Heart Disease Project

This project was the result of team effort developed while I completed a Machine Learning Foundations course by the University of Calgary.

The application was built using Python. It uses the following libraries:

- Pandas;
- NumPy;
- Sklearn;
- Tkinter;
- Matplotlib;

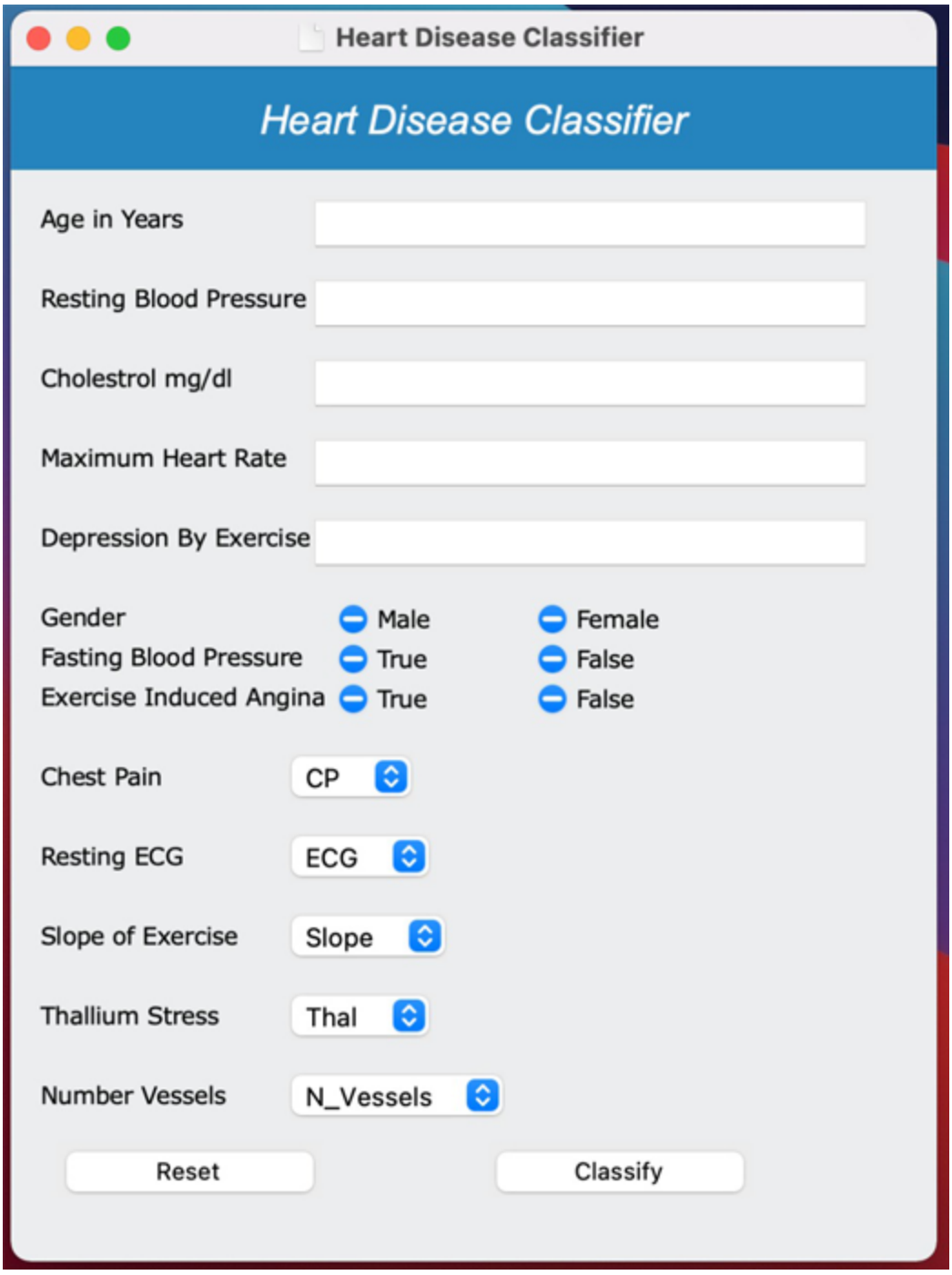
The purpose of the project was to analyze the UCI Heart Disease dataset to develop a classification Machine Learning model able to distinguish between the symptoms of heart disease patients.

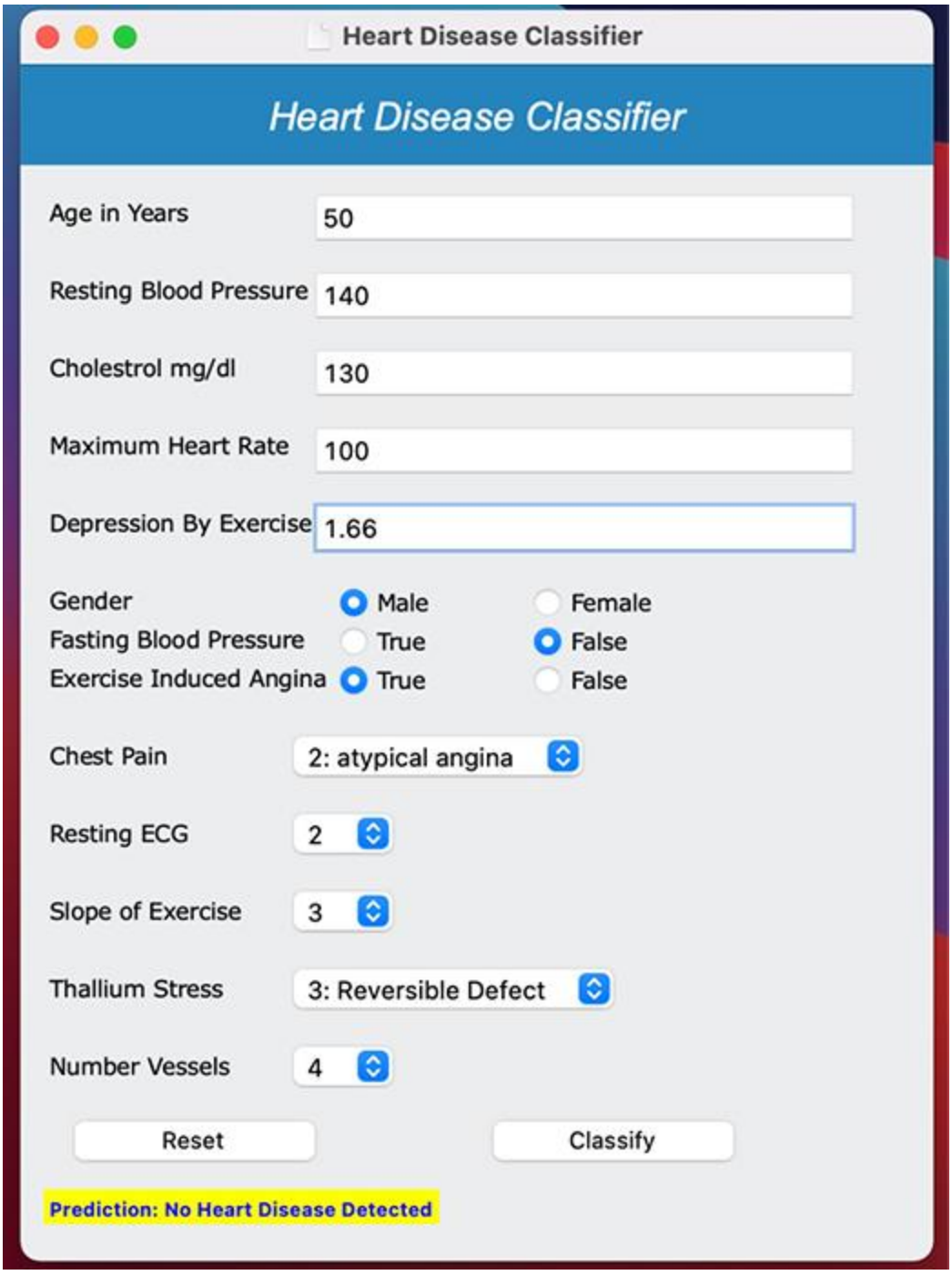
It is basically divided into the following phases:

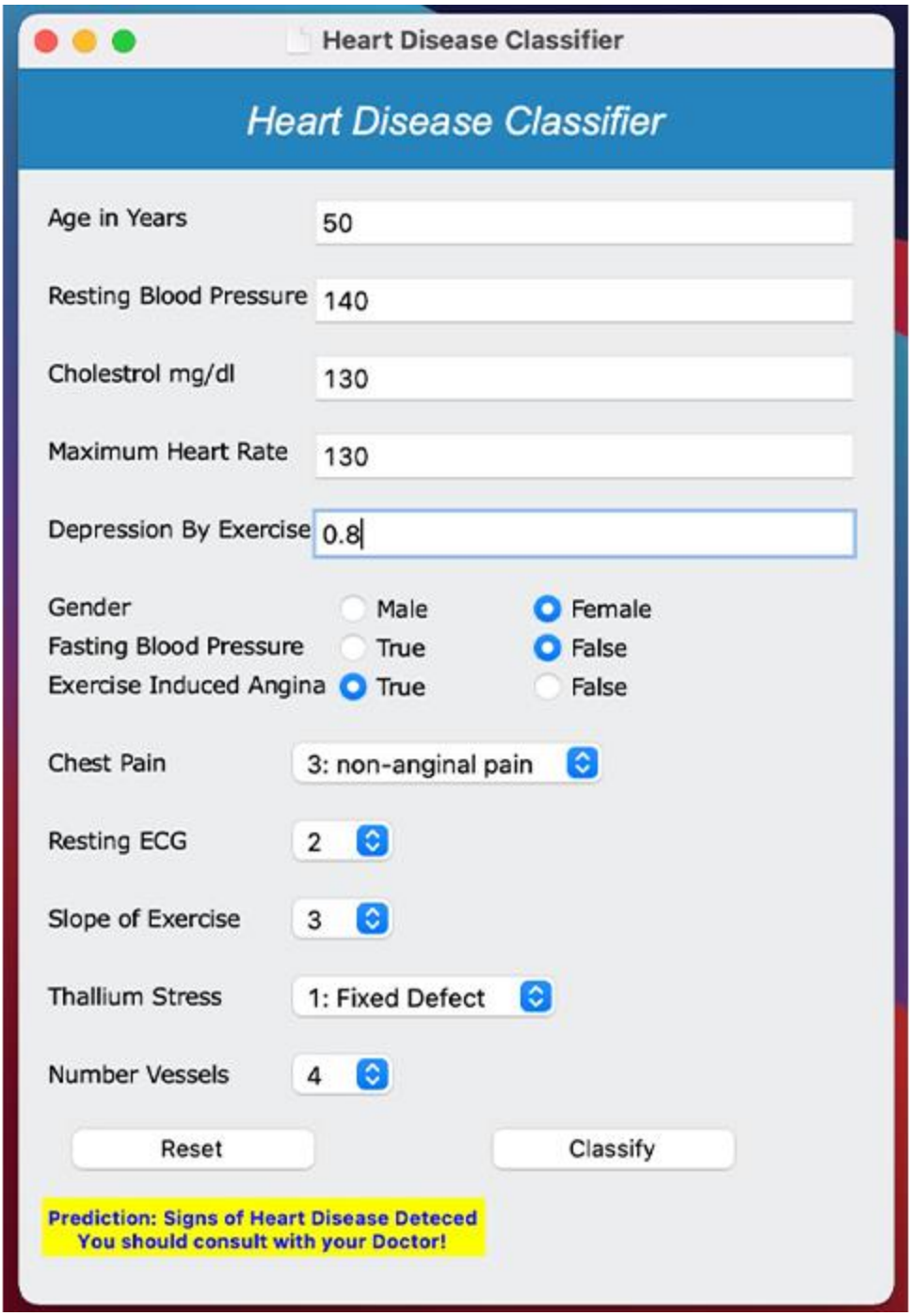
- Extracting the data from the dataset.
- Cleaning the data from unfit information.
- Transforming the data into a computer/machine readable format.
- Using the data to develop a machine learning application.
- building a Desktop Graphical User Interface to classify patient data on-demand.

Once it's built, it collects input data from the user, and applies it into the machine learning to generate the appropriate output response.

Below are some pictures that show how the application is built and its functionality. I have also pasted the source code.


Heart Disease Classifier
Heart Disease Classifier
Age in Years
Resting Blood Pressure
Cholestrol mg/dl
Maximum Heart Rate
Depression By Exercise
Gender
Male
Female
Fasting Blood Pressure
True
False
Exercise Induced Angina
True
False
Chest Pain
CP
Resting ECG
ECG
Slope of Exercise
Slope
Thallium Stress
Thal
Number Vessels
N_Vessels
Reset
Classify


Heart Disease Classifier
Heart Disease Classifier
Age in Years
50
Resting Blood Pressure
140
Cholestrol mg/dl
130
Maximum Heart Rate
100
Depression By Exercise
1.66
Gender
Male
Female
Fasting Blood Pressure
True
False
Exercise Induced Angina
True
False
Chest Pain
2: atypical angina
Resting ECG
2
Slope of Exercise
3
Thallium Stress
3: Reversible Defect
Number Vessels
4
Reset
Classify
Prediction: No Heart Disease Detected


Heart Disease Classifier
Heart Disease Classifier
Age in Years
50
Resting Blood Pressure
140
Cholestrol mg/dl
130
Maximum Heart Rate
130
Depression By Exercise
0.8
Gender
Male
Female
Fasting Blood Pressure
True
False
Exercise Induced Angina
True
False
Chest Pain
3: non-anginal pain
Resting ECG
2
Slope of Exercise
3
Thallium Stress
1: Fixed Defect
Number Vessels
4
Reset
Classify
Prediction: Signs of Heart Disease Deteced
You should consult with your Doctor!

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv("heart.csv")
y = df['target'].values
X = df.drop(['target'], axis = 1).values
from sklearn.model_selection import train_test_split
X_train, X_test, y_train,y_test = train_test_split(X,y,random_state = 42, test_size = 0.2)
from sklearn.linear_model import LogisticRegression
lr_model = LogisticRegression(max_iter=100)
lr_model.fit(X_train,y_train)
lr_model.score(X_test, y_test)
Final_Model = lr_model
import re
from tkinter import *
import tkinter as tk



def check_inputs():
   if age.get() == "":
      print("Age Field is Empty!!")
      Label(win,text="Age Field is Empty!!",fg="blue",bg="yellow",font = ("Calibri 10
bold")).place(x=12,y=580)



   elif rbp.get() == "":
      print("Resting Blood Pressure Field is Empty!!")
```

```
    Label(win,text="Resting Blood Pressure Field is Empty!!",fg="blue",bg="yellow",font = ("Calibri 10 bold")).place(x=12,y=580)


  elif chol.get() == "":
    print("Cholestrol Field is Empty!!")
    Label(win,text="Cholestrol Field is Empty!!",fg="blue",bg="yellow",font = ("Calibri 10 bold")).place(x=12,y=580)


  elif heart_rate.get() == "":
    print("Heart Rate Field is Empty!!")


    Label(win,text="Heart Rate Field is Empty!!",fg="blue",bg="yellow",font = ("Calibri 10 bold")).place(x=12,y=580)
  elif peak.get() == "":
    print("Depression By Exercise Field is Empty!!")
    Label(win,text="Depression By Exercise Field is Empty!!",fg="blue",bg="yellow",font = ("Calibri 10 bold")).place(x=12,y=580)


  else:
    predict()



def predict():
  gender_dict = {"Male":1, "Female":0}
  fbs_dict = {"True":1, "False":0}
  eia_dict = {"True":1, "False":0}
  cp_dict = {"1: typical angina":0,"2: atypical angina":1,"3: non-anginal pain":2,"4: asymptomatic":3}
  thal_dict = {"0: No Test":0,"1: Fixed Defect":1,"2: Normal Flow":2,"3: Reversible Defect":3}
  Pred_dict = {0:"Prediction: No Heart Disease Detected", 1:"Prediction: Heart Disease Deteced\nYou should consult with your Doctor!"}
```

```
    data = [float(age.get()),gender_dict[str(radio.get())], cp_dict[str(variable.get())], float(rbp.get()),
            float(chol.get()),fbs_dict[str(radio_fbs.get())], int(str(variable_ecg.get())) - 1 ,
float(heart_rate.get()),
            eia_dict[str(radio_eia.get())], float(peak.get()), int(str(variable_slope.get())) -
1,int(str(variable_n_vessels.get())) - 1,
            thal_dict[str(variable_thal.get())]]


    prediction = Final_Model.predict(np.array(data).reshape(1,13))
    pred_label = Pred_dict[prediction.tolist()[0]]
    Label(win,text=pred_label,fg="blue",bg="yellow",font = ("Calibri 10 bold")).place(x=12,y=580)




def reset():
    age.set("")
    rbp.set("")
    chol.set("")
    heart_rate.set("")
    peak.set("")




win =  Tk()


win.geometry("450x600")
win.configure(background="#Eaedee")
win.title("Heart Disease Classifier")
```

```
#win.iconbitmap('icon.ico')


title = Label(win, text="Heart Disease Classifier", bg="#2583be", width="300", height="2", fg="white", font = ("Arial 20 italic")).pack()


age = Label(win, text="Age in Years", bg="#Eaedee", font=("Verdana 12")).place(x=12, y=65)


rbp = Label(win, text="Resting Blood Pressure ", bg="#Eaedee", font=("Verdana 12")).place(x=12, y=105)


chol = Label(win, text="Cholestrol mg/dl ", bg="#Eaedee", font=("Verdana 12")).place(x=12, y=145)


heart_rate = Label(win, text="Maximum Heart Rate ", bg="#Eaedee", font=("Verdana 12")).place(x=12, y=185)


peak = Label(win, text="Depression By Exercise ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=225)




Gender = Label(win, text="Gender ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=265)


radio = StringVar()

Male = Radiobutton(win, text="Male",bg="#Eaedee",variable=radio,value="Male",font = ("Verdana 12")).place(x=160,y=265)

Female = Radiobutton(win, text="Female",bg="#Eaedee",variable=radio,value="Female",font = ("Verdana 12")).place(x=260,y=265)


FBS = Label(win, text="Fasting Blood Pressure ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=285)


radio_fbs = StringVar()
```

```
Male = Radiobutton(win, text="True",bg="#Eaedee",variable=radio_fbs,value="True",font = ("Verdana 12")).place(x=160,y=285)

Female = Radiobutton(win, text="False",bg="#Eaedee",variable=radio_fbs,value="False",font = ("Verdana 12")).place(x=260,y=285)


EIA = Label(win, text="Exercise Induced Angina",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=305)


radio_eia = StringVar()

Male = Radiobutton(win, text="True",bg="#Eaedee",variable=radio_eia,value="True",font = ("Verdana 12")).place(x=160,y=305)

Female = Radiobutton(win, text="False",bg="#Eaedee",variable=radio_eia,value="False",font = ("Verdana 12")).place(x=260,y=305)



cp = Label(win,text="Chest Pain ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=345)

variable = StringVar(win)

variable.set("CP")

w = OptionMenu(win, variable, "1: typical angina","2: atypical angina","3: non-anginal pain","4: asymptomatic")

w.place(x=140,y=345)


ecg = Label(win,text="Resting ECG ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=385)

variable_ecg = StringVar(win)

variable_ecg.set("ECG")

w_ecg = OptionMenu(win, variable_ecg, "1","2","3")

w_ecg.place(x=140,y=385)


exer_slope = Label(win,text="Slope of Exercise ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=425)

variable_slope = StringVar(win)
```

```
variable_slope.set("Slope")

w_slope = OptionMenu(win, variable_slope, "1","2","3")

w_slope.place(x=140,y=425)



thal_label = Label(win,text="Thallium Stress ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=465)

variable_thal = StringVar(win)

variable_thal.set("Thal")

w_thal = OptionMenu(win, variable_thal, "0: No Test","1: Fixed Defect","2: Normal Flow","3: Reversible
Defect")

w_thal.place(x=140,y=465)



n_vessels = Label(win,text="Number Vessels ",bg="#Eaedee",font = ("Verdana 12")).place(x=12,y=505)

variable_n_vessels = StringVar(win)

variable_n_vessels.set("N_Vessels")

w_n_vessels = OptionMenu(win, variable_n_vessels, "1","2","3","4")

w_n_vessels.place(x=140,y=505)



age = StringVar()

rbp = StringVar()

chol = StringVar()

heart_rate = StringVar()

peak  = StringVar()

Gender = StringVar()

FBS  = StringVar()

EIA  = StringVar()

cp  = StringVar()
```

```
ecg  = StringVar()

exer_slope  = StringVar()

thal_label  = StringVar()

n_vessels  = StringVar()


entry_age = Entry(win,textvariable = age,width=30)

entry_age.place(x=150,y=65)


entry_rbp = Entry(win,textvariable = rbp,width=30)

entry_rbp.place(x=150,y=105)


entry_chol = Entry(win,textvariable = chol,width=30)

entry_chol.place(x=150,y=145)


entry_heart_rate = Entry(win, textvariable = heart_rate,width=30)

entry_heart_rate.place(x=150,y=185)


entry_peak = Entry(win,textvariable = peak,width=30)

entry_peak.place(x=150,y=225)


reset = Button(win, text="Reset", width="12",height="1",activebackground="red",command=reset,
bg="Pink",font = ("Calibri 12 ")).place(x=24, y=540)

submit = Button(win, text="Classify", width="12",height="1",activebackground="violet",
bg="Pink",command=check_inputs,font = ("Calibri 12 ")).place(x=240, y=540)



win.mainloop()
```