September 7, 2022

Jie Wang

SYRACUSE uNIVERSITY Ischool

[Company address]

Final Project  
Heart Disease Prediction

IST 652 Scripting for Data Analysis

Table of Contents

[Introduction 3](#_Toc113454614)

[Data Sources 3](#_Toc113454615)

[Methodology 4](#_Toc113454616)

[Pre-processing 4](#_Toc113454617)

[Data 1: Record data set 4](#_Toc113454618)

[Data 2: Pubmed data 5](#_Toc113454619)

[Data descriptive analysis 5](#_Toc113454620)

[Data prediction analysis 5](#_Toc113454621)

[Meta analysis 6](#_Toc113454622)

[Result 6](#_Toc113454623)

[descriptive analysis 6](#_Toc113454624)

[prediction factor vs. target 6](#_Toc113454625)

[Correlations between age & serum cholesterol & resting blood pressure 7](#_Toc113454626)

[Explore prediction model 7](#_Toc113454627)

[Prediction models 7](#_Toc113454628)

[Important features 7](#_Toc113454629)

[PDP interact for Age & Cholesterol feature 8](#_Toc113454630)

[PDP interact for ST\_Depression & Rest\_ECG\_Normal & Resting\_Blood\_Pressure 8](#_Toc113454631)

[PDP interact for Thalassemia & Sex\_Male 9](#_Toc113454632)

[meta-analysis 9](#_Toc113454633)

[Conclusion 10](#_Toc113454634)

[Reference 11](#_Toc113454635)

[Supplementary code 12](#_Toc113454636)

# Introduction

In the United States, heart diseases cause more than 859,000 people deaths each year and $216 billion in health care system cost. Up TO 6.3% of ER visits are related to chest pain. An urgent question in these patients is whether they have heart disease, as any delay in diagnosis and treatment can have a negative impact on their prognosis. If patients at low risk for heart disease could be recognized early in the diagnostic process, it has the potential to reduce patient burden, length of stay at the ED, frequency of hospitalization and costs. To diagnose heart disease, a number of tools have been well developed. Three well known risk scores are the GRACE score, the HEART score, and the TIMI score. And risk scores combine and weigh various predictors to calculate the risk of heart disease for an individual patient. They are based on readily available information collected during the initial work-up of chest pain patients and have different focus(1).

A study from 1988 collected the data from Cleveland, Hungary, Switzerland, and Long Beach V. Total 76 attributes, include ‘target’, were included in this dataset. These datasets provide us a very rich resource to study the heart disease prediction factors. In addition, PubMed (23 million references) is an available basic research interface, which offer sophisticated instruments for searching an increasing number of medical publications, including heart diseases.

My central hypothesis is that which factor(s) could be used as a key feature for heart diseases prediction. To this end, 14 attributes will be focused in the heart disease datasets for model prediction. Furthermore, to double confirm the prediction, I will perform a Meta-analysis(2) on the paper regarding heart disease, which could improve precision of estimates of effect, answer questions not posed by the individual studies, and even settle controversies arising from apparently conflicting studies, and provide a supplementary support for my prediction models based on the heart disease data sets.

# Data Sources

There are two main data sources were used for analysis.

**Data source 1**: <https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/>

1. Title: heart disease databases
2. Source Information:
3. Creators:

-- 1. Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D.

-- 2. University Hospital, Zurich, Switzerland: William Steinbrunn, M.D.

-- 3. University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D.

-- 4. V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D.

(b) Donor: David W. Aha (aha@ics.uci.edu) (714) 856-8779 (c) Date: July, 1988

(c) Data description

**Note**: Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0).

**Data source 2:** <https://pubmed.ncbi.nlm.nih.gov/>

# Methodology

**Timeline

Description automatically generated**Figure 1 shows the overview of the study strategy.

## Pre-processing

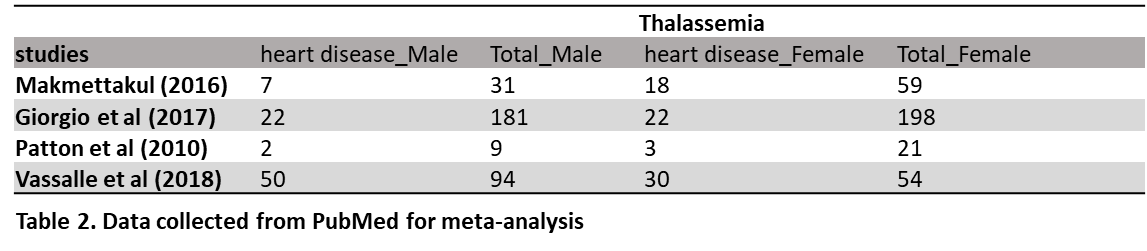
### Data 1: Record data set

**Shape

Description automatically generated with medium confidence**Four datasets are included in this study: ‘prcessed.switzeland.data’; ‘processed.hungarian.data’; ‘processed.cleveland.data’; and ‘processed.va.data’. The data files were then read in csv format with assigned the column name, followed by data frame combination. The data first will be well cleaned. 1) remove duplicated records; 2) removal records with null value or containing “?”. 3) modify the data type. The final clean data was shown in **Table 1**.

### Data 2: Pubmed data

Literatures from PubMed with selected with the key words “Talassemia” and “heart disease” from publish year 1980 to present. Literatures were further selected based on the title and abstracts. The inclusion criteria: thalassemia with heart complications and gender classification. Exclusion criteria: a. not relevant to the topic, thalassemia without heart complications, without gender classification; b. review articles; letters to editor; and case reports. At the end, total four studies are included in this study (**Table 2**).



## Data descriptive analysis

To descriptive general profiles of healthy control vs. heard disease patients. Descriptive analysis will be performed, such as 1) ST depression induced by exercise (Health vs. Heart disease); 2) Resting blood pressure (Health vs. Heart disease); 3) Age (Health vs. Heart disease); 4) serum cholesterol exercise (Health vs. Heart disease); 5) Maximum heart rate achieved (Health vs. Heart disease); 6) Exercise induced angina exercise (Health vs. Heart disease); 7) Number of major vessels (Health vs. Heart disease); 8 Fasting blood sugar (Health vs. Heart disease);

*Please see code for more details.*

## Data prediction analysis

To explore a prediction model. The dataset will be separated into training dataset and test dataset. Different prediction models, such as 1) Naïve Bayes, 2) Support Vector Machine (SVM), 3) K-nearest neighbors (KNN), and 4) Random Forest Classifier, followed by identifying the key feature(s) of prediction models.

*Please see code for more details.*

## Meta analysis

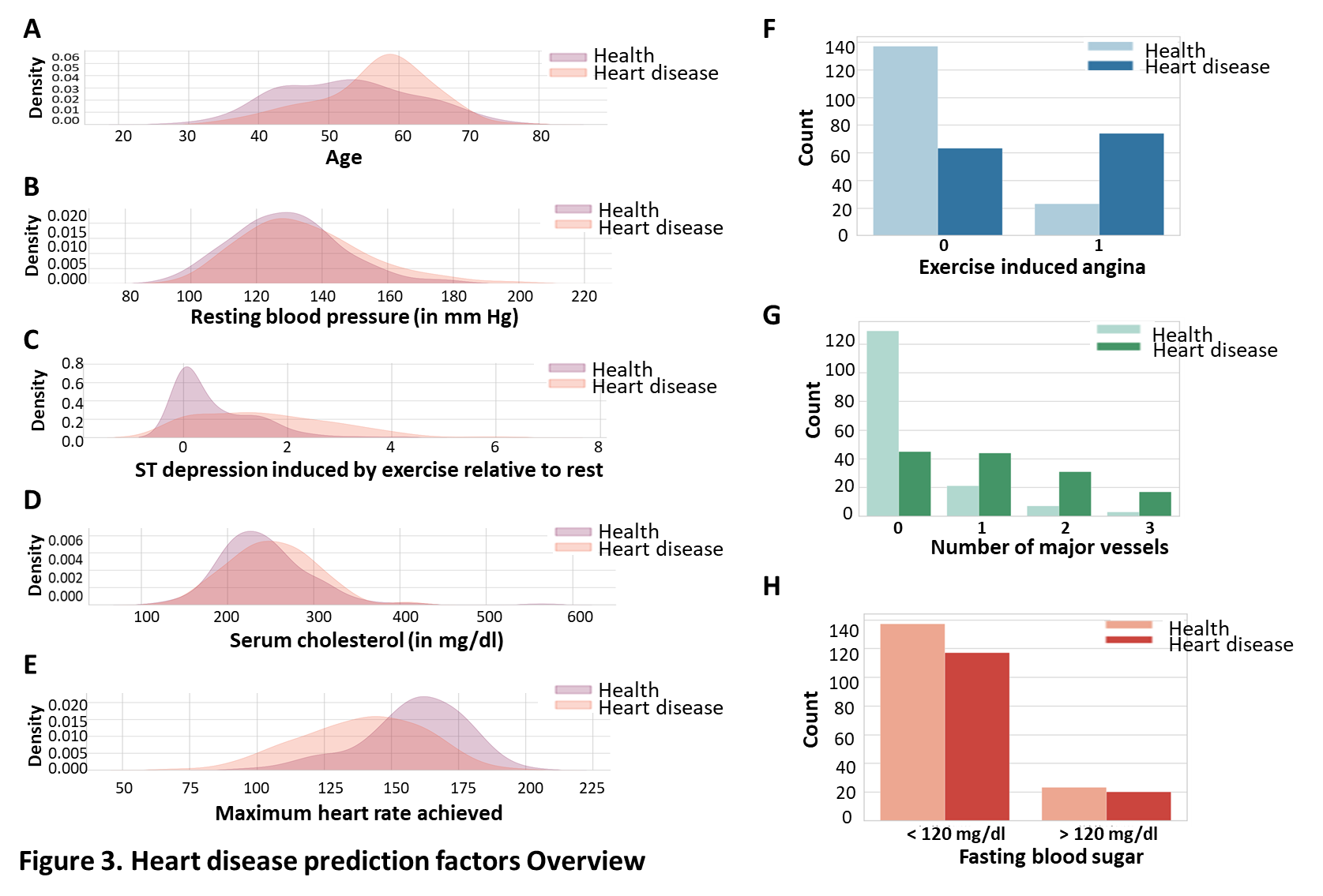
To support our prediction model, meta-analysis using python (3) will be performed regarding on heart diseases from PubMed, the predicted risk factor for heart disease will be used to value the prediction model that created inabove.

*Please see code for more details.*

# Result

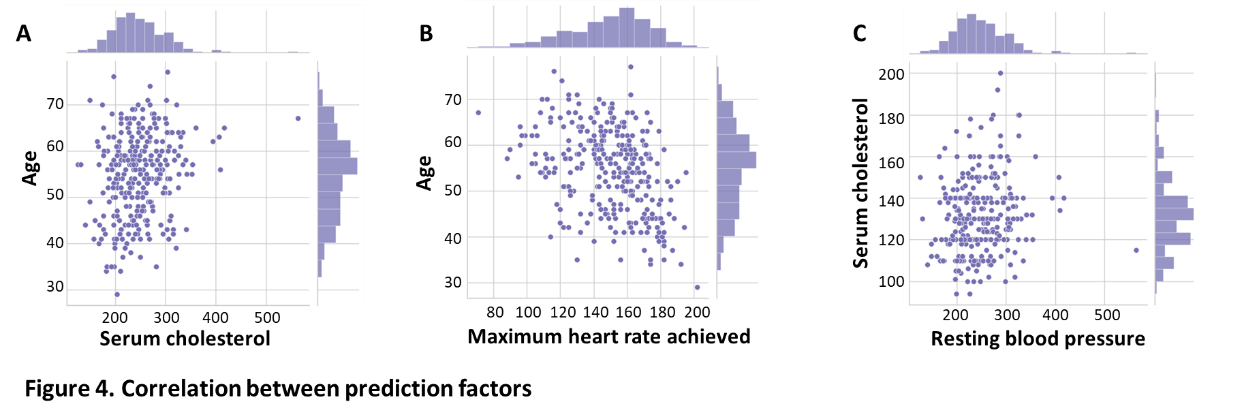
## descriptive analysis

### prediction factor vs. target

Figure 1 shows the overview of major prediction factor in healthy and heart disease. Heart disease more likely occurs in elder people (Figure 1A), and those people showed increased resting blood pressure (Figure 1B); increased ST depression induced by exercise (Figure 1C), and serum cholesterol (Figure 1D), while showed a lower maximum heart rate achieved(Figure 1E). In addition, it shows that heart disease would more likely have exercise induced angina (Figure 1F), and the affect vessels number are relative increased in heart disease patients (Figure 1G), while there is no clear heart disease preference regarding fasting blood sugar (Figure 1H).

### Correlations between age & serum cholesterol & resting blood pressure

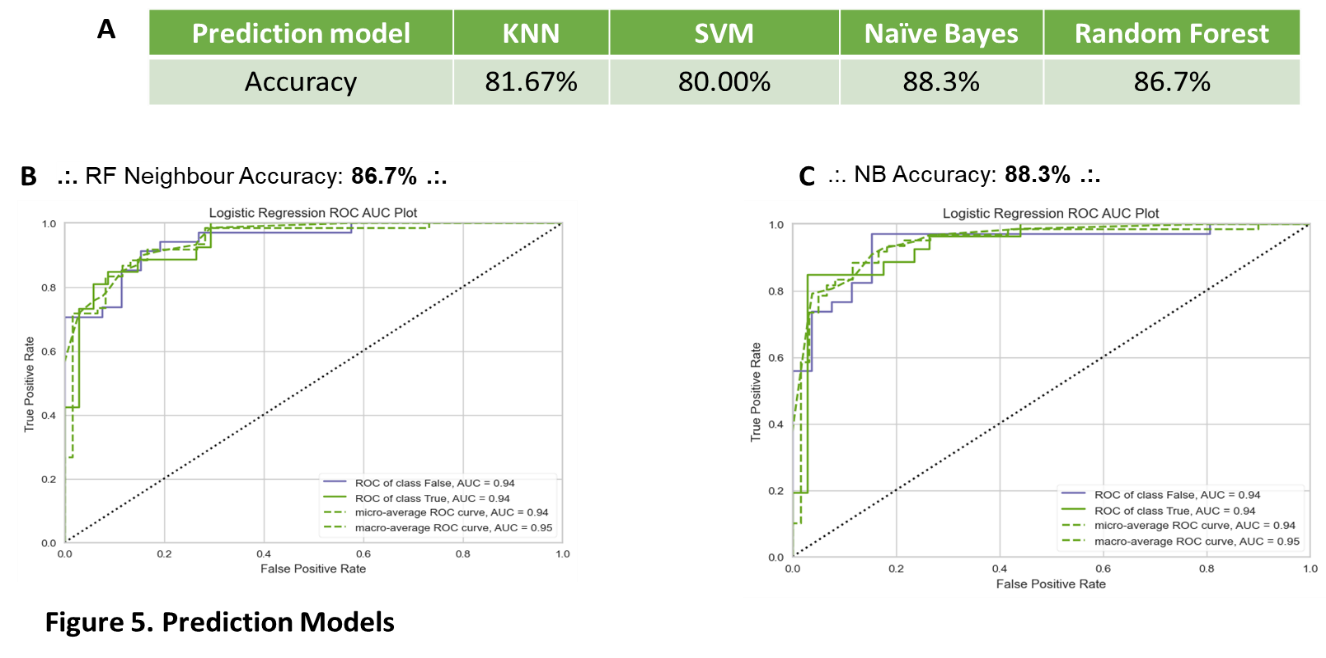
Figure 2 shows the correlation between prediction factors. Figure 2A shows no significant correlation between age and cholesterol; Figure 2B shows a negative correlation between age and maximum heart rate achieved, as the younger people show a greater maximum heart rate achieved. Figure 2C shows that no clear correlation between serum cholesterol and resting blood pressure.



## Explore prediction model

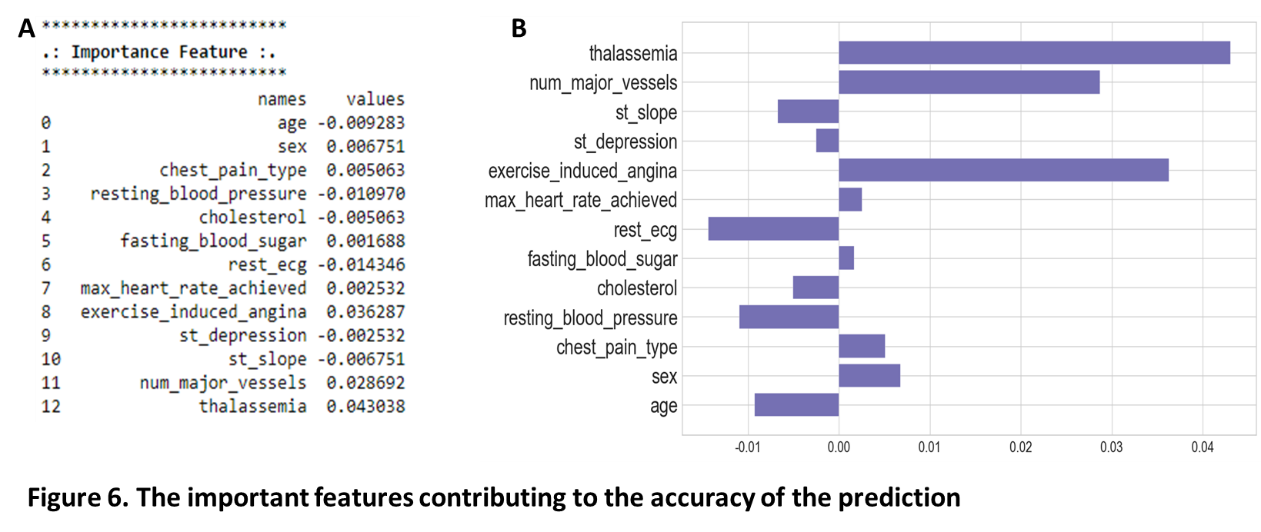
### Prediction models

Four prediction models were performed with all 13 prediction factors. The accuracy score were shown in Figure 3A, KNN (81.67%), SVM (80.00%), Naïve Bayes(88.30%), and Random Forest (86.7%). Figure 3B-C show the ROC curve of Random Forest (Figure 3B) and Naïve Bayes (Figure 3C).



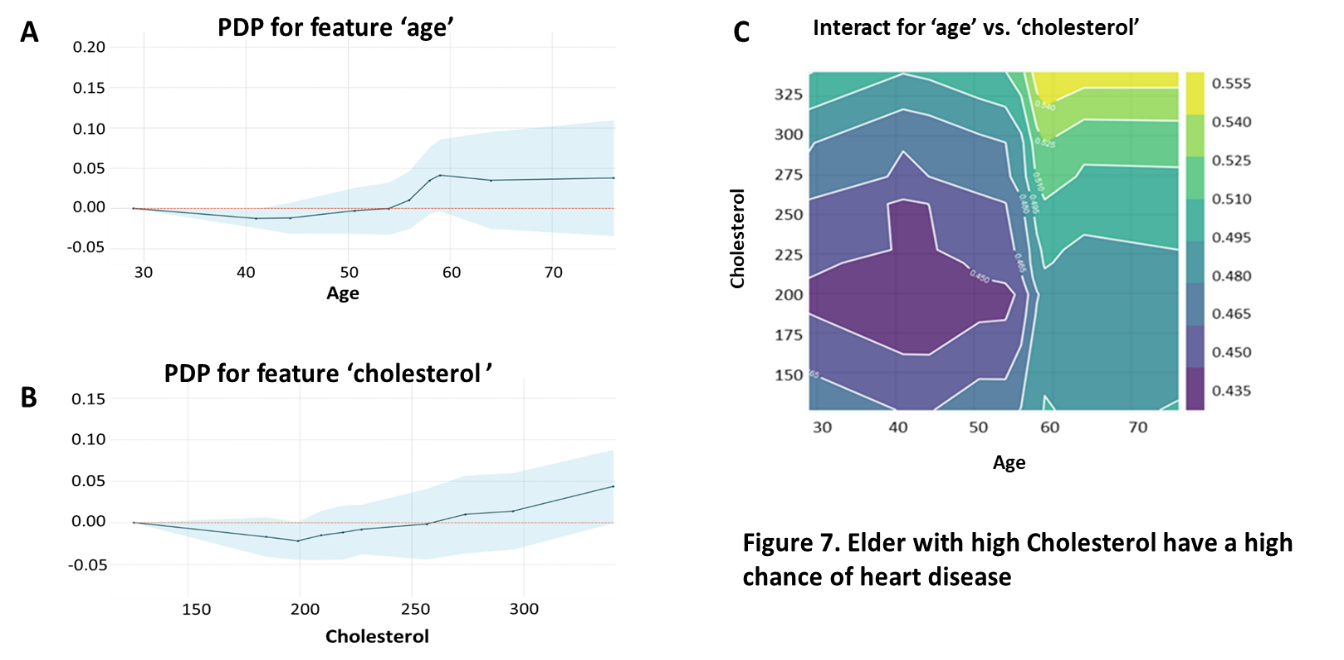
### Important features

I then check which feature(s) contribute the most to the model prediction, As shown in Figure 4A-B, the top three of the important features include Thalassemia, exercise\_induced\_angina, and number\_major\_vessels.

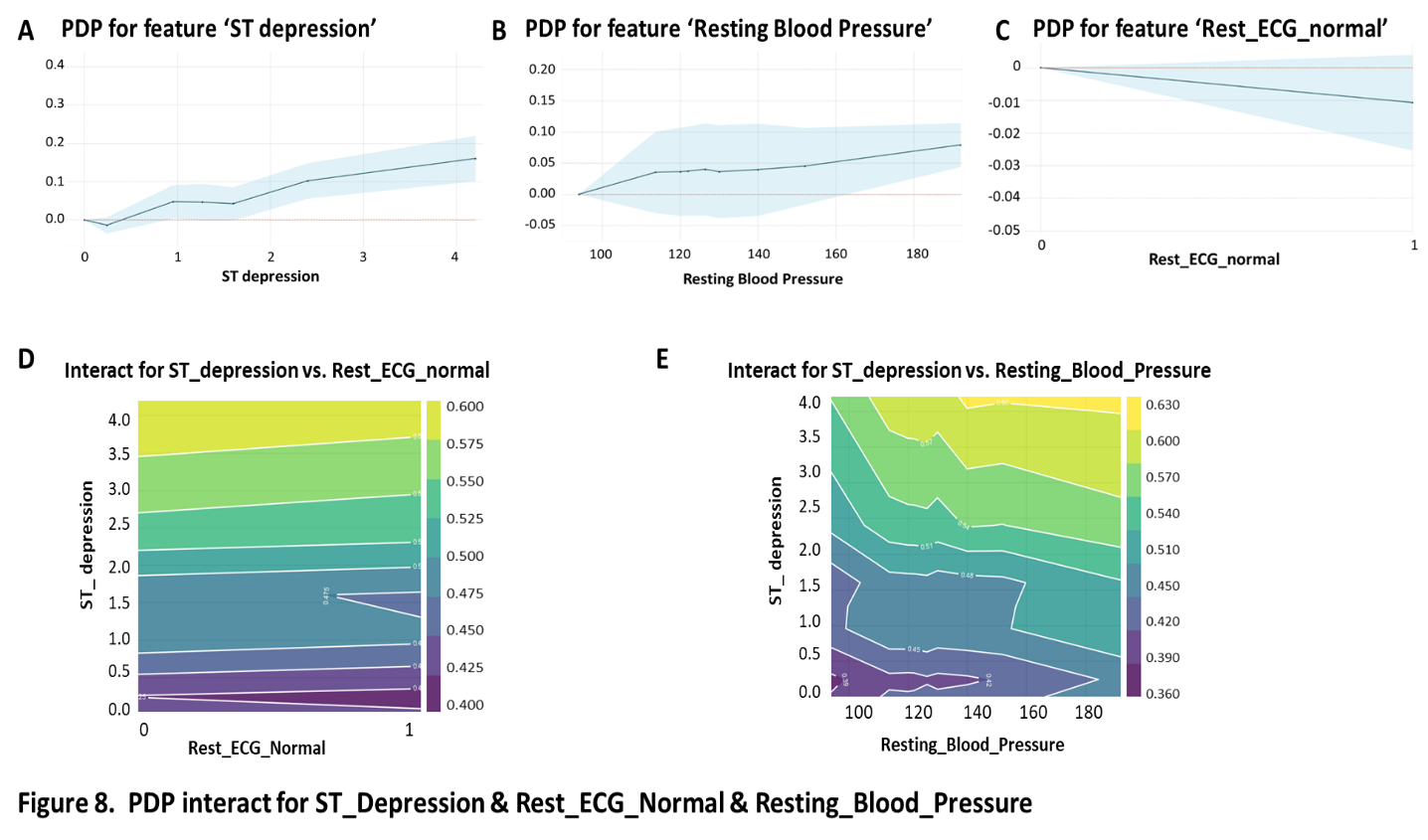


### PDP interact for Age & Cholesterol feature

The heart disease shows an increase in probability the higher age (Figure7A) and cholesterol (Figure 7B) it goes. Increase in heart disease probability at age 55, and higher cholesterol would more possible induce heart disease. For age below 55, people who has cholesterol between 150~250 is very likely to develop heart disease.



### PDP interact for ST\_Depression & Rest\_ECG\_Normal & Resting\_Blood\_Pressure

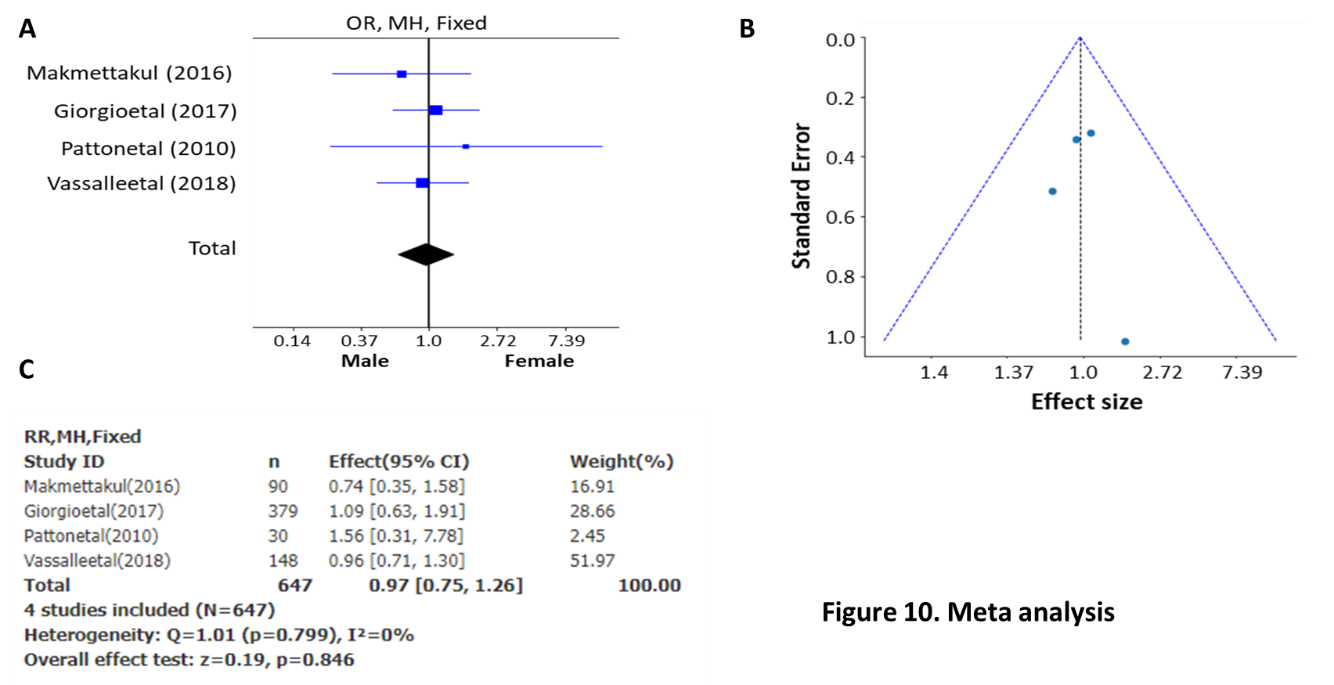


### PDP interact for Thalassemia & Sex\_Male

Thalassemia has no preference for gender. While study showed that there is gender differences in the bone disease in thalassemia, given that thalassemia is great risk factor for heart disease, I therefore, check whether gender difference affect heart disease in thalassemia(4). Figure 9 shows the PDP interact for Thalassemia and gender, male with thalassemia showed an increased possibility to have heart disease.

## meta-analysis

Figure shows the meta-analysis results from the selected published papers. Figure 7A shows the forest plot, indicating no gender difference in Thalassemia induced heart disease. Figure 7B shows the funnel plot, indicating no publication bias.



# Conclusion

Heart disease is the leading cause of death for men and women. One person dies every 34 seconds in United States from heart disease. In this study, four data sets from VA, Switzerland, Hungarian, and Cleveland were used, I observed that heart disease more possible occur in elder people, and those people showed increased ST depression induced by exercise, increased resting blood pressure, serum cholesterol, while showed a lower maximum heart rate achieved. In this data set, no significant correlation pattern between age and serum cholesterol, however, I observed a negative correlation between age and maximum heart rate achieve, that is, as age increased, the maximum heart rate is reduced.

Four different machine learning models such as Naïve Bayes, SVM, KNN, and Random Forest for prediction of heart disease were performed to predict heart disease. In all cases, they performed well, the only exception is SVM (80.00%), and Naïve Bayes shows the highest test accuracy (88.30%). And Thalassemia was identified as the most important feature for prediction model, however, even though the model prediction indicates that male with Thalassemia is more possible to develop heart disease, the meta-analysis on recent studies showed that there is no gender difference in Thalassemia related heart disease.

It has to be noted that 1) the data was small, which may provide a bias conclusion. 2) the data were collected and published in 1987, which is relative old, as the cardiopathy diagnoses are much more advanced nowadays, therefore the prediction factors showed in the data set have less value. 3) there are many types of heart disease, and major one is coronary artery disease, and different type of heart disease many present different symptoms, however, the “target” field in the dataset is general heart disease, which may mask the rare heart disease type. 4) the majority record in above data set is from Cleveland (America), therefore it may raise the reginal difference. Taken together, even though the limitation of the study, the study offers a new sight into the heart disease prediction model.

# Reference

1. Poldervaart JM, Langedijk M, Backus BE, Dekker IMC, Six AJ, Doevendans PA, Hoes AW, Reitsma JB. Comparison of the GRACE, HEART and TIMI score to predict major adverse cardiac events in chest pain patients at the emergency department. Int J Cardiol. 2017;227:656-61. Epub 2016/11/05. doi: 10.1016/j.ijcard.2016.10.080. PubMed PMID: 27810290.

2. Lee YH. An overview of meta-analysis for clinicians. Korean J Intern Med. 2018;33(2):277-83. Epub 2017/12/27. doi: 10.3904/kjim.2016.195. PubMed PMID: 29277096; PMCID: PMC5840596.

3. Masoumi S, Shahraz S. Meta-analysis using Python: a hands-on tutorial. BMC Med Res Methodol. 2022;22(1):193. Epub 2022/07/13. doi: 10.1186/s12874-022-01673-y. PubMed PMID: 35820854; PMCID: PMC9275021.

4. Kyriakou A, Savva SC, Savvides I, Pangalou E, Ioannou YS, Christou S, Skordis N. Gender differences in the prevalence and severity of bone disease in thalassaemia. Pediatr Endocrinol Rev. 2008;6 Suppl 1:116-22. Epub 2009/04/11. PubMed PMID: 19337164.

# Supplementary code

"""

IST652 scripting for data analysi

Final project by Jie Wang

09/05/2022

Project Name: Machine models for Prediction of Heart disease

Backgroud:

In the United States, heart diseases cause more than 859,000 people deaths each year and $216 billion in health care system cost.

Up TO 6.3% of ER visits are related to chest pain. An urgent question in these patients is whether they have an acute coronary syndromes (ACS), as any delay in diagnosis and treatment can have a negative impact on their prognosis.

If patients at low risk for ACS could be recognized early in the diagnostic process, it has the potential to reduce patient burden, length of stay at the ED, frequency of hospitalization and costs.

Data:

^^^https://archive.ics.uci.edu/

^^^^^^prcessed.switzeland.data;

^^^^^^processed.hungarian.data;

^^^^^^processed.cleveland.data;

^^^^^^processed.va.data

^^^https://pubmed.ncbi.nlm.nih.gov/

Aims:

^^^Aim 1: To explore general profiles of healthy control vs. heard disease patients.

^^^Aim 2: To explore a prediction model

^^^Aim 3: meta analysis using pubmed data

Methods: descriptive analysis and explore analysis by Python

"""

#########################

########################

###### set directory

%pwd

%cd C:\Documents\Syracuse\Course 5\_Processing\_IST652 Scripting for Data Analysis\Homework\9. Project 0203 Sep 09\raw data

#########################

########################

###### descritive analysis

##export libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import warnings

import os

import yellowbrick

import pickle

from matplotlib.collections import PathCollection

from statsmodels.graphics.gofplots import qqplot

from sklearn.preprocessing import MinMaxScaler

from sklearn.model\_selection import train\_test\_split# Import train\_test\_split function

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier

from sklearn.svm import SVC

from sklearn.naive\_bayes import GaussianNB

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, AdaBoostClassifier, ExtraTreesClassifier

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

from xgboost import XGBClassifier

from yellowbrick.classifier import PrecisionRecallCurve, ROCAUC, ConfusionMatrix

from yellowbrick.style import set\_palette

from yellowbrick.model\_selection import LearningCurve, FeatureImportances

from yellowbrick.contrib.wrapper import wrap

from sklearn.tree import DecisionTreeClassifier

from sklearn import metrics

from sklearn.datasets import make\_classification

from sklearn.inspection import permutation\_importance

from matplotlib import pyplot

from pdpbox import pdp

## libraries setting

warnings.filterwarnings('ignore')

plt.rcParams['figure.dpi'] = 100

set\_palette('dark')

sns.set\_style('whitegrid')

## color setting

sns.color\_palette('tab10')

############

####reading data

##reading basic information of the data

read information about the data

with open('heart-disease.names') as f:

print(f.read())

### read cleveland data, hungarian data, switzerland data, and va data

data\_cleveland = pd.read\_csv('processed.cleveland.data', names =['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal','target'])

data\_hungarian = pd.read\_csv('processed.hungarian.data', names =['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal','target'])

data\_switzeland = pd.read\_csv('processed.switzerland.data', names =['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal','target'])

data\_va = pd.read\_csv('processed.va.data', names =['age','sex','cp','trestbps','chol','fbs','restecg','thalach','exang','oldpeak','slope','ca','thal','target'])

##show the cleveland data

data\_cleveland

##show the hungarian data

data\_hungarian

##show the switzeland data

data\_switzeland

##show the va data

data\_va

##combine all four data sets

data\_all = [data\_cleveland, data\_hungarian, data\_switzeland, data\_va]

data\_all = pd.concat(data\_all)

data\_all

###################################

#### clean the data set

### drop the na value record and show the information about the data

data\_new = data\_all.dropna()

data\_new.info()

### show the value with ? character

data\_new.eq('?')

### remove the records with ? character

data = data\_new[~data\_new.eq('?').any(1)]

data

#### Print new Dataset info

print('\033[1m'+'.: Dataset Info :.'+'\033[0m')

print('\*' \* 40)

print('Total Rows:'+'\033[1m', data.shape[0])

print('\033[0m'+'Total Columns:'+'\033[1m', data.shape[1])

print('\033[0m'+'\*' \* 40)

print('\n')

# --- Print Dataset Detail ---

print('\033[1m'+'.: Dataset Details :.'+'\033[0m')

print('\*' \* 40)

data.info(memory\_usage = False)

#### fix the data types for the columns before analysis performed

## sex, cp, fbs, restecg, exang, slope, ca, thal should be object

## age, threstbps,chol, thalach, should be int

## oldpeak should be float

list1 = ['sex', 'cp']

list2 = ['age', 'trestbps', 'chol','thalach']

list3 = ['trestbps','chol','fbs','restecg','thalach','exang','slope']

data[list3] = data[list3].astype(int)

data['thal'] = data['thal'].astype(float)

data['ca'] = data['ca'].astype(float)

data[list1] = data[list1].astype(object)

data[list2] = data[list2].astype(int)

data['oldpeak'] = data['oldpeak'].astype(float)

data['thal'] = data['thal'].astype(int)

data['ca'] = data['ca'].astype(int)

lst=['fbs', 'restecg', 'exang', 'slope', 'ca', 'thal']

data[lst] = data[lst].astype(object)

data.info()

### view data

data

#### fix target value

### as mentioned in the data information showed above, target 0 indicates healthy, target 1,2,3,4 indicates heart disease

data['target'] = data['target'].replace([2,3,4],1)

#### delete the duplicated indix records

data = data[~data.index.duplicated()]

data

####descriptive stat

data.select\_dtypes(exclude='object').describe().T

###########################################

############ Aim 1 ###############

###########################################

############################

############################

##### Descriptive analysis

##### visulization

### I. Oldpeak (ST depression) in healthy and heart disease

fig = sns.FacetGrid(data, hue="target",aspect=4, palette='rocket')

fig.map(sns.kdeplot,'oldpeak',shade= True)

plt.legend(labels=['Healthy' , 'Heart disease'])

### 2. Oldpeak (ST trestbps) in healthy and heart disease

fig = sns.FacetGrid(data, hue="target",aspect=4, palette='rocket')

fig.map(sns.kdeplot,'trestbps',shade= True)

plt.legend(labels=['Healthy' , 'Heart disease'])

### 3. age in healthy and heart disease

fig = sns.FacetGrid(data, hue="target",aspect=4, palette='rocket')

fig.map(sns.kdeplot,'age',shade= True)

plt.legend(labels=['Healthy' , 'Heart disease'])

# 4. Serum cholesterol (in mg/dl) (chol) in healthy and heart disease

fig = sns.FacetGrid(data, hue="target",aspect=4, palette='rocket')

fig.map(sns.kdeplot,'chol',shade= True)

plt.legend(labels=['Healthy' , 'Heart disease'])

# 5. Maximum heart rate achieved (thalach) in healthy and heart disease

fig = sns.FacetGrid(data, hue="target",aspect=4, palette='rocket')

fig.map(sns.kdeplot,'thalach',shade= True)

plt.legend(labels=['Healthy' , 'Heart disease'])

### check correlation between chol and trestbps

sns.jointplot(data=data,

x='chol',

y='trestbps',

kind='scatter',

cmap='PuBu'

)

# check correlation between chol and age

sns.jointplot(data=data,

x='chol',

y='age',

kind='scatter',

cmap='PuBu'

)

# check correlation between thalach and age

sns.jointplot(data=data,

x='thalach',

y='age',

kind='scatter',

)

### check disease in no\_angina and with angina

fig = sns.countplot(x = 'exang', data = data, hue = 'target', palette='Blues')

plt.legend(['Healthy', 'Heart disease'])

### check disease Number of major vessels colored by fluoroscopy

fig = sns.countplot(x = 'ca', data = data, hue = 'target', palette='BuGn')

plt.legend(['Healthy', 'Heart disease'])

### check Fasting blood sugar in healthy and disease

fig = sns.countplot(x = 'fbs', data = data, hue = 'target', palette='Reds')

plt.legend(['Healthy', 'Sick'])

fig.set\_xticklabels(labels=[ 'low blood sugar','high blood sugar'])

###########################################

############ Aim 2 ###############

###########################################

###########################################

###########################################

############## Prediction models

### prepare data set

X=data.drop(columns='target')

y=data['target']

### normalizing

scaler=MinMaxScaler()

X=scaler.fit\_transform(X)

####define the function for learning curve

def plot\_LearningCurv(model):

loglc = LearningCurve(model, title='Logistic Regression Learning Curve')

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

loglc.finalize()

####define the function for learning curve

def plot\_RoC(model):

logrocauc = ROCAUC(model, classes=['False', 'True'],

title='Logistic Regression ROC AUC Plot')

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

logrocauc.score(X\_test, y\_test)

logrocauc.finalize()

plt.show()

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

### logistic regression

logreg=LogisticRegression()

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

y\_pred=logreg.predict(X\_test)

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

plot\_RoC(logreg)

plot\_LearningCurv(logreg)

#### KNN

from sklearn.neighbors import KNeighborsClassifier

knn\_model=KNeighborsClassifier(n\_neighbors=4)

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

y\_knn\_pred=knn\_model.predict(X\_test)

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

print('.:. K-Nearest Neighbour Accuracy:'+'\033[1m {:.2f}%'.format(KNNAcc\*100)+' .:.')

plot\_RoC(knn\_model)

plot\_LearningCurv(knn\_model)

### SVM

from sklearn.svm import SVC

svm\_model=SVC(probability=True)

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

y\_svm\_pred=svm\_model.predict(X\_test)

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

print('.:. SVM Neighbour Accuracy:'+'\033[1m {:.2f}%'.format(SVMacc\*100)+' .:.')

plot\_RoC(svm\_model)

plot\_LearningCurv(svm\_model)

### Naive Bays

from sklearn.naive\_bayes import GaussianNB

NB\_model=GaussianNB(var\_smoothing=0.08)

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

y\_pred\_NB=NB\_model.predict(X\_test)

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

print('.:. NB Accuracy:'+'\033[1m {:.1f}%'.format(NBacc\*100)+' .:.')

plot\_RoC(NB\_model)

plot\_LearningCurv(NB\_model)

### Random Classifier

from sklearn.ensemble import RandomForestClassifier

RF\_model = RandomForestClassifier(n\_estimators=1000, random\_state=1, max\_leaf\_nodes=20, min\_samples\_split=15)

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

y\_pred\_RF = RF\_model.predict(X\_test)

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

print('.:. RF Accuracy:'+'\033[1m {:.1f}%'.format(RFacc\*100)+' .:.')

plot\_RoC(RF\_model)

plot\_LearningCurv(RF\_model)

#### permutation feature importance with knn for classification

# define dataset

# define the model

model = KNeighborsClassifier()

# fit the model

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

# perform permutation importance

results = permutation\_importance(model, X\_train, y\_train, scoring='accuracy')

# get importance

importance = results.importances\_mean

# summarize feature importance

for i,v in enumerate(importance):

print('Feature: %0d, Score: %.5f' % (i,v))

# plot feature importance

pyplot.bar([x for x in range(len(importance))], importance)

pyplot.show()

#### fix the plot, and make it clearer

## check the importance of each feature

names = ['age', 'sex', 'chest\_pain\_type', 'resting\_blood\_pressure', 'cholesterol', 'fasting\_blood\_sugar', 'rest\_ecg', 'max\_heart\_rate\_achieved',

'exercise\_induced\_angina', 'st\_depression', 'st\_slope', 'num\_major\_vessels', 'thalassemia']

importance\_value = {'names': names, 'values': importance}

df = pd.DataFrame(data=importance\_value)

# --- print feature importance score ---

print('\*' \* 25)

print('\033[1m'+'.: Importance Feature :.'+'\033[0m')

print('\*' \* 25)

print(df)

###print the importance score plot

fig, ax = plt.subplots(figsize =(16, 9))

ax.barh(names, importance)

ax.tick\_params(axis='x', labelsize=18)

ax.tick\_params(axis='y', labelsize=26)

################################

###################### alternative prediction model

dt =data

dt.columns = ['age', 'sex', 'chest\_pain\_type', 'resting\_blood\_pressure', 'cholesterol', 'fasting\_blood\_sugar', 'rest\_ecg', 'max\_heart\_rate\_achieved',

'exercise\_induced\_angina', 'st\_depression', 'st\_slope', 'num\_major\_vessels', 'thalassemia', 'target']

###### reset the name, make the table more explainable

dt['sex'][dt['sex'] == 0] = 'female'

dt['sex'][dt['sex'] == 1] = 'male'

dt['chest\_pain\_type'][dt['chest\_pain\_type'] == 1] = 'typical angina'

dt['chest\_pain\_type'][dt['chest\_pain\_type'] == 2] = 'atypical angina'

dt['chest\_pain\_type'][dt['chest\_pain\_type'] == 3] = 'non-anginal pain'

dt['chest\_pain\_type'][dt['chest\_pain\_type'] == 4] = 'asymptomatic'

dt['fasting\_blood\_sugar'][dt['fasting\_blood\_sugar'] == 0] = 'lower than 120mg/ml'

dt['fasting\_blood\_sugar'][dt['fasting\_blood\_sugar'] == 1] = 'greater than 120mg/ml'

dt['rest\_ecg'][dt['rest\_ecg'] == 0] = 'normal'

dt['rest\_ecg'][dt['rest\_ecg'] == 1] = 'ST-T wave abnormality'

dt['rest\_ecg'][dt['rest\_ecg'] == 2] = 'left ventricular hypertrophy'

dt['exercise\_induced\_angina'][dt['exercise\_induced\_angina'] == 0] = 'no'

dt['exercise\_induced\_angina'][dt['exercise\_induced\_angina'] == 1] = 'yes'

dt['st\_slope'][dt['st\_slope'] == 1] = 'upsloping'

dt['st\_slope'][dt['st\_slope'] == 2] = 'flat'

dt['st\_slope'][dt['st\_slope'] == 3] = 'downsloping'

dt['thalassemia'][dt['thalassemia'] == 1] = 'normal'

dt['thalassemia'][dt['thalassemia'] == 2] = 'fixed defect'

dt['thalassemia'][dt['thalassemia'] == 3] = 'reversable defect'

######## dummies the table

dt['num\_major\_vessels'] = dt['num\_major\_vessels'].astype('int')

dt = pd.get\_dummies(dt, drop\_first=True)

dt.head()

#### split the data into test and train set

X\_train11, X\_test11, y\_train11, y\_test11 = train\_test\_split(dt.drop('target', 1), dt['target'], test\_size = .2, random\_state=10)

#### predict with random forest classification

model11 = RandomForestClassifier(max\_depth=5)

model11.fit(X\_train11, y\_train11)

estimator = model11.estimators\_[1]

feature\_names = [i for i in X\_train11.columns]

y\_train\_str = y\_train11.astype('str')

y\_train\_str[y\_train\_str == '0'] = 'no disease'

y\_train\_str[y\_train\_str == '1'] = 'disease'

y\_train\_str = y\_train\_str.values

y\_predict = model11.predict(X\_test11)

y\_pred\_quant = model11.predict\_proba(X\_test11)[:, 1]

y\_pred\_bin = model11.predict(X\_test11)

#####################

##########print important keys for the model

#### resting\_blood\_pressure for disease prediction

base\_features = dt.columns.values.tolist()

base\_features.remove('target')

feat\_name = 'resting\_blood\_pressure'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

#### age for disease prediction

feat\_name = 'age'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

## PDP for feature st\_depression

feat\_name = 'st\_depression'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

## PDP for feature cholesterol

feat\_name = 'cholesterol'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

## PDP for feature resting\_blood\_pressure

feat\_name = 'resting\_blood\_pressure'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

## PDP for feature rest\_ecg\_normal

feat\_name = 'rest\_ecg\_normal'

pdp\_dist = pdp.pdp\_isolate(model=model11, dataset=X\_test11, model\_features=base\_features, feature=feat\_name)

pdp.pdp\_plot(pdp\_dist, feat\_name)

plt.show()

### interaction between rest\_ecg\_normal & st\_depression

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['rest\_ecg\_normal', 'st\_depression'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['rest\_ecg\_normal', 'st\_depression'], plot\_type='contour')

plt.show()

### interaction between rest\_ecg\_left ventricular hypertrophy & st\_depression

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['rest\_ecg\_left ventricular hypertrophy', 'st\_depression'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['rest\_ecg\_left ventricular hypertrophy', 'st\_depression'], plot\_type='contour')

plt.show()

### interaction between sex\_male & thalassemia\_7

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['sex\_male', 'thalassemia\_7'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['sex\_male', 'reversable defect'], plot\_type='contour')

plt.show()

# interaction between sex\_male & thalassemia\_reversable defect

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['sex\_male', 'thalassemia\_reversable defect'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['sex\_male', 'thalassemia\_normal'], plot\_type='contour')

plt.show()

# interaction between age & cholesterol

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['age', 'cholesterol'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['age', 'cholesterol'], plot\_type='contour')

plt.show()

# interaction between resting\_blood\_pressure & st\_depression

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['resting\_blood\_pressure', 'st\_depression'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['resting\_blood\_pressure', 'st\_depression'], plot\_type='contour')

plt.show()

# interaction between resting\_blood\_pressure & rest\_ecg\_normal

inter1 = pdp.pdp\_interact(model=model11, dataset=X\_test11, model\_features=base\_features, features=['resting\_blood\_pressure', 'rest\_ecg\_normal'])

pdp.pdp\_interact\_plot(pdp\_interact\_out=inter1, feature\_names=['resting\_blood\_pressure', 'rest\_ecg\_normal'], plot\_type='contour')

plt.show()

###########################################

############ Aim 3 ###############

###########################################

###############################################

##############################################

###################get information from pubmed

from Bio import Entrez

from xml.etree.ElementTree import iterparse

import csv

import time

import datetime

import urllib.request as libreq

import feedparser

from unidecode import unidecode

from Bio import Entrez

def search(query):

Entrez.email = 'jwang326@syr.edu'

handle = Entrez.esearch(db='pubmed',

sort='relevance',

retmax='20', ## show 20 only, for view purpose

retmode='xml',

term=query)

results = Entrez.read(handle)

return results

def fetch\_details(id\_list):

ids = ','.join(id\_list)

Entrez.email = 'jwang326@syr.edu'

handle = Entrez.efetch(db='pubmed',

retmode='xml',

id=ids)

results = Entrez.read(handle)

return results

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

results = search('thalassemia')

id\_list = results['IdList']

papers = fetch\_details(id\_list)

for i, paper in enumerate(papers['PubmedArticle']):

print("{}) {}".format(i+1, paper['MedlineCitation']['Article']['ArticleTitle']))

##########################

###############meta analysis

import PythonMeta as PMA

import os

def showstudies(studies, dtype):

text= "%-13s %-18s %-20s \n"%('study ID', 'male group', 'female group')

text += "%-10s % -10s %-10s % -10s % -10s \n"%(' ', 'e1', 'n1','e2','n2')

for i in range(len(studies)):

text += "%-10s % -10s %-10s % -10s % -10s \n" %(

studies[i][4],

str(studies[i][0]),

str(studies[i][1]),

str(studies[i][2]),

str(studies[i][3])

)

return text

def showresults(rults):

text = '%-12s %-8s %-13s %-2s'%('study ID', 'sample size', 'ES[95% CI]', 'Weight(%)\n')

for i in range(1, len(rults)):

text +='%-15s %-6d %-4.2f[%.2f,%.2f]\t%6.2f\n' %(

rults[i][0],

rults[i][5],

rults[i][1],

rults[i][3],

rults[i][4],

100\*(rults[i][2]/rults[0][2])

)

text+= "%-15s %-6d %-4.2f[%.2f %.2f]\t%6d\n"%(

rults[0][0],

rults[0][5],

rults[0][1],

rults[0][3],

rults[0][4],

100

)

text += "total %d studys(N=%d)\n"%(len(rults)-1, rults[0][5])

text +='heterogeneity: Tau\u00b2 =%.3f '%(rults[0][12]) if not rults[0][12] == None else "heterogeneity: "

text +='Q(Chisquare) =%.2f(p=%s); I\u00b2 = %s\n'%(

rults[0][7],

rults[0][8],

str(round(rults[0][9],2))+"%")

text += "Overall effect test: z=%.2f, p=%s\n"%(rults[0][10],rults[0][11])

return text

def main(sample, settings):

d = PMA.Data()

m = PMA.Meta()

f = PMA.Fig()

#should always tell the datatype first!!!

d.datatype = settings['datatype']#set data type, 'CATE' for binary data or 'CONT' for continuous data

studies = d.getdata(d.readfile('binary studies.csv')) #load data

# m.subgroup=d.dubgroup #get data from a data file, see examples of data files

m.nototal=d.nototal

print(showstudies(studies, d.datatype)) #show studies

m.datatype= d.datatype #set data type for meta-analysis calculating

m.models = settings['models'] #set effect models: 'Fixed' or 'Random'

m.algorithm = settings['algorithm'] #set algorithm, based on datatype and effect size

m.effect = settings['effect'] #set effect size:RR/OR/RD for binary data; SMD/MD for continuous data

results = m.meta(studies, nosubgrp =True) #performing the analysis

print(m.models + ' ' + m.algorithm + " " + m.effect) #show results table

print(showresults(results))

f.size = [10,8]

f.nototal =False

f.dpi =200

f.forest(results).show() #show forest plot

f.funnel(results).show() #show funnel plot

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

d = PMA.Data()

samp\_cont=d.readfile('binary studies.csv')

settings = {

'datatype':'CATE',

'models':'Fixed',

'algorithm':'MH',

'effect':'OR'}

main(samp\_cont, settings)

############End