rich':4,'Unknown':5})

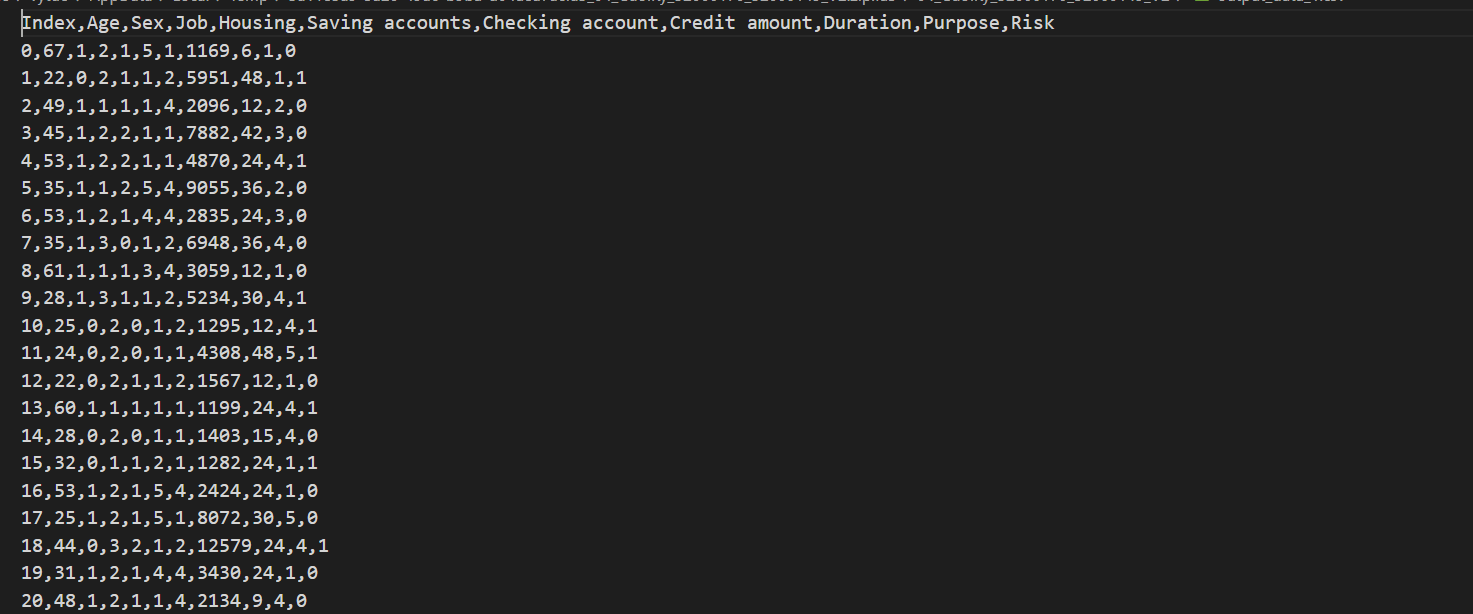
data['Checking account'] = data['Checking account'].map({'little': 1, 'moderate': 2, 'rich': 3,'Unknown':4})

data['Purpose'] = data['Purpose'].map({'radio/TV': 1, 'education': 2, 'furniture/equipment': 3, 'car': 4, 'business': 5,'domestic appliances':6, 'repairs':7, 'vacation/others':8})

data['Risk'] = data['Risk'].map({'good':0, 'bad': 1})

data.to\_csv('output\_data\_1.csv', index=False) # index=False to exclude the index column

Sau khi chuyển đổi, dữ liệu như sau:



Hình 7. Dữ liệu sau khi chuyển đổi

Bước 2: Dự đoán mô hình với các thuật toán khác nhau ( bao gồm logictis regression)

from flask import Flask, render\_template, request

from sklearn.linear\_model import LogisticRegression

from sklearn.tree import DecisionTreeClassifier

from sklearn.neighbors import KNeighborsClassifier

from sklearn.naive\_bayes import GaussianNB

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix

import pandas as pd

import numpy as np

import json

web = Flask(\_\_name\_\_)

app = Flask(\_\_name\_\_, static\_url\_path='/')

data = pd.read\_csv("output\_data\_1.csv")

X = data[['Age', 'Sex', 'Job', 'Housing', 'Saving accounts', 'Checking account', 'Credit amount', 'Duration', 'Purpose']]

y = data['Risk']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2)

@web.route("/")

def home():

return render\_template("index.html")

@web.route('/predict', methods=['POST'])

def predict():

# Get JSON data from the request

data = request.json

# Extract features from JSON data

age = float(data.get('age'))

sex = float(data.get('sex'))

housing = float(data.get('housing'))

saving = float(data.get('saving'))

checking = float(data.get('checking'))

purpose = float(data.get('purpose'))

job=data.get("job")

credit=float(data.get('credit'))

duration=data.get("duration")

algorithm = int(data.get('algorithm'))

para1=(data.get('para1'))

para2=(data.get("para2"))

para3=(data.get("para3"))

# Convert the extracted features into a float array

features\_array = np.array([age, sex, job,housing, saving, checking, credit, duration, purpose], dtype=float)

inp = features\_array.reshape(1, -1)

if algorithm == 1:

model\_type = "Logistic Regression"

pr3=(float(para3))

model = LogisticRegression(solver=para1,penalty=para2, C=pr3,)

elif algorithm == 2:

model\_type = "Decision Tree"

model = DecisionTreeClassifier(max\_depth=2)

elif algorithm == 3:

model\_type = "KNN"

pr1=int(para1)

model = KNeighborsClassifier(n\_neighbors=pr1,metric=para2,weights=para3)

elif algorithm == 4:

model\_type = "Gaussian NB"

model = GaussianNB()

else:

return {'error': 'Invalid Algorithm'}

model.fit(X, y)

temp = evaluate\_model\_performance(model)

prediction = model.predict(inp)

if prediction[0] >= 0.5:

# comment = "The model performs well with high accuracy."

comment="BAD"

# elif accuracy > 0.7:

# comment = "The model shows good accuracy, but there is room for improvement."

else:

# comment = "The model's accuracy is below expectations; consider further optimization."

comment="GOOD"

return json.dumps({'prediction': int(prediction[0]), 'model': model\_type, 'comment': comment,'score':temp})

def evaluate\_model\_performance(model\_train):

# Make predictions on the test data

predictions = model\_train.predict(X\_test)

# Compute confusion matrix

cm = confusion\_matrix(y\_test, predictions)

# Calculate performance metrics

accuracy = accuracy\_score(y\_test, predictions)

precision = precision\_score(y\_test, predictions)

recall = recall\_score(y\_test, predictions)

f1 = f1\_score(y\_test, predictions)

# Extract elements from the confusion matrix

tn, fp, fn, tp = cm.ravel()

# Calculate additional metrics

npv = tn / (tn + fn) # Negative Predictive Value

specificity = tn / (tn + fp) # Specificity

# Determine the overall model comment based on the metrics

commnet\_acc = f"Mô hình đạt độ chính xác {accuracy:.2%}. Điều này có nghĩa là tỷ lệ dự đoán đúng về rủi ro tài chính của khách hàng. "

comment\_pre = f"Tỷ lệ precision là {precision:.2%}. Có nghĩa là {precision:.2%} trong số các trường hợp được dự đoán là gặp rủi ro tín dụng thì thực sự là gặp rủi ro tín dụng. "

comment\_recall = f"Tỷ lệ recall là {recall:.2%}. Tức là mô hình đoán đúng {recall:.2%} các trường hợp gặp rủi ro thực sự "

comment\_f1 = f"F1-score đạt {f1:.2f}. Là một độ đo kết hợp giữa precision và recall, là trung bình cộng của precision và recall, được điều chỉnh theo độ nặng của precision và recall."

comment\_npv = f"Tỷ lệ NPV là {npv:.2%}. Điều này cho biết {npv:.2%} trong số các trường hợp được dự đoán là không gặp rủi ro tín dụng thì thực sự không gặp rủi ro tín dụng"

# Bình luận về Specificity

comment\_spe = f"Tỷ lệ specificity là {specificity:.2%}. Điều này là độ đo về khả năng mô hình loại bỏ đúng các trường hợp không gặp rủi ro tín dụng."

return commnet\_acc+"\n" +comment\_pre+"\n" +comment\_recall+"\n" +comment\_f1+"\n" +comment\_npv+"\n"+comment\_spe+"\n"

if \_\_name\_\_ == '\_\_main\_\_':

web.run(debug=True)

# TÀI LIỆU THAM KHẢO

# PHỤ LỤC