In the name of ALLAH

Numerical Analysis Project Documentation

Seyed Mohsen Razavi Zadegan Jahromi

Student Number : 40030489

Data set : Body Measurements

Goal : Predicting age based body measurements.

Summer 2023

About Dataset

This dataset contains information of 715 persons (391 Males & 324 Females).

Some thing such as :

Gender (Male=1, Female= 2)

Age (1 year and above)

Head Circumference (in inches)

Shoulder Width (in inches)

Chest Width (in inches)

Belly (in inches)

Waist (in inches)

Hips (in inches)

Shoulder To Waist (in inches)

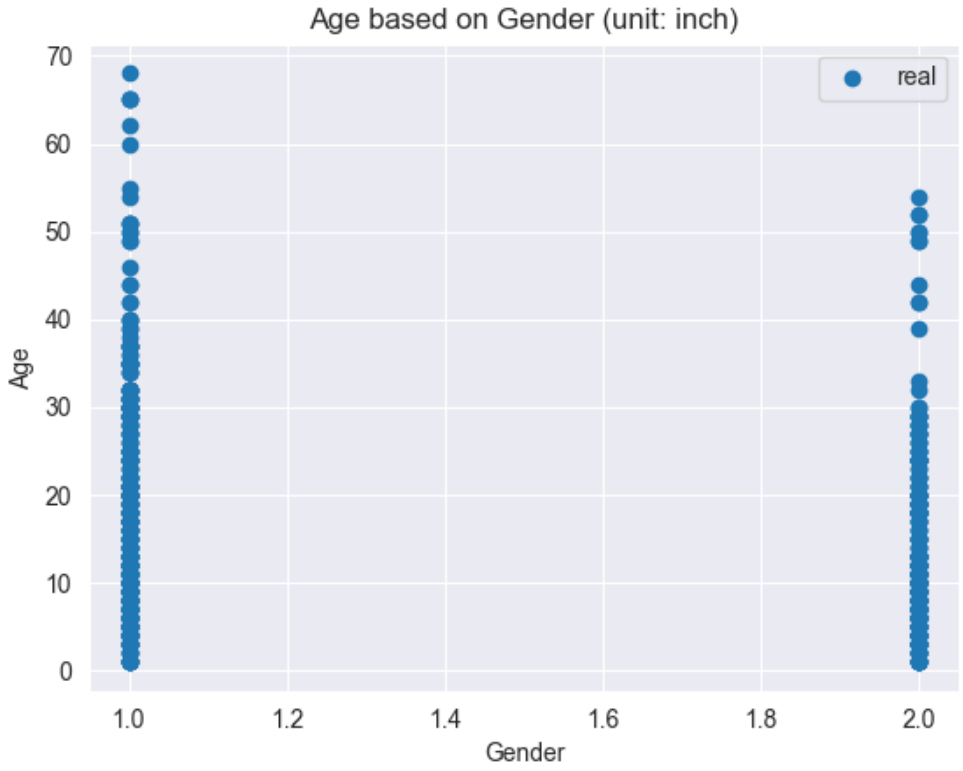
Waist To Knee (in inches)

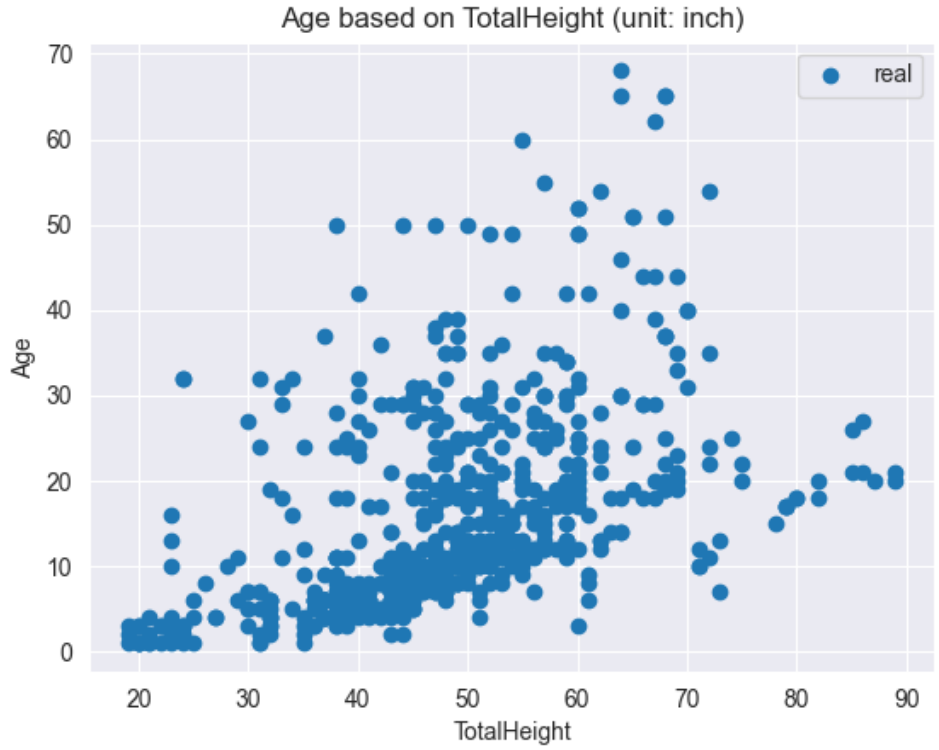
Leg Length (in inches)

Total Height - from head to toe (in inches)

Arm Length (in inches)

These charts help you have a better perception of the dataset :


Age based on Gender (unit: inch)
real
Age
Gender
0
10
20
30
40
50
60
70
1.0
1.2
1.4
1.6
1.8
2.0


Age based on TotalHeight (unit: inch)
real
Age
TotalHeight
0
10
20
30
40
50
60
70
20
30
40
50
60
70
80
90

Linear Regression

Linear regression is a type of statistical analysis used to predict the relationship between two variables. It assumes a linear relationship between the independent variable and the dependent variable, and aims to find the best-fitting line that describes the relationship. The line is determined by minimizing the sum of the squared differences between the predicted values and the actual values.

Linear regression is commonly used in many fields, including economics, finance, and social sciences, to analyze and predict trends in data. It can also be extended to multiple linear regression, where there are multiple independent variables, and logistic regression, which is used for binary classification problems.

Simple Linear Regression

In a simple linear regression, there is one independent variable and one dependent variable. The model estimates the slope and intercept of the line of best fit, which represents the relationship between the variables. The slope represents the change in the dependent variable for each unit change in the independent variable, while the intercept represents the predicted value of the dependent variable when the independent variable is zero.

The result of simple linear regression is a linear equation:

$$y = w * x + i$$

w: slope, i: intercept

Multiple Linear Regression

Multiple linear regression is a technique to understand the relationship between a single dependent variable and multiple independent variables. The formulation for multiple linear regression is also similar to simple linear regression with the small change that instead of having one beta variable, you will now have betas for all the variables used.

$$\mathrm{y} = \mathrm{w1} * \mathrm{x} + \mathrm{w2} * \mathrm{t} + \mathrm{w3}*\mathrm{e} + \ldots + \mathrm{i}$$

Or :

$$y = i + \sum_{n=0}^{m} wn * xn$$

In this project we have used multiple linear regression according to our goal with python programming language and its libraries.

About Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation via the off-side rule.

The last version of python is 3.11.4 right now. But I used python 3.10.5 because I had this version installed on my system. Actually there is no difference between these versions. All of them are python 3 !

Required Libraries

Sklearn : I used this library to create my model and predict age.

Numpy : numpy usually is used for numerical analysis and controlling data, arrays and matrices.

Pandas : I used pandas to read data from dataset. As file format is .csv reading data from it is not as easy as reading from .txt file.

Matplotlib : matplotlib is python's library to plot diagrams.

Seaborn : I used seaborn to add some styles to plots created by matplotlib.

So lets start …

Starting project

First create a virtual environment with the command below:

```
python -m venv name
```

I named it venv.

Then install libraries using pip:

```
pip install sklearn
pip install numpy
pip install pandas
pip install matplotlib
pip install seaborn
```

import libraries in the main file:

```
import pandas
from sklearn.linear_model import LinearRegression
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

I read dataset using pandas:

```
body_measurements_df = pandas.read_csv('Body Measurements _
original_CSV.csv')
```

this gives us our dataset as a dataframe.

Note : because the dataset is near of main file in the root of project it is enough to give the function name of csv file.

The goal is to predict age using other data, so separate them :

```
input_fields = ['Gender', 'HeadCircumference',
'ShoulderWidth', 'ChestWidth', 'Belly', 'Waist', 'ArmLength',
'ShoulderToWaist', 'WaistToKnee', 'LegLength', 'TotalHeight']

inputs = np.array(body_measurements_df[input_fields])
target = np.array(body_measurements_df['Age'])
```

Note : to make code prettier, first I created list of input fields then I used it to slice dataframe.

As sklearn works better with numpy arrays, I convert dataframe to numpy array directly.

Then we need to create an instance from LinearRegression class which we imported from sklearn.linear_model and fit this on the inputs and target :

```
model = LinearRegression()

model.fit(np.nan_to_num(inputs), np.nan_to_num(target))
```

Note : to fix error of having Nan in inputs or target, I used nan_to_num method of numpy.

Now we can predict using our fitted model and error :

```
predicted_age = int(model.predict(user_data))
rmse = np.sqrt(np.mean(np.square(target - predicted_age)))
```

Note : as we know that age is disjoint variable, and can't be a floating point number I converted it to an integer. RMSE or Root Mean Square Error is calculated through :

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(p_i - y_i)^2}{n}}$$

p: predicted value, y : real value

then we can print the result and its error:

```
print(predicted_age, '+-', rmse)
```

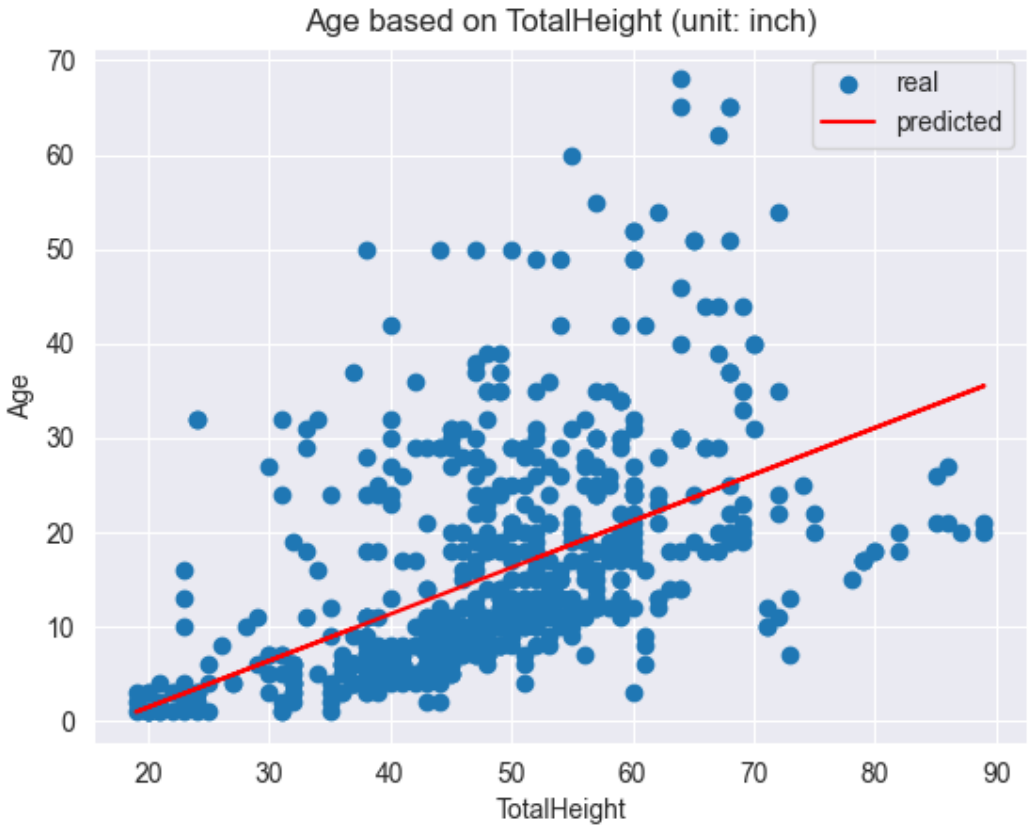
Our program is finished but it is one more thing we should do. How user can give input to program ? I think it is good to specify a .txt file input for user to enter his input values there. I name this file "user_input_data.txt" and create it near to main file. This way we can read input values:

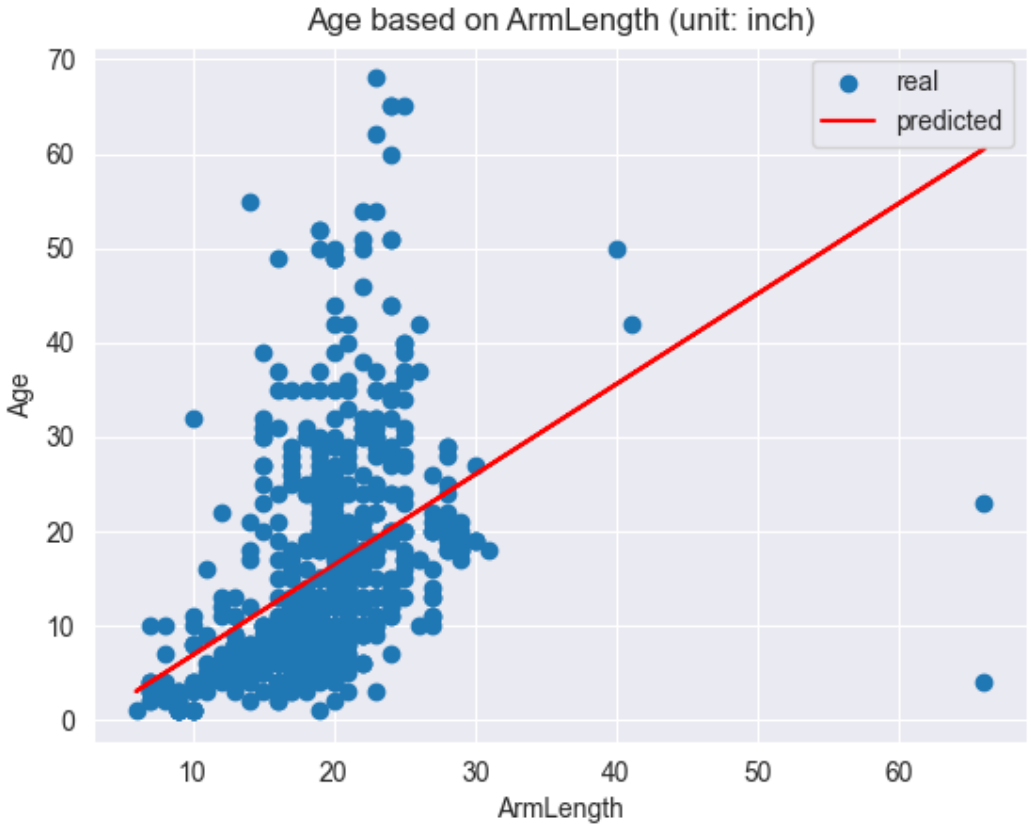
```
with open('user_input_data.txt', 'r') as file:
    try:
        user_data = [[float(s.strip().split(':')[1]) for s in
file.readlines()[2:]]]
    except:
        print('Invalid input file')
        exit()
```

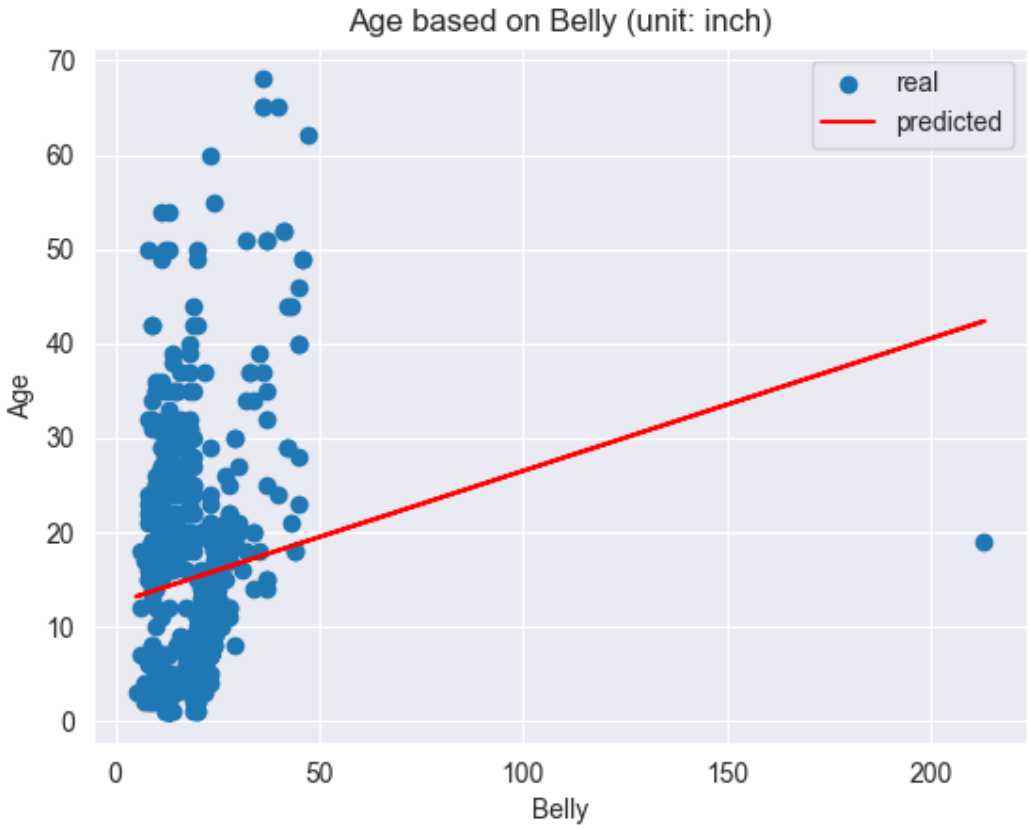
Note : if user left a field blank our program will have errors, so it is good to don't let program to be continued if user_data creation didn't work correctly.

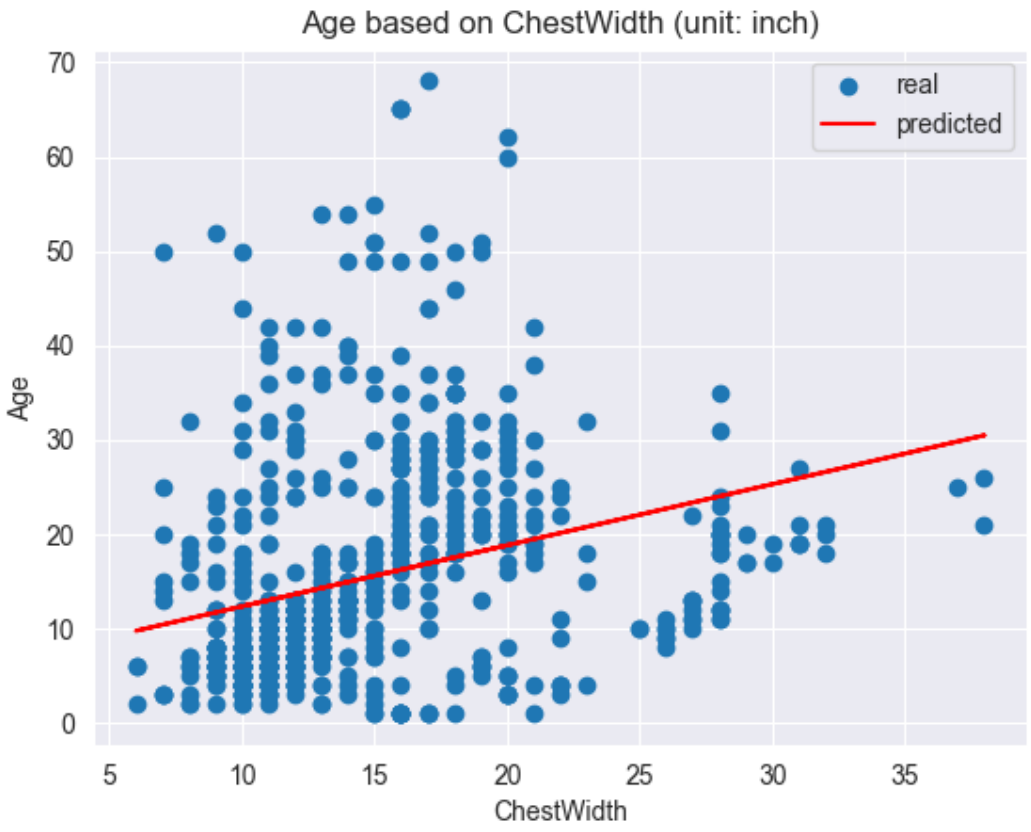
Finished 😊

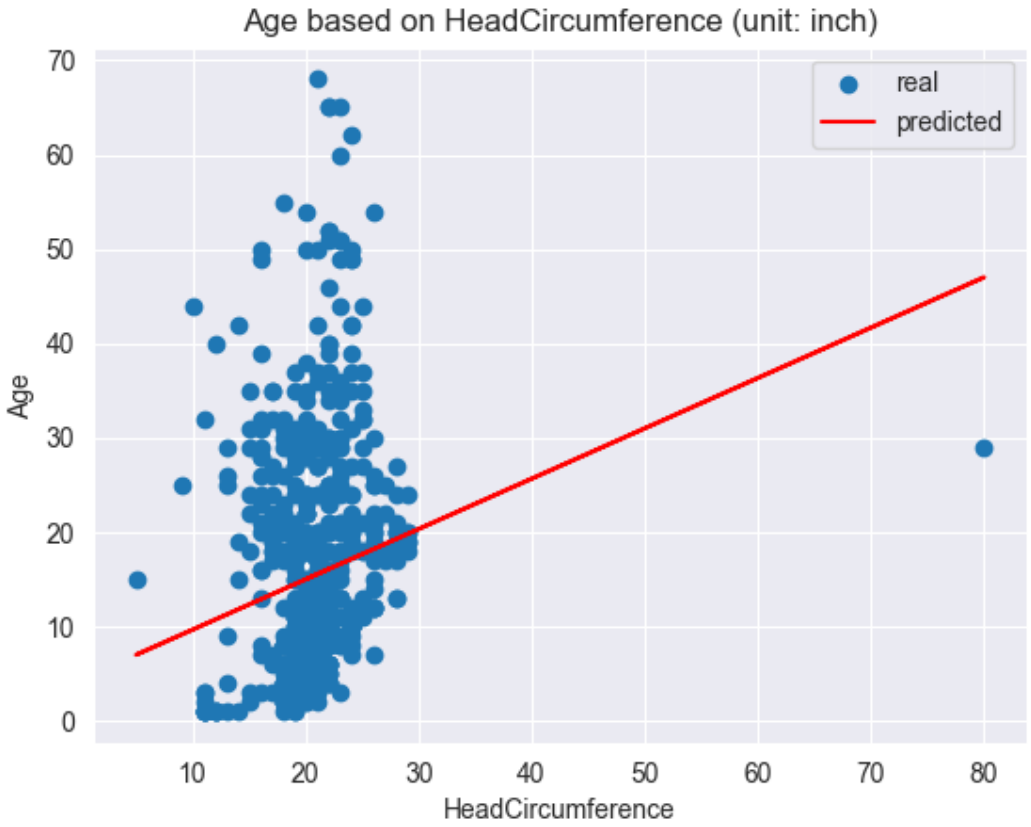
But as humans retrieve 80 percent of information using his sight, it is good to create charts and diagrams of our predictions and real data together :)

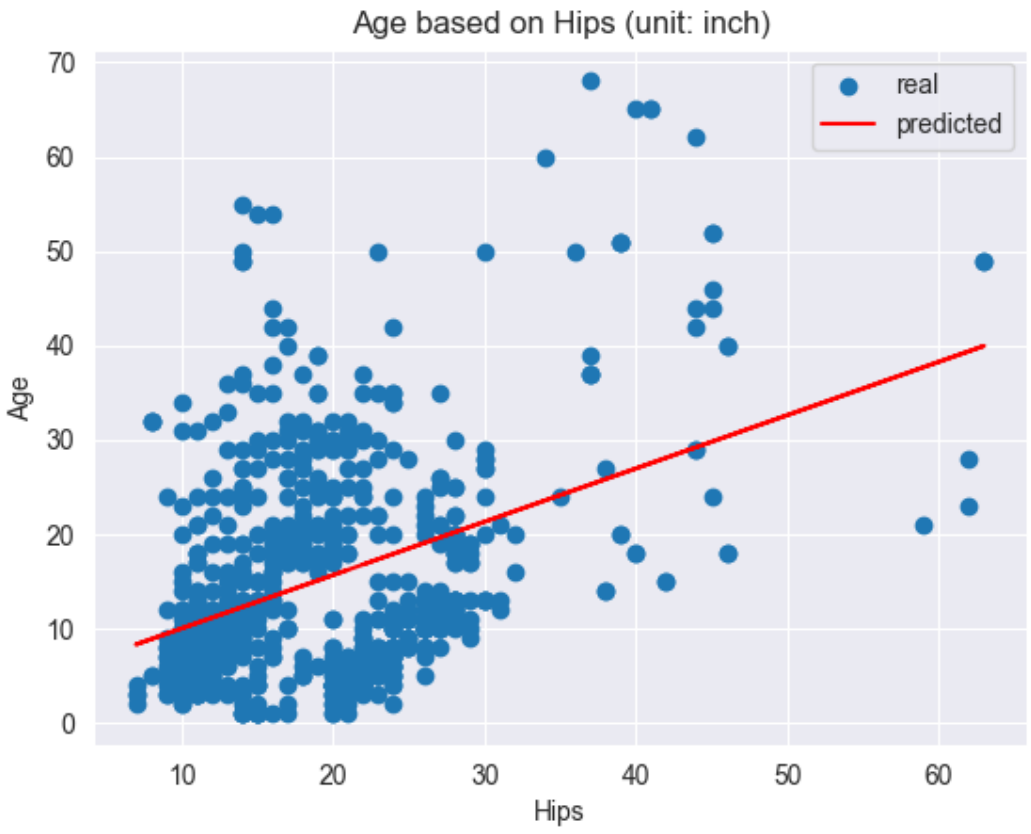

Age based on TotalHeight (unit: inch)
real
predicted
Age
TotalHeight
0
10
20
30
40
50
60
70
20
30
40
50
60
70
80
90

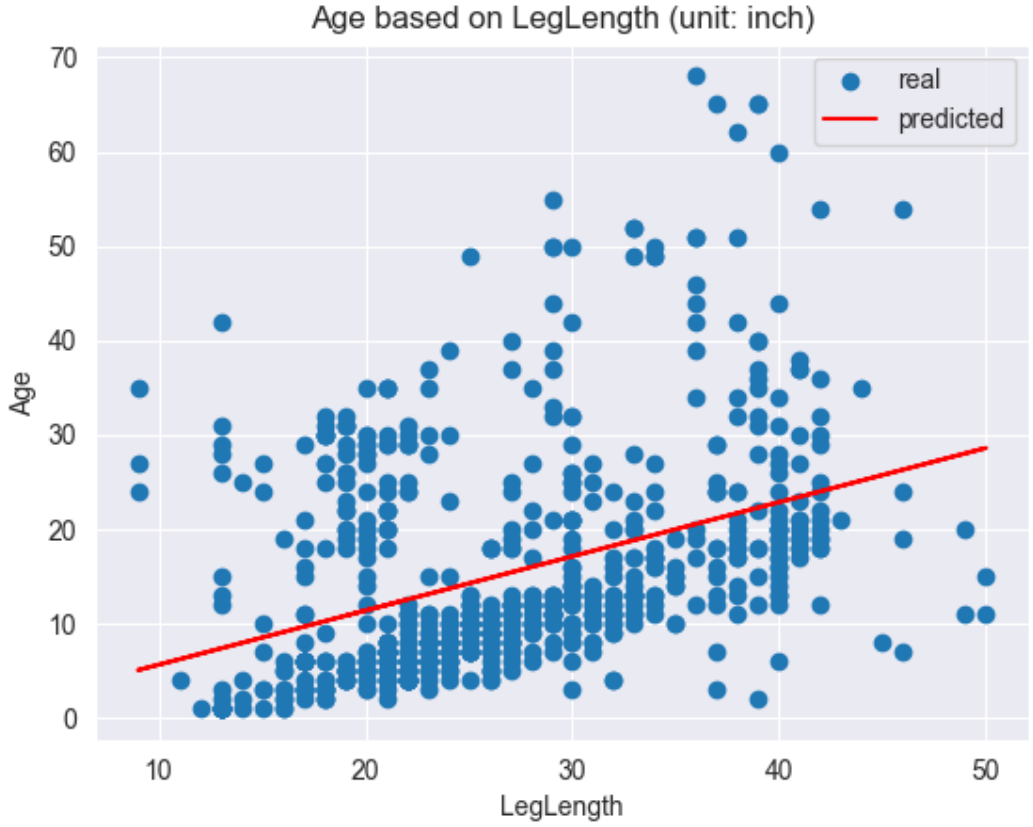

Age based on ArmLength (unit: inch)
real
predicted
Age
ArmLength
0
10
20
30
40
50
60
70
10
20
30
40
50
60

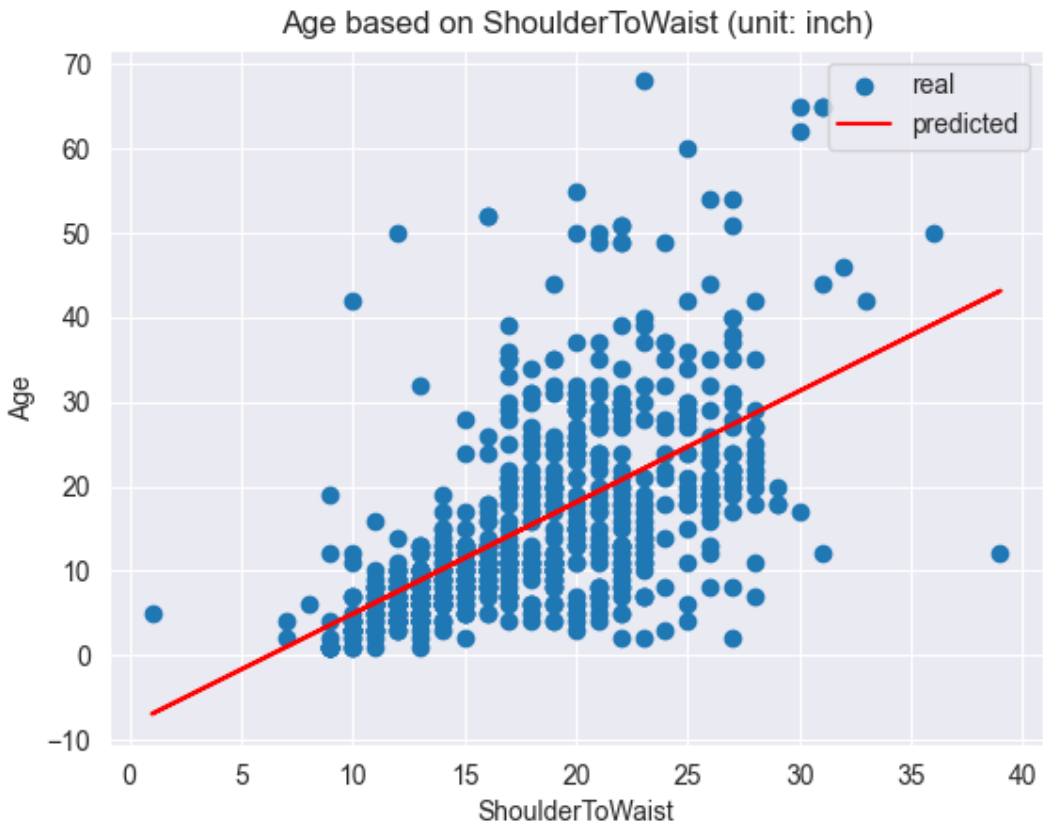

Age based on Belly (unit: inch)
real
predicted
Age
Belly
0
10
20
30
40
50
60
70
0
50
100
150
200

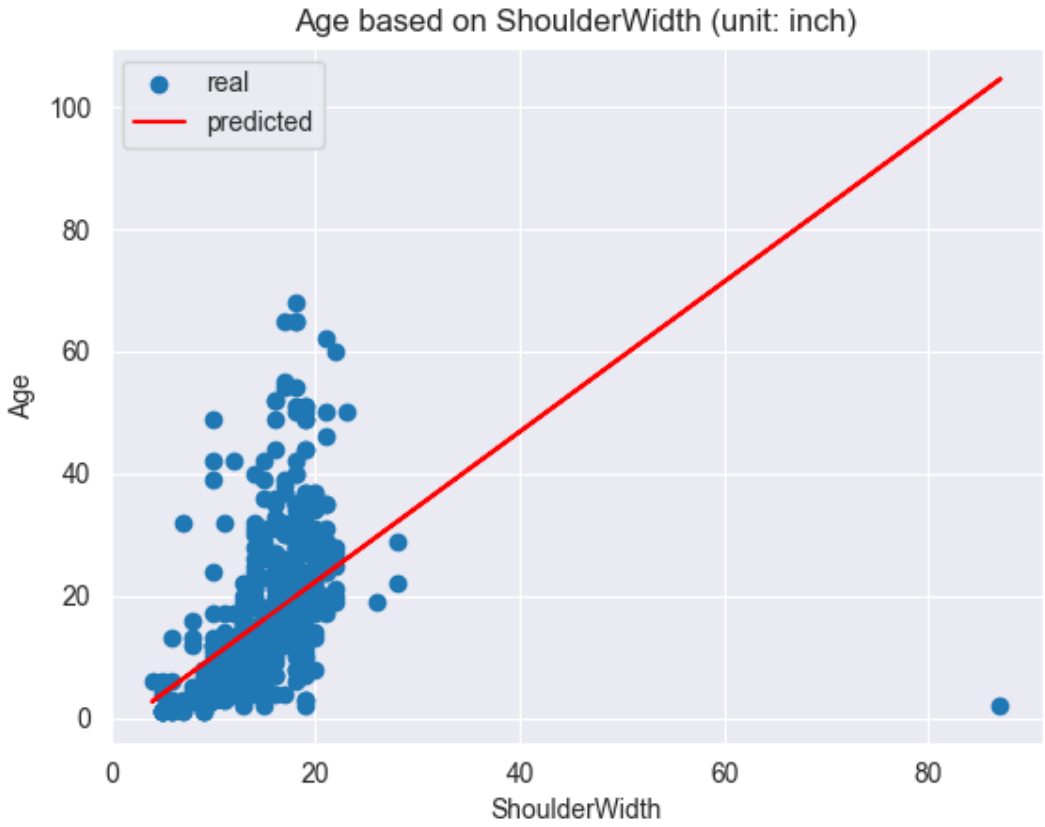

Age based on ChestWidth (unit: inch)
real
predicted
Age
ChestWidth
0
10
20
30
40
50
60
70
5
10
15
20
25
30
35

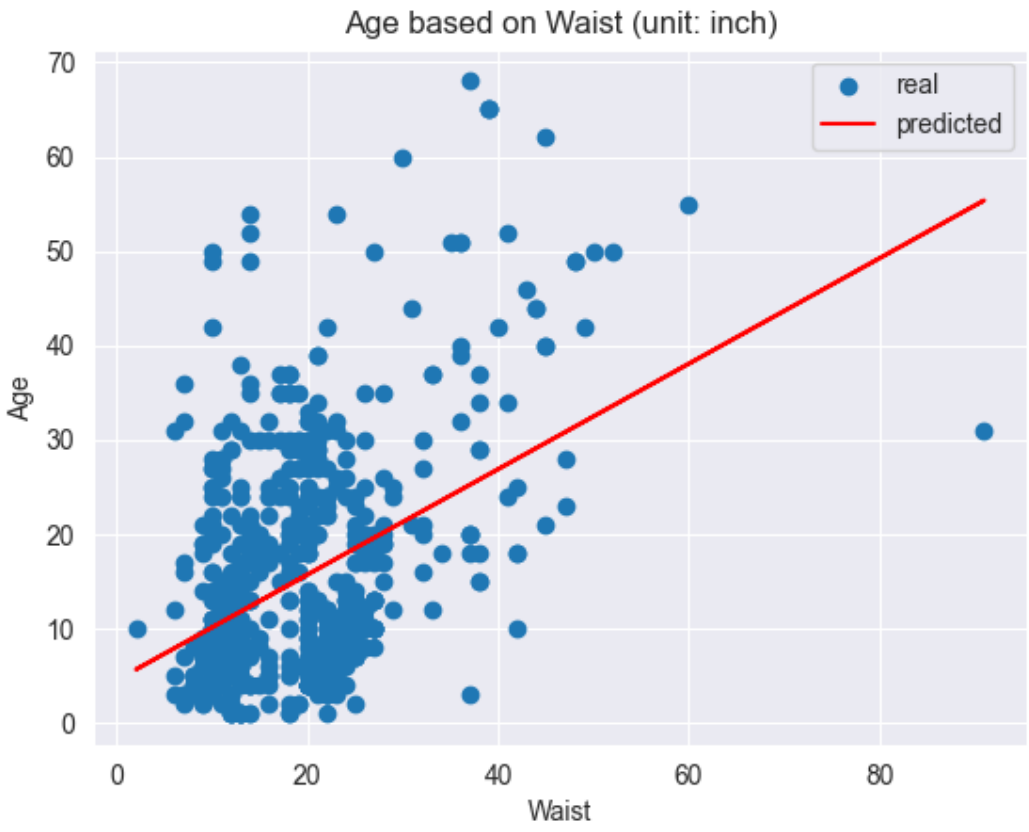

Age based on HeadCircumference (unit: inch)
real
predicted
Age
HeadCircumference
0
10
20
30
40
50
60
70
10
20
30
40
50
60
70
80

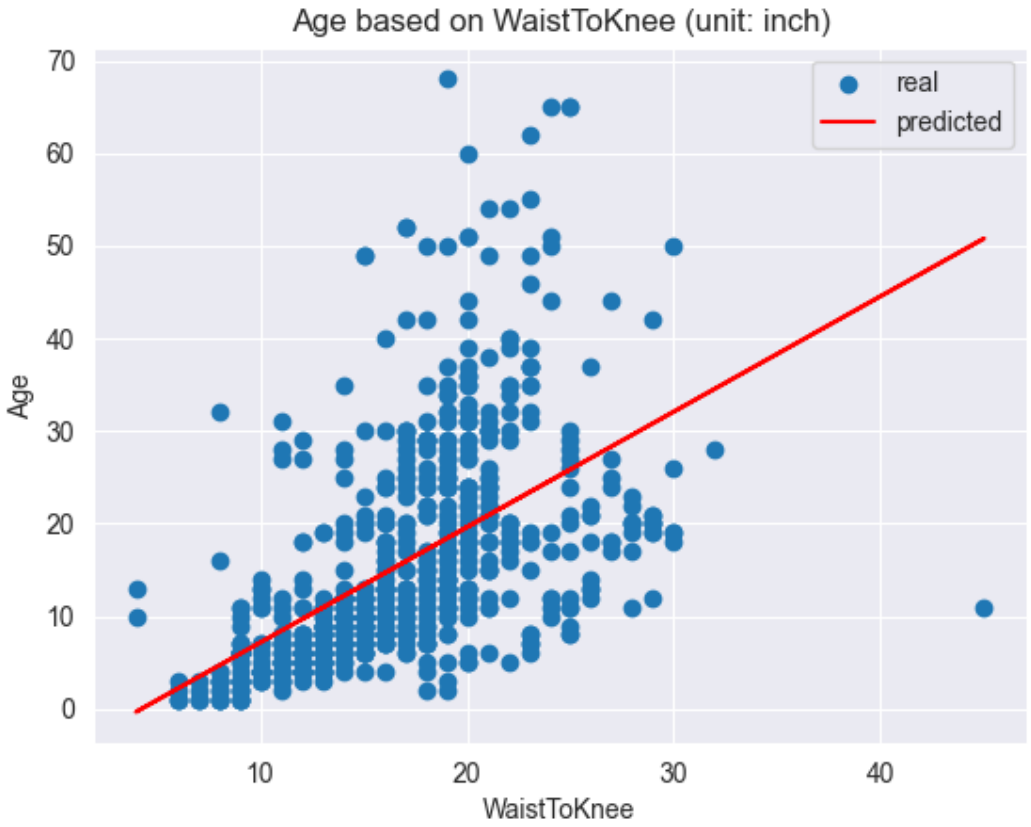

Age based on Hips (unit: inch)
real
predicted
Age
Hips
0
10
20
30
40
50
60
70
10
20
30
40
50
60


Age based on LegLength (unit: inch)
real
predicted
Age
LegLength
70
60
50
40
30
20
10
0
10
20
30
40
50


Age based on ShoulderToWaist (unit: inch)
real
predicted
Age
ShoulderToWaist
70
60
50
40
30
20
10
0
−10
0
5
10
15
20
25
30
35
40


Age based on ShoulderWidth (unit: inch)
real
predicted
Age
ShoulderWidth
0
20
40
60
80
100


Age based on Waist (unit: inch)
real
predicted
Age
Waist
0
10
20
30
40
50
60
70
80



Github link (private until deadline): https://github.com/MohsenRazavi/numeric-analysis-project