Data Science Lab 3

**201635833 Lee Sungmin**

Histogram

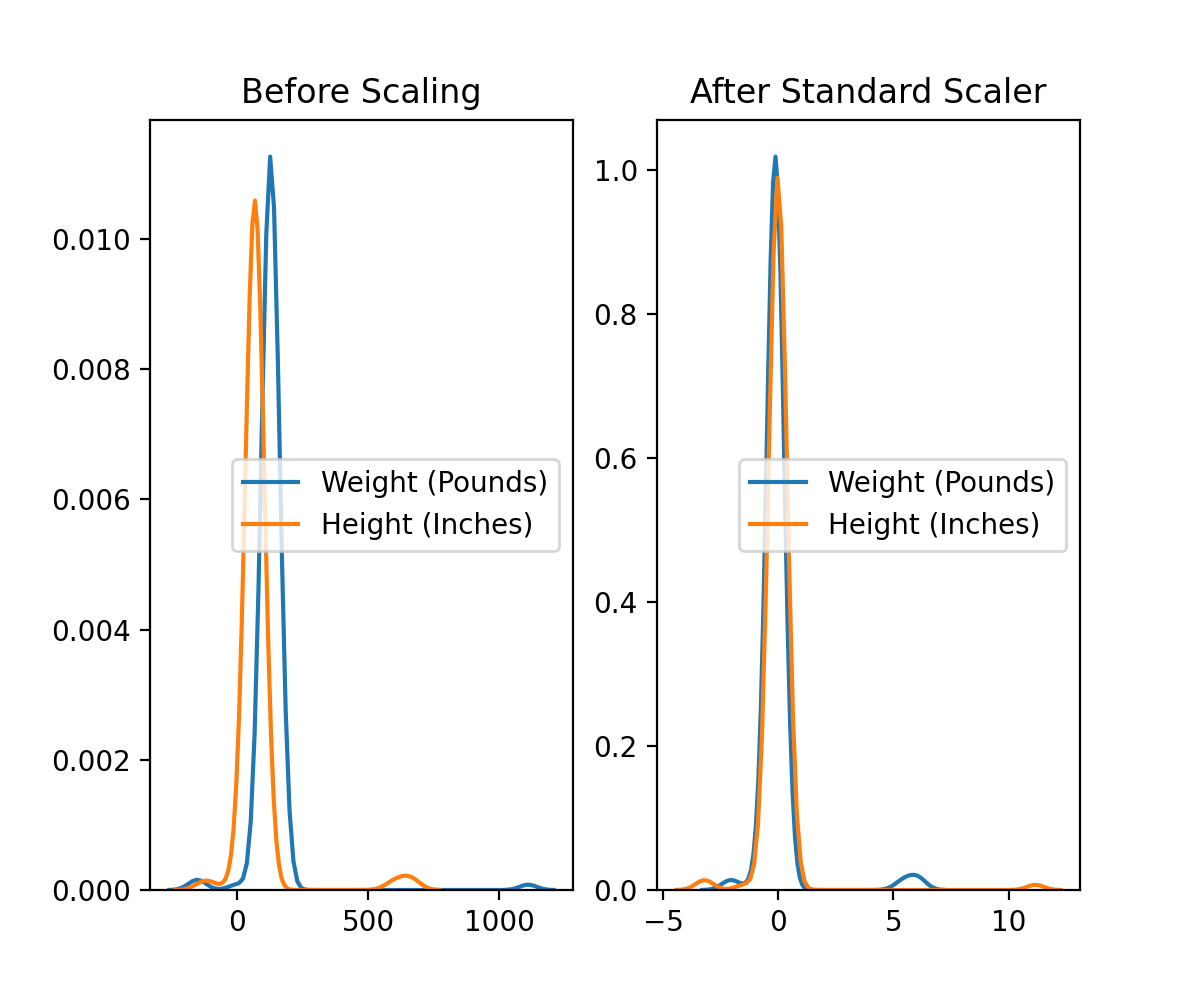
스크린샷이(가) 표시된 사진

자동 생성된 설명

import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
import sklearn  
from sklearn import preprocessing  
  
  
#read csv file which directory is in project file  
#Also set Missing data to np.nan  
df = pd.read\_csv("bmi\_data\_lab3.csv",na\_values=np.nan)  
  
tempset = df[df['BMI']==1]  
height = tempset['Height (Inches)']  
plt.hist(height,bins=10)  
plt.subplot(331)  
plt.hist(height)  
plt.title("BMI : 1")  
plt.xlabel('Height (Inches)')  
  
tempset = df[df['BMI']==2]  
height = tempset['Height (Inches)']  
plt.hist(height,bins=10)  
plt.subplot(332)  
plt.hist(height)  
plt.title("BMI : 2")  
plt.xlabel('Height (Inches)')  
  
tempset = df[df['BMI']==3]  
height = tempset['Height (Inches)']  
plt.subplot(333)  
plt.hist(height)  
plt.title("BMI : 3")  
plt.xlabel('Height (Inches)')  
  
tempset = df[df['BMI']==1]  
height = tempset['Weight (Pounds)']  
plt.hist(height,bins=10)  
plt.subplot(334)  
plt.hist(height)  
plt.title("BMI : 1")  
plt.xlabel('Weight (Pounds)')  
  
tempset = df[df['BMI']==2]  
height = tempset['Weight (Pounds)']  
plt.hist(height,bins=10)  
plt.subplot(335)  
plt.hist(height)  
plt.title("BMI : 2")  
plt.xlabel('Weight (Pounds)')  
  
tempset = df[df['BMI']==3]  
height = tempset['Weight (Pounds)']  
plt.hist(height,bins=10)  
plt.subplot(336)  
plt.hist(height)  
plt.title("BMI : 3")  
plt.xlabel('Weight (Pounds)')  
  
plt.show()

**Standardization**

import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
import sklearn  
import seaborn as sns  
from sklearn import preprocessing  
  
  
#read csv file which directory is in project file  
#Also set Missing data to np.nan  
df = pd.read\_csv("bmi\_data\_lab3.csv",na\_values=np.nan)  
  
score = np.array(df[['Height (Inches)','Weight (Pounds)']])  
  
scaler = preprocessing.StandardScaler()  
scaled\_df = scaler.fit\_transform(score)  
scaled\_df = pd.DataFrame(  
scaled\_df, columns=['Weight (Pounds)', 'Height (Inches)'])  
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(6, 5))  
ax1.set\_title('Before Scaling')  
sns.kdeplot(df['Weight (Pounds)'], ax=ax1)  
sns.kdeplot(df['Height (Inches)'], ax=ax1)  
ax2.set\_title('After Standard Scaler')  
sns.kdeplot(scaled\_df['Weight (Pounds)'], ax=ax2)  
sns.kdeplot(scaled\_df['Height (Inches)'], ax=ax2)  
plt.show()



**minMaxScaler**

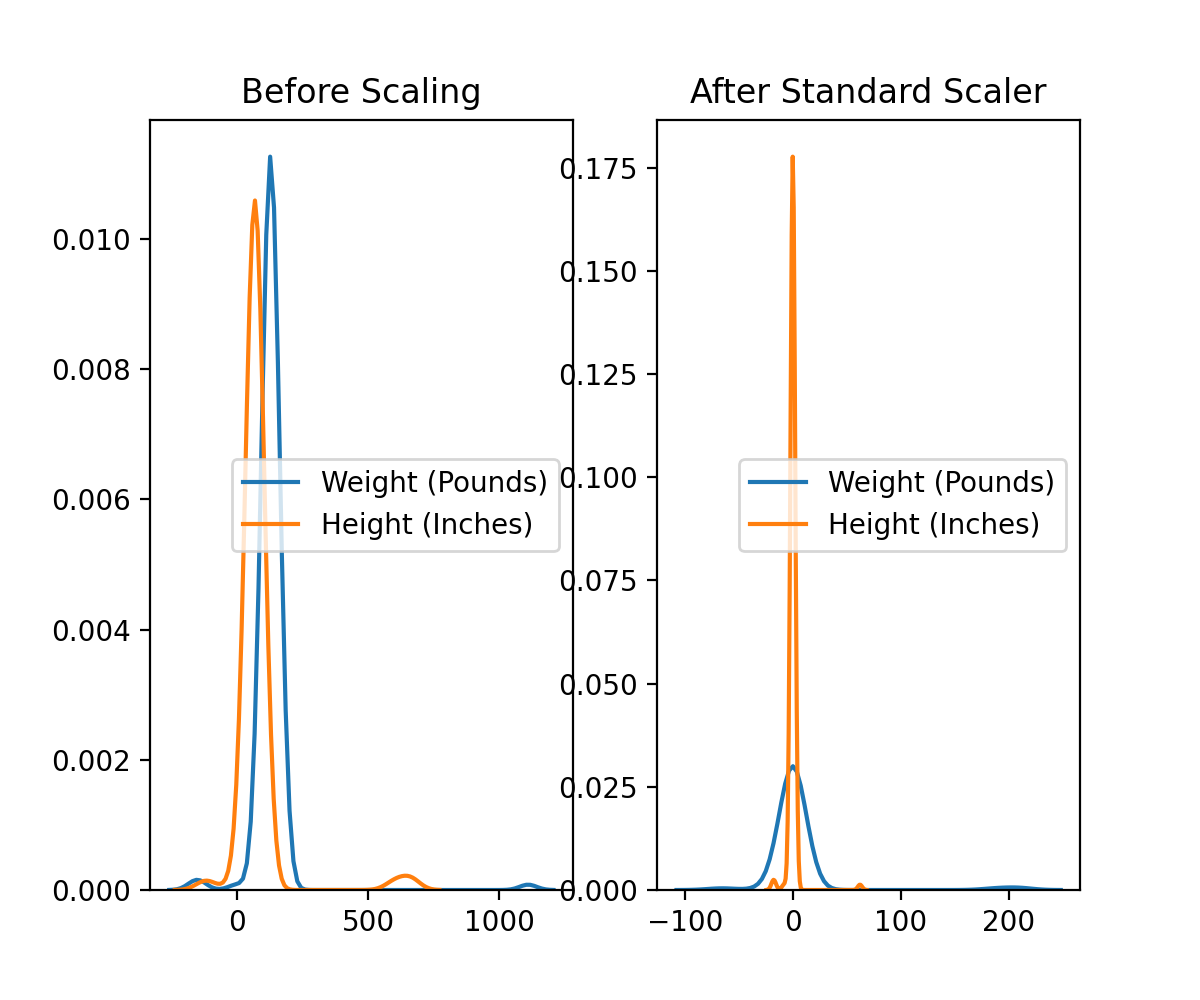
import numpy as np  
import matplotlib.pyplot as plt  
import pandas as pd  
import sklearn  
import seaborn as sns  
from sklearn import preprocessing  
  
  
#read csv file which directory is in project file  
#Also set Missing data to np.nan  
df = pd.read\_csv("bmi\_data\_lab3.csv",na\_values=np.nan)  
  
score = np.array(df[['Height (Inches)','Weight (Pounds)']])  
  
scaler = preprocessing.MinMaxScaler()  
scaled\_df = scaler.fit\_transform(score)  
scaled\_df = pd.DataFrame(  
scaled\_df, columns=['Weight (Pounds)', 'Height (Inches)'])  
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(6, 5))  
ax1.set\_title('Before Scaling')  
sns.kdeplot(df['Weight (Pounds)'], ax=ax1)  
sns.kdeplot(df['Height (Inches)'], ax=ax1)  
ax2.set\_title('After Standard Scaler')  
sns.kdeplot(scaled\_df['Weight (Pounds)'], ax=ax2)  
sns.kdeplot(scaled\_df['Height (Inches)'], ax=ax2)  
plt.show()

**스크린샷이(가) 표시된 사진

자동 생성된 설명**

**Robuster Scaler**

score = np.array(df[['Height (Inches)','Weight (Pounds)']])  
scaler = preprocessing.RobustScaler()  
scaled\_df = scaler.fit\_transform(score)  
scaled\_df = pd.DataFrame(  
scaled\_df, columns=['Weight (Pounds)', 'Height (Inches)'])  
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(6, 5))  
ax1.set\_title('Before Scaling')  
sns.kdeplot(df['Weight (Pounds)'], ax=ax1)  
sns.kdeplot(df['Height (Inches)'], ax=ax1)  
ax2.set\_title('After Standard Scaler')  
sns.kdeplot(scaled\_df['Weight (Pounds)'], ax=ax2)  
sns.kdeplot(scaled\_df['Height (Inches)'], ax=ax2)  
plt.show()



**Wrong values make them NaN**

# change value to NaN which is out of range

def age\_limit(age):  
 if age>0 and age<100:  
 return age  
 return np.nan  
  
def height\_limit(height):  
 if height > 0 and height < 80:  
 return height  
 return np.nan  
  
def weight\_limit(weight):  
 if weight>0 and weight < 440:  
 return weight  
 return np.nan  
  
def BMI\_limit(bmi):  
 if bmi ==0 or bmi == 1 or bmi == 2 or bmi ==3 or bmi ==4:  
 return bmi  
 return np.nan  
  
#read csv file which directory is in project file  
#Also set Missing data to np.nan  
df = pd.read\_csv("bmi\_data\_lab3.csv",na\_values=np.nan)  
  
df['Age'] = df['Age'].apply(age\_limit)  
df['Height (Inches)']=df['Height (Inches)'].apply(height\_limit)  
df['Weight (Pounds)'] = df['Weight (Pounds)'].apply(weight\_limit)  
df['BMI'] = df['BMI'].apply(BMI\_limit)  
  
array = df.values  
print(array)  
print(df.isna().sum())

**Print every NAN value # in row and Column**

**그리기이(가) 표시된 사진

자동 생성된 설명**

print(df[df.isnull().any(axis=1)].iloc[:, []].index)  
print(df.columns[df.isna().any()].tolist())

#this is row and column for including np.nan

print(df.dropna().values)

#Print except NaN value

**Mean, Median , ffill , bfill**

temp = df  
#Save original DataFrame  
  
#Mean application part  
df = df.dropna()  
#Drop row with NaN value  
height\_mean = df['Height (Inches)'].mean()  
weight\_mean = df['Weight (Pounds)'].mean()  
BMI\_mean = df['BMI'].mean()  
#get mean of each column which has nan value  
df = temp

#apply mean to NaN  
df['Height (Inches)']= df['Height (Inches)'].fillna(height\_mean)  
df['Weight (Pounds)']=df['Weight (Pounds)'].fillna(weight\_mean)  
df['BMI'] = df['BMI'].fillna(BMI\_mean)  
print(df.values)  
  
#Median application part  
df = temp  
df = df.dropna()  
height\_median = df['Height (Inches)'].median()  
weight\_median = df['Weight (Pounds)'].median()  
BMI\_median = df['BMI'].median()  
#get mean of each column which has nan value  
df = temp

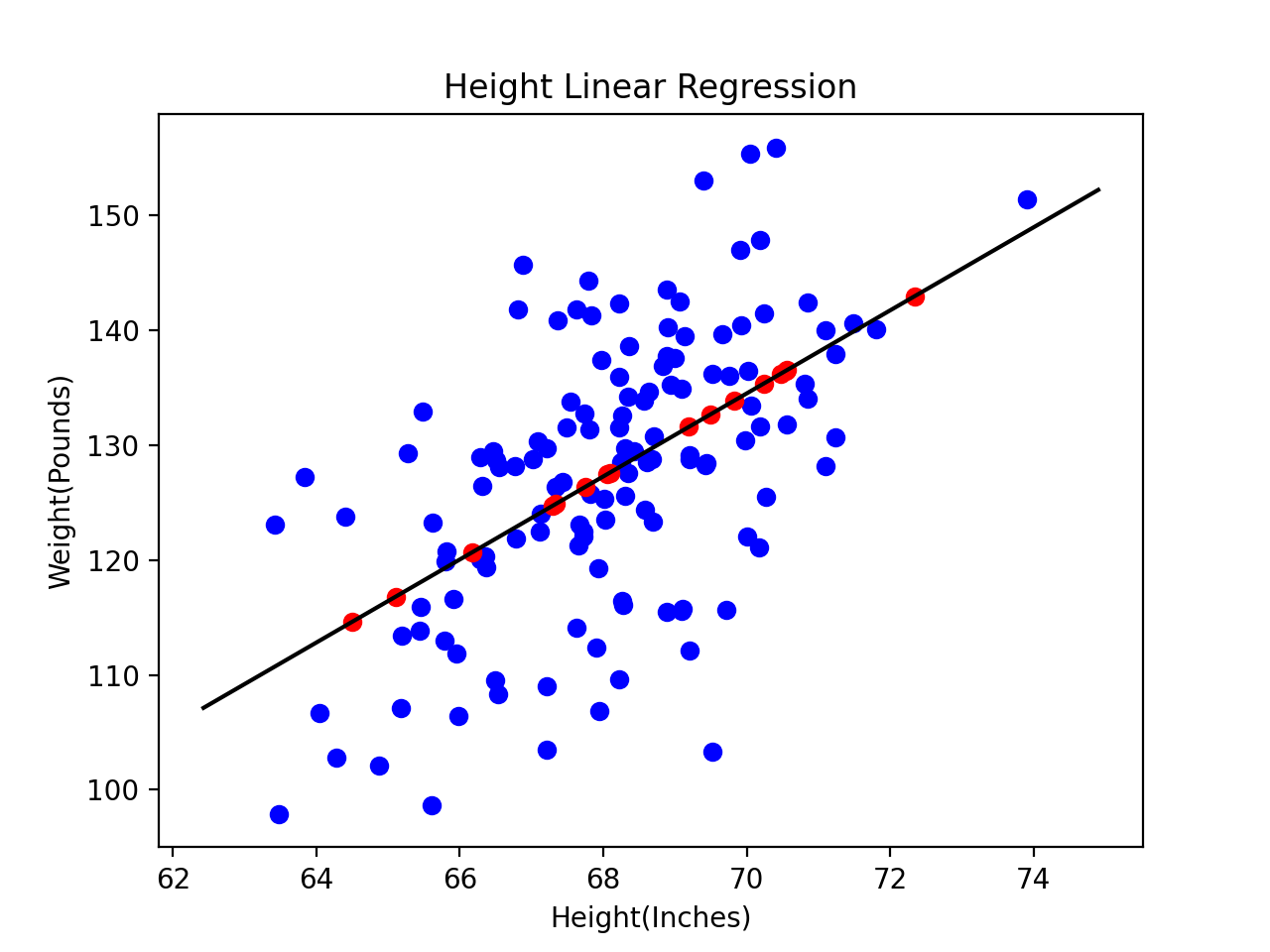
#apply median to NaN  
df['Height (Inches)']= df['Height (Inches)'].fillna(height\_median)  
df['Weight (Pounds)']=df['Weight (Pounds)'].fillna(weight\_median)  
df['BMI'] = df['BMI'].fillna(BMI\_median)  
print(df.values)

#ffil part  
df = temp  
df = df.fillna(axis=0 , method="ffill")

print(df.values)

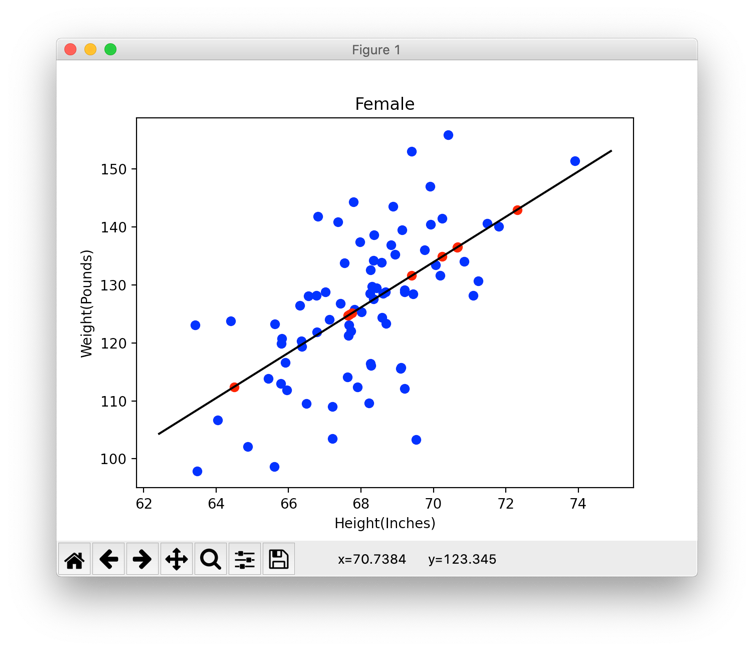
#bfil part  
df = temp  
df = df.bfill(axis=0 , method = "bfill")  
print(df.values)

**Regression ( Weight & Height)**

****

#read csv file which directory is in project file  
#Also set Missing data to np.nan  
df = pd.read\_csv("bmi\_data\_lab3.csv",na\_values=np.nan)  
  
df['Age'] = df['Age'].apply(age\_limit)  
df['Height (Inches)']=df['Height (Inches)'].apply(height\_limit)  
df['Weight (Pounds)'] = df['Weight (Pounds)'].apply(weight\_limit)  
df['BMI'] = df['BMI'].apply(BMI\_limit)  
temp = df  
  
df = df.dropna()  
  
Height = df["Height (Inches)"]  
Weight = df["Weight (Pounds)"]  
df = temp  
Height2 = df["Height (Inches)"].loc[df["Weight (Pounds)"].isna()]  
Weight2 = df["Weight (Pounds)"].loc[df["Height (Inches)"].isna()]  
wh = df.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh = wh[wh.isnull().any(axis=1)].to\_numpy()  
  
reg = linear\_model.LinearRegression()  
  
reg.fit(Height.values.reshape(-1,1), Weight)  
  
px = np.array([Height.min()-1,Height.max()+1])  
py = reg.predict(px[:,np.newaxis])  
  
for data in wh:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py[0])/reg.coef\_)+px[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg.predict([[data[0]]])  
plt.scatter(Height, Weight,color = "blue")  
plt.scatter(wh[:,0],wh[:,1],color = "red")  
plt.plot(px,py,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("Height Linear Regression")  
plt.show()

**Regression(Female & Male)**

**모니터, 컴퓨터이(가) 표시된 사진

자동 생성된 설명**

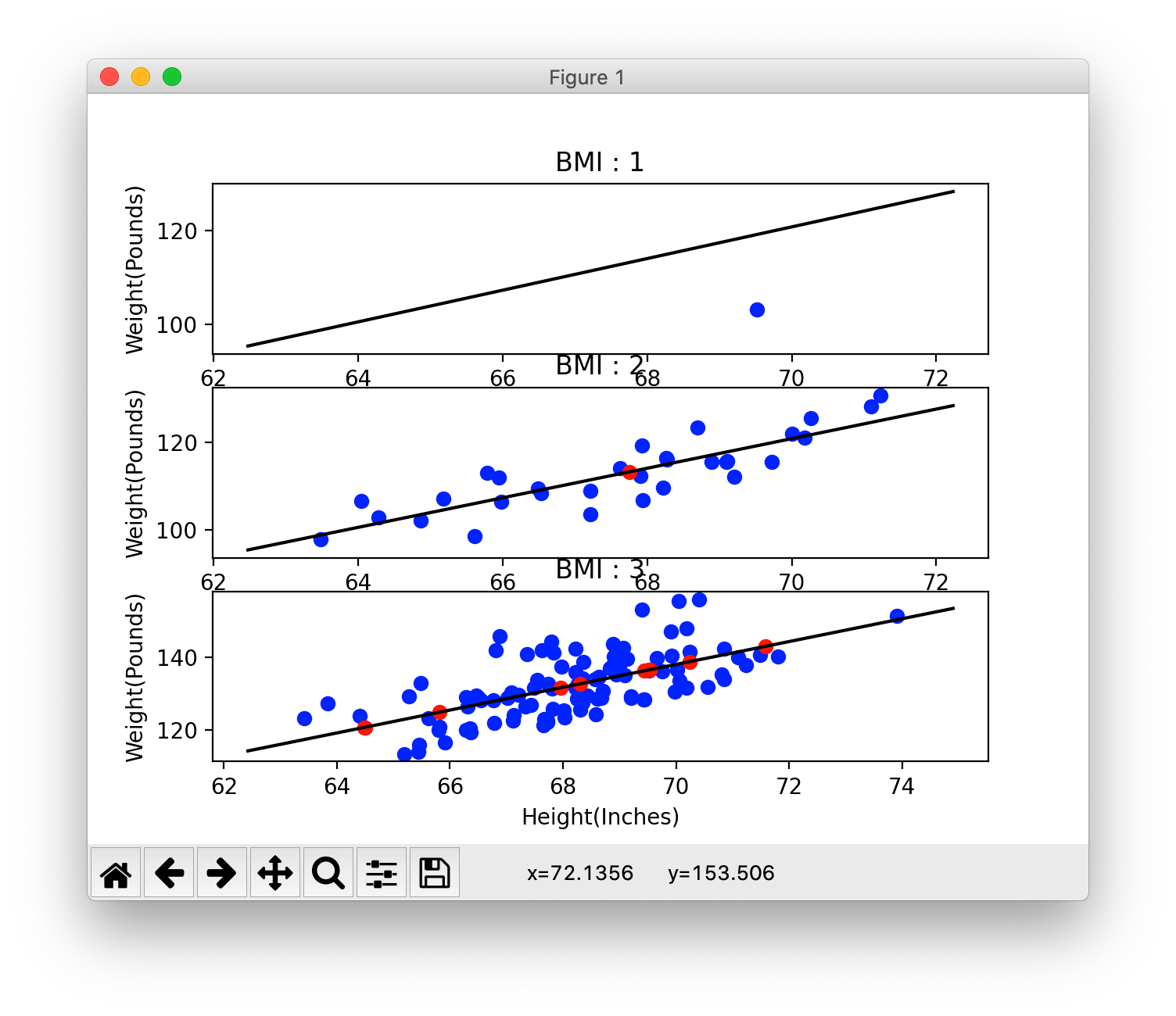
origin = df  
  
df = df.loc[df['Sex']=='Female']  
temp = df  
df = df.dropna()  
  
Height = df["Height (Inches)"]  
Weight = df["Weight (Pounds)"]  
df = temp  
Height2 = df["Height (Inches)"].loc[df["Weight (Pounds)"].isna()]  
Weight2 = df["Weight (Pounds)"].loc[df["Height (Inches)"].isna()]  
wh = df.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh = wh[wh.isnull().any(axis=1)].to\_numpy()  
  
reg = linear\_model.LinearRegression()  
  
reg.fit(Height.values.reshape(-1,1), Weight)  
  
px = np.array([Height.min()-1,Height.max()+1])  
py = reg.predict(px[:,np.newaxis])  
  
for data in wh:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py[0])/reg.coef\_)+px[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg.predict([[data[0]]])  
plt.scatter(Height, Weight,color = "blue")  
plt.scatter(wh[:,0],wh[:,1],color = "red")  
plt.plot(px,py,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("Female")  
plt.show()

origin = df  
  
df = df.loc[df['Sex']=='Male']  
temp = df  
df = df.dropna()  
  
Height = df["Height (Inches)"]  
Weight = df["Weight (Pounds)"]  
df = temp  
Height2 = df["Height (Inches)"].loc[df["Weight (Pounds)"].isna()]  
Weight2 = df["Weight (Pounds)"].loc[df["Height (Inches)"].isna()]  
wh = df.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh = wh[wh.isnull().any(axis=1)].to\_numpy()  
  
reg = linear\_model.LinearRegression()  
  
reg.fit(Height.values.reshape(-1,1), Weight)  
  
px = np.array([Height.min()-1,Height.max()+1])  
py = reg.predict(px[:,np.newaxis])  
  
for data in wh:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py[0])/reg.coef\_)+px[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg.predict([[data[0]]])  
plt.scatter(Height, Weight,color = "blue")  
plt.scatter(wh[:,0],wh[:,1],color = "red")  
plt.plot(px,py,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("Male")  
plt.show()

**Regression(BMI)**

df = df.loc[df['BMI']==1]  
temp = df  
df = df.dropna()  
  
Height = df["Height (Inches)"]  
Weight = df["Weight (Pounds)"]  
df = temp  
wh = df.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh = wh[wh.isnull().any(axis=1)].to\_numpy()  
  
reg = linear\_model.LinearRegression()  
  
reg.fit(Height.values.reshape(-1,1), Weight)  
  
px = np.array([Height.min()-1,Height.max()+1])  
py = reg.predict(px[:,np.newaxis])  
  
for data in wh:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py[0])/reg.coef\_)+px[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg.predict([[data[0]]])  
  
df2 = origin  
df2 = df2.loc[df2['BMI']==2]  
temp = df2  
df2 = df2.dropna()  
  
Height2 = df2["Height (Inches)"]  
Weight2 = df2["Weight (Pounds)"]  
df2 = temp  
wh2 = df2.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh2 = wh2[wh2.isnull().any(axis=1)].to\_numpy()  
  
reg2 = linear\_model.LinearRegression()  
  
reg2.fit(Height2.values.reshape(-1,1), Weight2)  
  
px2 = np.array([Height2.min()-1,Height2.max()+1])  
py2 = reg2.predict(px2[:,np.newaxis])  
  
for data in wh2:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py2[0])/reg.coef\_)+px2[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg2.predict([[data[0]]])

df3 = origin  
df3 = df3.loc[df3['BMI']==3]  
temp = df3  
df3 = df3.dropna()  
  
Height3 = df3["Height (Inches)"]  
Weight3 = df3["Weight (Pounds)"]  
df3 = temp  
wh3 = df3.loc[:,['Height (Inches)','Weight (Pounds)']]  
wh3 = wh3[wh3.isnull().any(axis=1)].to\_numpy()  
  
reg3 = linear\_model.LinearRegression()  
  
reg3.fit(Height3.values.reshape(-1,1), Weight3)  
  
px3 = np.array([Height3.min()-1,Height3.max()+1])  
py3 = reg3.predict(px3[:,np.newaxis])  
  
for data in wh3:  
 if not(data[0]>0 and data[0]<80):  
 data[0]=((data[1]-py3[0])/reg3.coef\_)+px3[0]  
 if not(data[1]>0 and data[1] < 400):  
 data[1]=reg3.predict([[data[0]]])  
  
plt.subplot(311)  
plt.scatter(Height, Weight,color = "blue")  
plt.scatter(wh[:,0],wh[:,1],color = "red")  
plt.plot(px2,py2,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("BMI : 1")  
  
plt.subplot(312)  
plt.scatter(Height2, Weight2,color = "blue")  
plt.scatter(wh2[:,0],wh2[:,1],color = "red")  
plt.plot(px2,py2,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("BMI : 2")  
  
plt.subplot(313)  
plt.scatter(Height3, Weight3,color = "blue")  
plt.scatter(wh3[:,0],wh3[:,1],color = "red")  
plt.plot(px3,py3,color="black")  
plt.xlabel("Height(Inches)")  
plt.ylabel("Weight(Pounds)")  
plt.title("BMI : 3")  
plt.show()

****