IB Mathematics Analysis and Approaches HL

Logistic Regression

A mathematical approach of understanding the factors that influence the likeliness of a person acquiring Coronary Heart Disease

Introduction:

Coronary Heart Disease is the leading cause of death worldwide. Each year cardiovascular deaths cause approximately 17 million deaths accounting for 1/3rd of the deaths worldwide[1]. The primary cause of CHD (Coronary Heart Disease) is the accumulation of fatty material (plaque) in the coronary arteries causing a reduced flow of blood to the heart muscle leading to heart attacks, heart failure etc. Although the Coronary Heart Disease is an emergent health condition, a challenging task till date remains the causes of the disease. Till date we cannot identify and pinpoint the causes of the disease in order to prevent it from causing any further effects on our body. There are several factors such as the lifestyle, eating, smoking habits, age etc. which can cause the disease, however even a survey by the WHO showed that CHD can be predicted only with a **67% accuracy**[2].

In our present times the high usage of automation and computers allow us to access and extrapolate information from data which can be used by the doctors for the accurate prediction. The various causes for the disease got me interested in predicting the likeliness of a person having coronary heart disease in the future.

With a passion in AI and machine learning, I decided to take a statistical approach in predicting the likeliness of a person getting the coronary heart disease based on various factors of a person's health and lifestyle. Therefore the aim for this exploration is "To predict the most important factors that affect the likeliness of a person acquiring the coronary heart disease using logistic regression". I would be first deriving the logistic regression equation, I would then be developing, training and testing a machine learning model to predict the likeliness of a person acquiring the coronary heart disease based on a dataset obtained from the Framingham Heart

[1] Admin. "Coronary Heart Disease." *Health Knowledge*, 27 June 2010, https://www.healthknowledge.org.uk/public-health-textbook/disease-causation-diagnostic/2b-epidemiology-diseases-phs/chronic-diseases/coronary-heart-disease#:~:text=Coronary%20heart%20disease%20is%20now,million%20deaths%20each%20year4.

[2] Mohammad, Rami Mustafa. "(PDF) Prediction of Coronary Heart Disease Using Machine ..." *Prediction of Coronary Heart Disease Using Machine Learning: An Experimental Analysis*, July 2019, https://www.researchgate.net/publication/335094208_Prediction_of_Coronary_Heart_Disease_using_Machine_Learning_An_Experimental_Analysis.

Study[3]. I would be splitting the data for training and testing the data to verify accuracy of the model and then I would be using a confusion matrix to evaluate the logistic regression model that has been developed.

Derivation:

Like linear regression, logistic regression is similar to a linear regression except logistic regression predicts whether something is True or False, instead of predicting continuous data obtained from a linear regression. Logistic regression transforms its output using a logistic sigmoid function to output a probability value that can be mapped onto two different classes[4]. An advantage of using a logistic regression is the ability to classify a data point based on the inputs given. An example would be if the logistic regression model shows a >50% chance of a person having the Coronary Heart Disease, we can classify the person experiencing a coronary heart disease.

As mentioned above logistic regression can be used as a classification algorithm, since the logistic regression uses binary classification, the likely outputs for a given input into the function is outputted as probabilities between 0 and 1 on the y-axis. The logistic regression can be used for data mapped on the following basis.

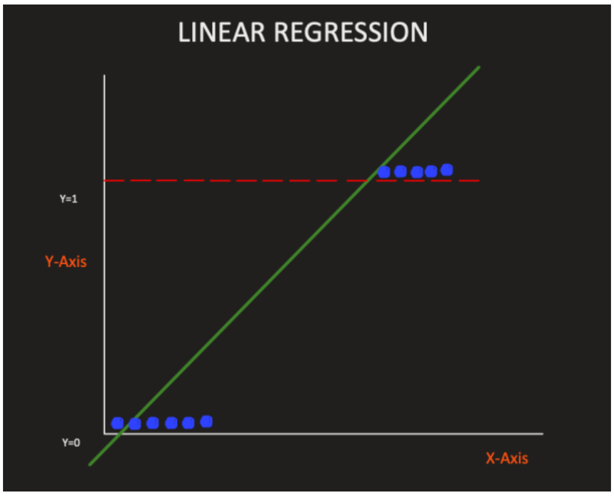


Figure 1: Linear Regression mapped into Binary Classification (Made using OneNote)

[3] "Home." *Framingham Heart Study*, 1 Oct. 2021, https://framinghamheartstudy.org/.

[4] Gupta, Sparsh. "What Makes Logistic Regression a Classification Algorithm?" *Medium*, Towards Data Science, 17 July 2020, https://towardsdatascience.com/what-makes-logistic-regression-a-classification-algorithm-35018497b63f.

If we try to fit a Regression line into a binary classification problem, an input into the linear regression function gives an output of continuous data which is not suitable for classification as certain x values give an output of $y < 0$ and $y > 1$, which is not suitable for the prediction. The Range of the linear regression may lie outside $[0,1]$. Therefore the range and domain of the logistic regression function would be:

$$y\epsilon[0,1]$$
$$x\epsilon R$$

The term Logistic refers to the "log odds" probability that is modelled, the term "odds" is defined as the ratio of the probability that an event occurs to the probability that an event doesn't occur.

$$odds = \frac{P(event)}{1 - P(event)}$$

We can consider the conditional probability $Pr(Y = 1|X)$ abbreviated as $p(X)$. As mentioned above in the context of $P(Y = 1|X)$, $Y = 1$ is not a number but the class or category that satisfies the classification. In this exploration $Y = 1$ would be the data points for the patients who are likely to have a Coronary Heart Disease. Hence we need to model the probabilities using a curve where the predictor domain is $X\epsilon\boldsymbol{R}$ and the range of $p(X)$ that Y is true given X is between 0 and 1. In order to accomplish this we would be using a sigmoid function which takes in any real value of x and outputs a probability value between 0 and 1.

$$\sigma(t) = \frac{e^t}{e^t + 1} = \frac{1}{1 + e^{-t}}$$

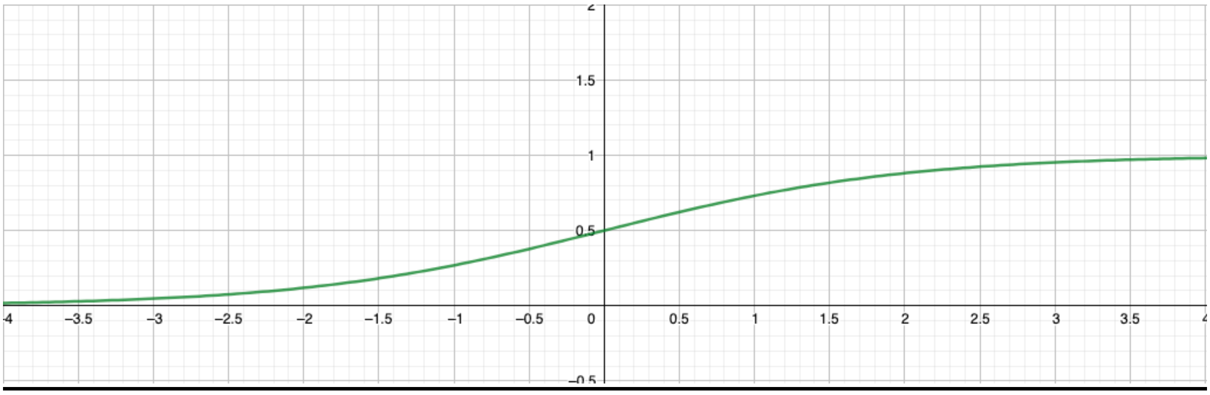


Figure 2: Sigmoid Function Graph (Made using GeoGebra)

Furthermore for any given variable of t, let us consider t as a linear function with one input and one output, with β_0 as the intercept and β_1 as the slope of the function.

$$t = \beta_1 X + \beta_0$$

Substituting in the general logistic function p which gives output values between 0 and 1 gives the equation:

$$p(x) = \sigma(t) = \frac{1}{1 + e^{-(\beta_1 X + \beta_0)}}$$

Assuming that the regression line passes through the origin we can modify the $p(x)$ equation to be:

$$p(x) = \sigma(t) = \frac{1}{1 + e^{-\beta_1 X}}$$

The data separable into two different classes of 0 and 1 can be modelled using a Logistic function for the given variable in a linear function. However the relationship between the input variable x and output probability cannot be easily interpreted from the sigmoid function. Therefore we use a **Logit** or a “log-odd” function for interpreting the relationship in a linear manner.

The logit function can be defined as:

$$g\big(p(x)\big) = \log\left(\frac{p(x)}{1 - p(x)} = \beta X\right)$$

We can also express the logit function of an observation y as a linear function of K input variables of X.

$$\log\left(\frac{p(x)}{1 - p(x)} = \sum_{j=0}^{K} \beta_j x_j\right)$$

$$\frac{p(x)}{1 - p(x)} = e^{\left(\sum_{j=0}^{K} \beta_j x_j\right)}$$

$$= \prod_{j=0}^{K} e^{\beta_j x_j}$$

g → logit function

$p(x)$ → probability of a dependent variable lying in 2 classes: either 0 or 1, given a linear combination of predictors

β_0 → Intercept in the regression equation

β → The slope of the regression equation multiplied by some value of the predictor

From the dataset we would be working a dataset that has m observations and n features. We will have m row vectors of X_i. The y values only can take the classes of "0" and "1", where 0 are the patients who do not have coronary heart disease and 1 are patients who are have coronary heart disease.

$$y\epsilon\{0,1\} \qquad\qquad x_i = [1, x_1, x_2, \ldots, x_n]^T$$

The parameters for the data will be given in a column vector $\hat{\beta}$:

$$\hat{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_n \end{bmatrix}$$

We would be estimating parameters in order to make predictions in the logistic regression model. Like how the least squares method is used for parameter estimation in linear regression and fit a model, the maximum likelihood estimation is used to estimate parameters in logistic regression. Based on the labels of 0 and 1 to each value in the dataset:

- For samples labelled "1", we must estimate $\hat{\beta}$ such that $\widehat{p(x)}$ is as close to 1 as possible.
- For samples labelled "0", we must estimate $\hat{\beta}$ such that $1 - \widehat{p(x)}$ is as close to 1 as possible.

Mathematically for every sample with label 1 we want to estimate $\hat{\beta}$ such that the product of all conditional probabilities of class 1 samples is as close to 1 as possible. After understanding the requirements, we want to find the beta parameters such that the product of both the products is maximum over all elements of the dataset.

$$L(\beta) = \prod_{i=1, y_i=1}^{N} p(x_i) \times \prod_{i=1, y_i=0}^{N} (1 - p(x_i))$$

x_i → The feature vector of the i^{th} sample.

This function that we need to optimize is known as the likelihood function.

$$L(\beta) = \prod_{i=1}^{N} p(x_i)^{y_i} \times \prod_{i=1}^{N} \big(1 - p(x_i)\big)^{1-y_i}$$

We can use the log likelihood function to convert this into a summation:

$$l(\beta) = \sum_{i=1}^{N} y_i \log(p(x_i)) + (1 - y_i)\log\big(1 - p(x_i)\big)$$

$l(\beta)$ → log likelihood

Substituting $p(x)$ exponent form in the equation gives us:

$$l(\beta) = \sum_{i=1}^{N} y_i \log\left(\frac{1}{1 + e^{-\beta x_i}}\right) + (1 - y_i)\log\left(1 - \left(\frac{1}{1 + e^{-\beta x_i}}\right)\right)$$

We now group the coefficients of y_i ,

$$l(\beta) = \sum_{i=1}^{N} y_i \left[\log\left(\frac{1}{1 + e^{-\beta x_i}}\right) - \log\left(\frac{e^{-\beta x_i}}{1 + e^{-\beta x_i}}\right)\right] + \log\left(\frac{e^{-\beta x_i}}{1 + e^{-\beta x_i}}\right)$$

Upon simplification, we get,

$$l(\beta) = \sum_{i=1}^{N} y_i\big[\log\big(e^{\beta x_i}\big)\big] + \log\left(\frac{e^{-\beta x_i}}{1 + e^{-\beta x_i}} \times \frac{e^{\beta x_i}}{e^{\beta x_i}}\right)$$

$$l(\beta) = \sum_{i=1}^{N} y_i \beta x_i + \log\left(\frac{1}{1 + e^{\beta x_i}}\right)$$

$$l(\beta) = \sum_{i=1}^{N} y_i \beta x_i - \log\big(1 + e^{\beta x_i}\ \big)$$

The above is the final form of the log-likelihood function, the goal is to find the value of beta that maximizes the function. This final likelihood equation consists of non-algebraic terms like logarithms and exponents, such equations are known as transcendental equations[5]. The value of such equations cannot be computed exactly and they do not have close formed solutions.

[5] "Transcendental Equation." *THERMOPEDIA*, https://www.thermopedia.com/content/1202/.

However we can use numerical methods to approximate a solution. The numerical method that I would be using is the Newton Raphson[6][7] method.

Using the first two terms of the Taylor series expansion:

$$\nabla_{\beta} l(\beta) = \nabla_{\beta} l(\beta^{*}) + (\beta - \beta^{*}) \nabla_{\beta\beta} l(\beta^{*})$$

$$\nabla_{\beta} l(\beta^{*}) + (\beta - \beta^{*}) \nabla_{\beta\beta} l(\beta^{*}) = 0$$

$$\beta = \beta^{*} - \frac{\nabla_{\beta} l(\beta^{*})}{\nabla_{\beta\beta} l(\beta^{*})}$$

We need to compute this for t iterations then β will converge to the approximate coefficient vector.

$$\beta_{t+1} = \beta^{t} - \frac{\nabla_{\beta} l(\beta^{t})}{\nabla_{\beta\beta} l(\beta^{t})}$$

The Newton Raphson equation involves the computing of the gradient ∇_{β} with respect to β. In order to determine this gradient we bring the gradient symbol into the log likelihood function. We bring the gradient into the summation as, the derivative of the sum is the same as the sum of individual derivatives.

$$\nabla_{\beta} l = \nabla_{\beta} \sum_{i=1}^{N} y_i \beta x_i - \log\left(1 + e^{\beta x_i}\right)$$

$$\nabla_{\beta} l = \sum_{i=1}^{N} \nabla_{\beta}\left[y_i \beta x_i - \log\left(1 + e^{\beta x_i}\right)\right]$$

$$\nabla_{\beta} l = \sum_{i=1}^{N} \nabla_{\beta}[y_i \beta x_i] - \nabla_{\beta}\left[\log\left(1 + e^{\beta x_i}\right)\right]$$

$$\nabla_{\beta} l = \sum_{i=1}^{N} y_i x_i - \left[\frac{1}{1 + e^{\beta x_i}} e^{\beta x_i} x_i\right]$$

By multiplying and dividing the second term with $e^{-\beta x_i}$ we get the following equation:

$$\nabla_{\beta} l = \sum_{i=1}^{N} y_i x_i - \left[\frac{1}{1 + e^{-\beta x_i}} x_i\right]$$

[6] "Newton Raphson Method." *Brilliant Math & Science Wiki*, https://brilliant.org/wiki/newton-raphson-method/.

[7] "Logistic Regression - CMU Statistics." *Logistic Regression*, 2015, https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch12.pdf.

Replacing the exponent term with the probability formula defined earlier:

$$\nabla_{\beta} l = \sum_{i=1}^{N} y_i x_i - [p(x_i) x_i\ ]$$

By taking x_i common we get the final log-likelihood gradient:

$$\nabla_{\beta} l = \sum_{i=1}^{N} [y_i - p(x_i)] x_i$$

The gradient found above can just be substituted into the numerator of the Newton Raphson equation, we must find the Hessian Matrix or $\nabla_{\beta\beta} l$ which is a matrix of second order derivatives with respect to β coefficients. Due to its complexity it is not required to know what is the Hessian Matrix, but we must find an equation to compute the Hessian Matrix in order to substitute it back into the Newton Raphson equation. The second derivative or $\nabla_{\beta\beta} l$ can be represented as the gradient of the first derivative $\nabla_{\beta} l$.

$$\nabla_{\beta\beta} l = \nabla_{\beta}(\nabla_{\beta} l)$$

$$\nabla_{\beta\beta} l = \nabla_{\beta}\left(\sum_{i=1}^{N} [y_i - p(x_i)] x_i\right)$$

We bring the gradient into the summation and remove y_i as it is independent of β.

$$\nabla_{\beta\beta} l = \sum_{i=1}^{N} \nabla_{\beta} [y_i - p(x_i)] x_i$$

$$\nabla_{\beta\beta} l = \sum_{i=1}^{N} \nabla_{\beta} - p(x_i) x_i$$

Replacing $p(x_i)$ with its β equivalent gives us:

$$\nabla_{\beta\beta} l = \sum_{i=1}^{N} \nabla_{\beta} - \left[\frac{1}{1 + e^{-\beta x_i}}\right] x_i$$

Applying the gradient to the matrix results in:

$$\nabla_{\beta\beta} l = \sum_{i=1}^{N} \left[\frac{1}{1 + e^{-\beta x_i}}\right]^2 e^{-\beta x_i}(-x_i) x_i$$

$$\nabla_{\beta\beta} l = -\sum_{i=1}^{N} \left[\frac{e^{-\beta x_i}}{1 + e^{-\beta x_i}}\right] \left[\frac{1}{1 + e^{-\beta x_i}}\right] {x_i}^T x_i$$

$$\nabla_{\beta\beta} l = -\sum_{i=1}^{N} p(x_i)(1 - p(x_i) x_i^T x_i$$

Now that we have the gradient vector $\nabla_\beta l$ and the Hessian Matrix $\nabla_{\beta\beta} l$, we must convert them both into their matrix representation.

$$\nabla_\beta l = \sum_{i=1}^{N} [y_i - p(x_i)] x_i$$

$$\nabla_\beta l = X^T (Y - \hat{Y})$$

Matrix Representation of Hessian Matrix:

$$\nabla_{\beta\beta} l = -\sum_{i=1}^{N} p(x_i)(1 - p(x_i) x_i^T x_i$$

$$\nabla_{\beta\beta} l = -X^T P(1 - P) X$$

If we consider $P(1 - P)$ as the diagonal matrix W. Then the Hessian Matrix becomes:

$$\nabla_{\beta\beta} l = -X^T W X$$

By substituting these two terms into the Newton Raphson equation we get the final equation:

$$\beta_{t+1} = \beta^t - \frac{\nabla_\beta l(\beta^t)}{\nabla_{\beta\beta} l(\beta^t)}$$

$$\beta^{(t+1)} = \beta^{(t)} + \left(X^T W^{(t)} X\right)^{-1} X^T \left(Y - \hat{Y}^{(t)}\right)$$

Now we just have to execute this for ‘t iterations’ until the value converges and we get the maximum β coefficients. Once the beta coefficient have been estimated we can substitute them back into the $p(x)$ equation to determine the probability of a data point belonging to a particular class. In our case if $p(x) > 0.5$ we consider the data point to belong to the class of “1” else if $p(x) < 0.5$ we consider the data point to belong to the class of “0”.

$$p(X) = \sigma(t) = \frac{1}{1 + e^{-\beta_1 X}}$$

Application of Logistic regression:

From the dataset that I have obtained from the Framingham Heart Study, there are multiple input/independent variable based on which the logistic regression model computes the probabilities and classifies the patients. The following are the input variables given in the dataset:

Variable	**About the Variable**	**Data**
Sex	Male or Female	Nominal
Age	Age of the patient	Continuous
currentSmoker	Whether or not the patient is currently a smoker	Nominal
cigsPerDay	Average number of cigarettes smoked in one day	Continuous
BPMeds	Whether or not the patient was on blood pressure medication	Nominal
prevalentStroke	Whether or not the patient previously had a heart Stroke	Nominal
prevalentHyp	Whether or not the patient was in Hypertension	Nominal
Diabetes	Whether or not the patient had diabetes	Nominal
totalChol	Total cholesterol level	Continuous
sysBP	Systolic Blood Pressure	Continuous
diaBP	Diastolic Blood Pressure	Continuous
BMI	Body Mass Index	Continuous
heartrate	Heart Rate	Continuous
glucose	Glucose level	Continuous
TenYearCHD	10 year risk of having CHD	Binary (1: Yes, 0: No)

Table 1: Variables in the Dataset

Since this is a dataset with thousands of values, computing the probabilities and classifying the data would be an impossible task to do manually. Therefore I would be developing a machine learning model that uses Python and various libraries to visualize and determine the factors that influence a person having coronary heart disease. The code used to develop the logistic regression model can be found in Appendix 1.

First I would be visualizing the data using Matplotlib[8] library to draw histograms that show the distribution of each input variable with respect to the number of patients in the dataset.

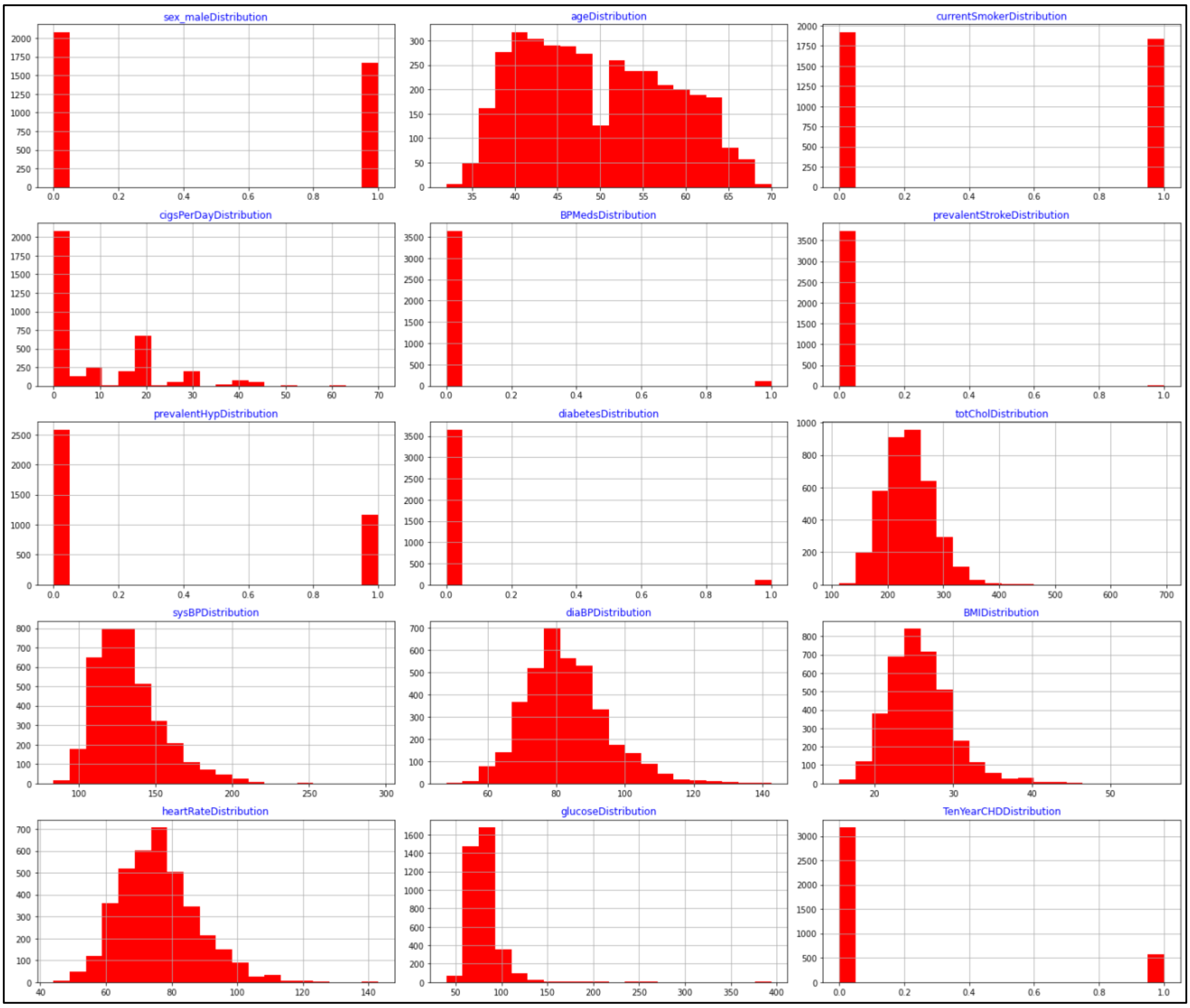


Figure 3: Histogram showing distribution of each input variable

The next part of the data is to develop an understanding of the demographics of patients who are likely to a coronary heart disease using the TenYearCHD binary values provided in the dataset. For that firstly I would be plotting a graph that shows the number of patients who are likely to have Coronary Heart Disease.

[8] “Visualization with Python¶.” *Matplotlib*, https://matplotlib.org/.

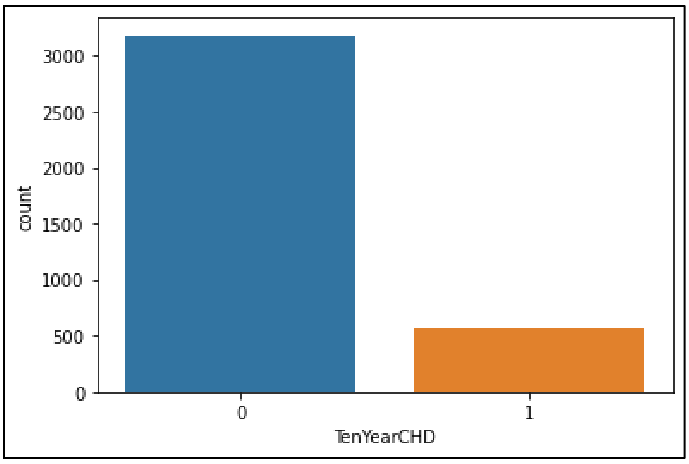


Figure 4: TenYearCHD vs Patient count graph

From the dataset we can conclude that there are 3179 patients who don't have a Ten Year risk of coronary heart disease and there are 572 patients with Ten Year risk of coronary heart disease. I also want to understand the age demographics of the patients who are likely to have the Ten Year Coronary heart disease risk, I would again be plotting a double bar graph that shows the above relationship.

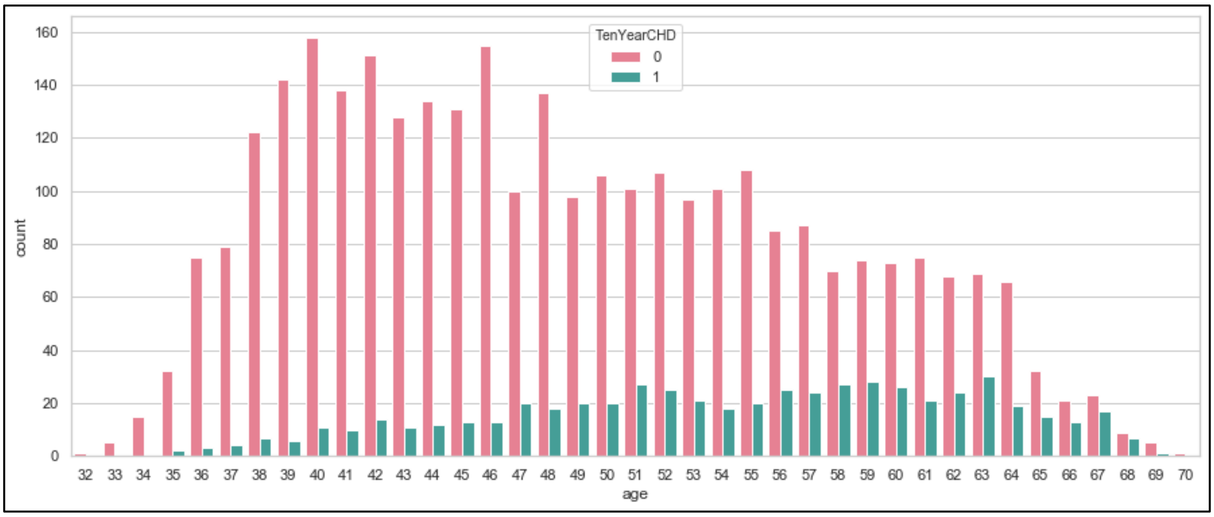
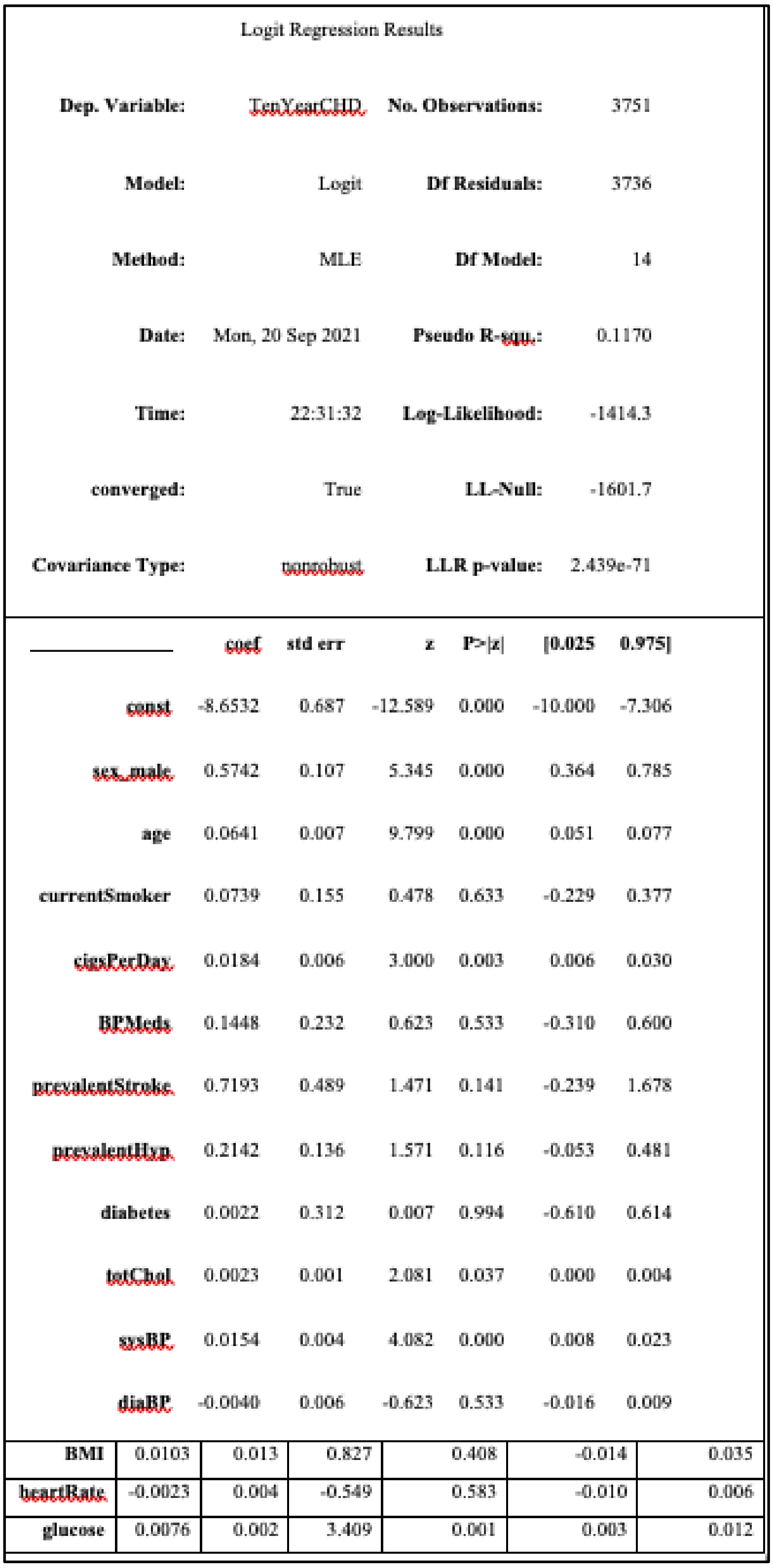


Figure 5: Age and TenYearCHD vs number of patients bar graph

From the above graph, we can infer that patients between the ages of 51 and 63 are most likely to have a coronary heart disease. After observing the results I thought, what if we are able to determine the relationship between each parameter and the risk of having CHD. Proceeding with the logistic regression, I would be running the dataset through a logit function first to determine the relationship between each parameter and the Ten Year risk of having CHD.



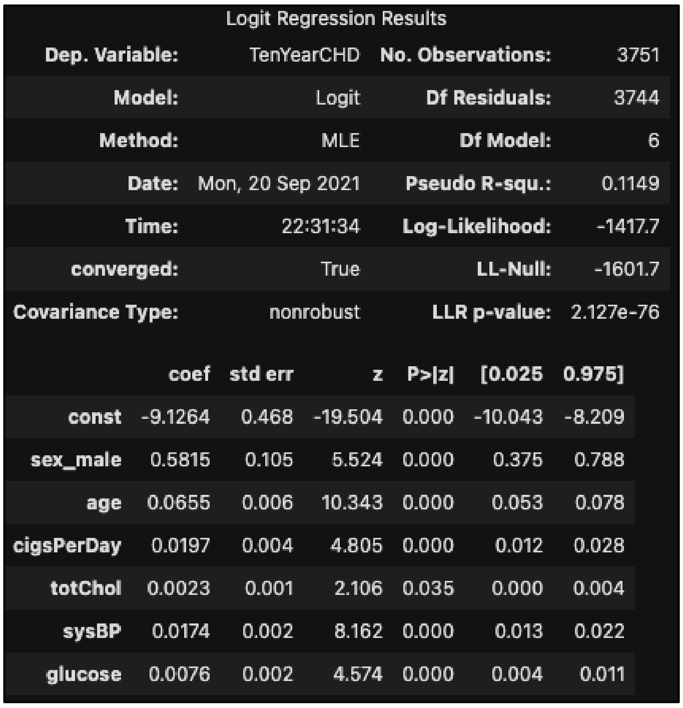
Logit Regression Results

Dep. Variable:	TenYearCHD	**No. Observations:**	3751
Model:	Logit	**Df Residuals:**	3736
Method:	MLE	**Df Model:**	14
Date:	Mon, 20 Sep 2021	**Pseudo R-squ.:**	0.1170
Time:	22:31:32	**Log-Likelihood:**	-1414.3
converged:	True	**LL-Null:**	-1601.7
Covariance Type:	nonrobust	**LLR p-value:**	2.439e-71

	coef	**std err**	**z**	**P>|z|**	**[0.025**	**0.975]**
const	-8.6532	0.687	-12.589	0.000	-10.000	-7.306
sex_male	0.5742	0.107	5.345	0.000	0.364	0.785
age	0.0641	0.007	9.799	0.000	0.051	0.077
currentSmoker	0.0739	0.155	0.478	0.633	-0.229	0.377
cigsPerDay	0.0184	0.006	3.000	0.003	0.006	0.030
BPMeds	0.1448	0.232	0.623	0.533	-0.310	0.600
prevalentStroke	0.7193	0.489	1.471	0.141	-0.239	1.678
prevalentHyp	0.2142	0.136	1.571	0.116	-0.053	0.481
diabetes	0.0022	0.312	0.007	0.994	-0.610	0.614
totChol	0.0023	0.001	2.081	0.037	0.000	0.004
sysBP	0.0154	0.004	4.082	0.000	0.008	0.023
diaBP	-0.0040	0.006	-0.623	0.533	-0.016	0.009
BMI	0.0103	0.013	0.827	0.408	-0.014	0.035
heartRate	-0.0023	0.004	-0.549	0.583	-0.010	0.006
glucose	0.0076	0.002	3.409	0.001	0.003	0.012

Figure 6: Logistic Regression Results

From the results of the logistic regression we can observe that some input variables have a P[z] value greater than the preferred 5% showing that some input variables have a less significant relationship with the probability of having CHD. To find a solution to this bug, I would have to eliminate the variables that have a P-value higher than 5% one at a time and repeat the logistic regression until all the input variables have a P-value less than 5%. This is known as the Backward elimination method. After conducting the Backward elimination, the following are the results of the Logistic Regression.



```
                          Logit Regression Results
     Dep. Variable:        TenYearCHD   No. Observations:       3751
            Model:              Logit       Df Residuals:       3744
           Method:                MLE           Df Model:          6
             Date:   Mon, 20 Sep 2021     Pseudo R-squ.:     0.1149
             Time:           22:31:34     Log-Likelihood:    -1417.7
        converged:               True            LL-Null:    -1601.7
  Covariance Type:          nonrobust        LLR p-value:  2.127e-76

                coef  std err        z  P>|z|   [0.025  0.975]
       const  -9.1264    0.468  -19.504  0.000  -10.043  -8.209
    sex_male   0.5815    0.105    5.524  0.000    0.375   0.788
         age   0.0655    0.006   10.343  0.000    0.053   0.078
  cigsPerDay   0.0197    0.004    4.805  0.000    0.012   0.028
     totChol   0.0023    0.001    2.106  0.035    0.000   0.004
       sysBP   0.0174    0.002    8.162  0.000    0.013   0.022
     glucose   0.0076    0.002    4.574  0.000    0.004   0.011
```

Figure 7: Optimized Logistic Regression Results

After running the Backward elimination method, the input variables for which the P-value is less than 5% are:

Variable	About the Variable	Data
Sex	Male or Female	Nominal
Age	Age of the patient	Continuous
cigsPerDay	Average number of cigarettes smoked in one day	Continuous
totalChol	Total cholesterol level	Continuous
sysBP	Systolic Blood Pressure	Continuous
glucose	Glucose level	Continuous

Table 2: Relevant Variables

Earlier we had defined that the probabilities from a logistic regression can be obtained from the equation:

$$p(x) = \sigma(x) = \frac{1}{1 + e^{-\beta_1 X}}$$

The equation can also be written as:

$$p(x) = \frac{e^{\beta_1 x}}{1 + e^{\beta_1 x}}$$

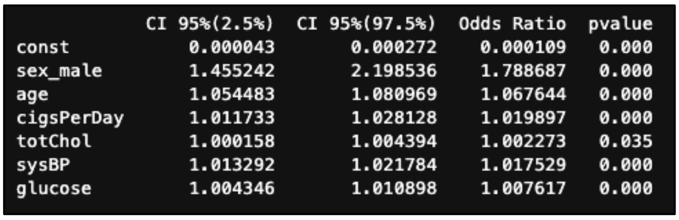
If the regression equation $e^{\beta_1 x}$ has multiple input variables, we can modify the equation as:

$$p(x) = \frac{e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots}}{1 + e^{\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots}}$$

Hence the logit of this $p(x)$ equation is going to be:

$$logit(p) = \log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 * sex + \beta_2 * age + \beta_3 * cigsperDay + \beta_4 * totChol + \beta_5 * sysBp + \beta_6 * glucose$$

In order to determine the relationship between the input variables and the Ten Year CHD risk, I would be using another statistical tool known as the odds ratio, confidence interval and p-value. Odds ratio is a measure of association between the presence or the absence of two variables[9]. Confidence Intervals indicate the degree of uncertainty associated with the odds ratio.



```
             CI 95%(2.5%)  CI 95%(97.5%)  Odds Ratio  pvalue
const            0.000043       0.000272    0.000109   0.000
sex_male         1.455242       2.198536    1.788687   0.000
age              1.054483       1.080969    1.067644   0.000
cigsPerDay       1.011733       1.028128    1.019897   0.000
totChol          1.000158       1.004394    1.002273   0.035
sysBP            1.013292       1.021784    1.017529   0.000
glucose          1.004346       1.010898    1.007617   0.000
```

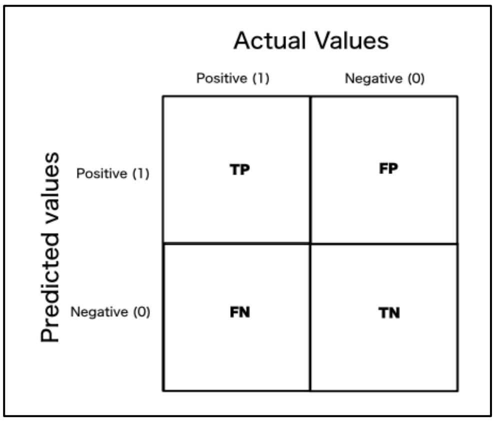
Figure 8: Odds ratio and confidence intervals

- From the above figure we can interpret and gain a lot of insights about the data. In the dataset the "sex" of a person is associated with a binary value where (sex=1) is for **males** and (sex=0) is for **females**. The odds for acquiring CHD for males is $e^{0.5815} = 1.788687$. In terms of percentage, this indicates that the odds for men acquiring CHD is 78.8% more likely than females.
- The likelihood of being diagnosed with CHD with respect to age can be looked at by the increase in age of 1 year. From the odds ratio it is about $e^{0.0655} = 1.067644$. This indicates that an increase in age of 1 year can increase the likelihood of a person acquiring CHD almost by 7%.

[9] Brooks, Steve. "Odds Ratio - Confidence Interval." *Select Statistical Consultants*, 9 Apr. 2020, https://select-statistics.co.uk/calculators/confidence-interval-calculator-odds-ratio/#:~:text=An%20odds%20ratio%20is%20a,or%20absence%20of%20two%20properties.&text=The%20value%20of%20the%20odds,uncertainty%20associated%20with%20that%20ratio.

- The likelihood of being diagnosed with CHD with respect to cigarettes smoked per day can be looked at by the increase in every cigarette that is smoked. From the odds ratio it is about $e^{0.0197} = 1.019895$. This indicates that for every cigarette that you smoke the likelihood of a person acquiring CHD increases by 2%.
- From the odds ratio of systolic blood pressure, the ratio is about $e^{0.0174} = 1.017552$. This indicates that for every unit increase in systolic blood pressure, the likelihood of a person acquiring CHD increases by 1.7%.
- From the odds ratio of glucose, the odds ratio is about $e^{0.0076} = 1.007628$. This indicates that for every unit increase in glucose, the likelihood of a person acquiring CHD increases by 0.7%.

I wondered if the logistic regression model that I have developed gave me accurate results. In order to check the accuracy of the model, I would be using a confusion matrix. Since this exploration is a classification problem, the confusion matrix[10] shows the ways in which the classification problem is confused when it makes the predictions. A confusion matrix shows the 2×2 matrix showing the following factors:



		Actual Values	
Predicted values		Positive (1)	Negative (0)
	Positive (1)	TP	FP
	Negative (0)	FN	TN

Figure 9: Confusion Matrix

- TP – True Positive
- FP – False Positive
- FN – False Negative
- TN – True Negative

[10] Brownlee, Jason. "What Is a Confusion Matrix in Machine Learning." *Machine Learning Mastery*, 14 Aug. 2020, https://machinelearningmastery.com/confusion-matrix-machine-learning/#:~:text=A%20confusion%20matrix%20is%20a%20summary%20of%20prediction%20results%20on,in%20which%20your%20classification%20model.

The accuracy is given by the sum of correct predictions divided by the total number of predictions. From a confusion Matrix it can be determined using the following formula:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

Other measures of performance that can be determined from a confusion matrix are:

Recall: Number of correctly predicted positive classes from all positive classes.

$$Recall = \frac{TP}{TP + FN}$$

Precision: From all classes that have been predicted positive, how many classes are actually positive.

$$Precision = \frac{TP}{TP + FP}$$

Accuracy is more important in a classification when the true negatives and true positive values are more important in the prediction. However when False negatives and false positives are more important, F1 score[11] is a better metric to evaluate a model. In most of the real life classification situations, there is an imbalanced class distribution making F1 score a better metric to evaluate a model.

$$F1\ score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

Using the scikit-learn library, I would be computing the confusion matrix and determining few of the metrics.

[11] Huilgol, Purva. “Accuracy vs. F1-Score.” *Medium*, Analytics Vidhya, 24 Aug. 2019, https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2.

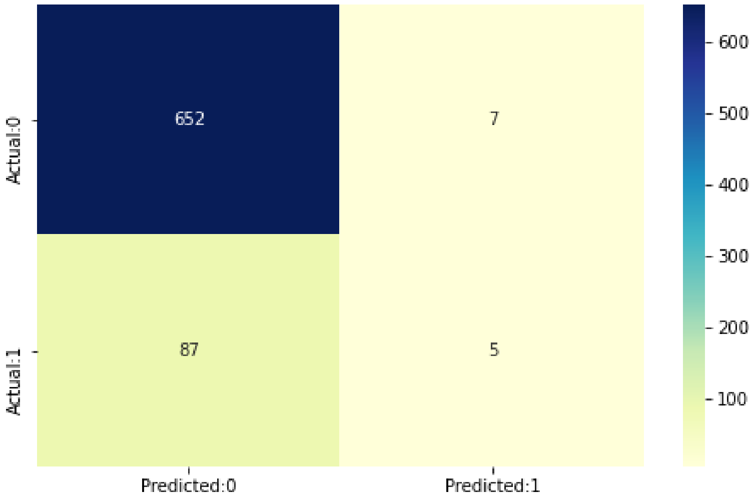


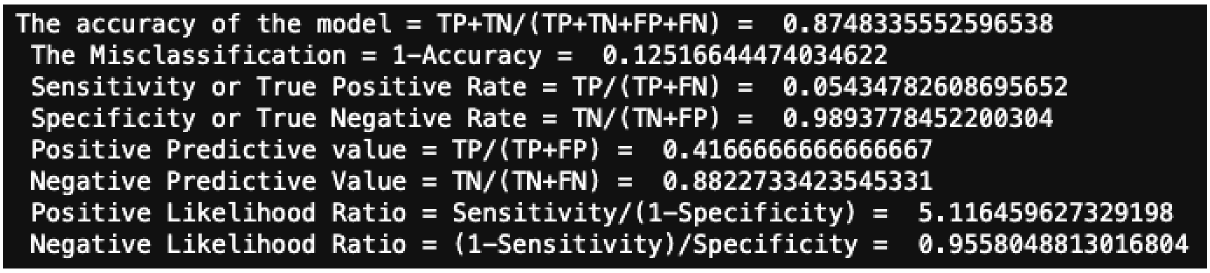
Figure 10: Confusion matrix of the regression model

For any good classification model, the True Negative Rate and True Positive Rate of predictions have to be as close to 100% as possible, and the False Positive and False Negative predictions have they be as close to 0% as possible (they have to be minimized).

From the above confusion matrix we can see that the number of correct predictions are $652 + 5 = 657$ predictions. The number of false predictions are $87 + 7 = 94$ predictions.

- TP = 5
- FP = 7
- FN = 87
- TN = 652

The following are the metrics determined from the confusion matrix:



```
The accuracy of the model = TP+TN/(TP+TN+FP+FN) =  0.8748335552596538
 The Misclassification = 1-Accuracy =  0.12516644474034622
 Sensitivity or True Positive Rate = TP/(TP+FN) =  0.05434782608695652
 Specificity or True Negative Rate = TN/(TN+FP) =  0.9893778452200304
 Positive Predictive value = TP/(TP+FP) =  0.4166666666666667
 Negative Predictive Value = TN/(TN+FN) =  0.8822733423545331
 Positive Likelihood Ratio = Sensitivity/(1-Specificity) =  5.116459627329198
 Negative Likelihood Ratio = (1-Sensitivity)/Specificity =  0.9558048813016804
```

Figure 11: Metrics from a confusion matrix

- Accuracy of the model – 87.4833%
- Misclassification – 12.516%
- True Positivity Rate – 5.4347%
- True Negativity Rate – 98.937%
- Positive Likelihood Ratio – 5.11645
- Negative Likelihood Ratio – 0.9558

Conclusion:

Based on the metrics obtained from the confusion matrix, the model is highly specific and the negative values are predicted more accurately than the positive. In summary I found the most important input variables were determined through an elimination process helping me to determine the most important factors that affect the risk of having Coronary Heart Disease. Earlier I also determined that men are more likely to acquire CHD than women. Increase in age, number of cigarettes per day, systolic Blood pressure show increasing chances of having heart disease. The logistic regression model has predicted with an 87.5% accuracy and this has been validated using the confusion matrix. Through the logistic regression I was able to determine the most significant factors that affected the likeliness of getting the coronary heart disease by eliminating the factors that had a correlation less than 5% with TenYearCHD.

With any model that is being developed, it has its own advantages and disadvantages, especially statistical methods where the scope for application of various tools in a given context is large. The biggest advantage of using logistical regression in that statistical probabilities of correlation between two variables can be considered allowing us to identify the most important factors. Being one of the simplest machine learning algorithms that can be implemented, the model also works with great efficiency and can lead to greater efficiency provided the right kind of dataset is used. The logistic regression model also outputs calibrated probabilities for each parameter in the dataset, this then allows to remove the unwanted parameters and run the model again to get a better understanding of the relationship between each of the parameter.

However there are various disadvantages of implementing a logistic regression as well, the logistic regression only works very efficiently when the dataset is linearly separable. Since it is a classification problem of whether the person is likely to get the coronary heart disease or not, the data is linearly separable whereas this is not true in real life. The classification of the person is predetermined from the dataset given to us, each person is already assigned a value of whether or not they are likely to get the disease. By splitting into the training and testing data, efficiency of the model can be checked and determined to understand the influence of various factors.

The way the research is conducted and the scientific nature of the data collection process will also affect the way in which a logistic regression model performs. There must not be any repetition of the measurements in the dataset, each training data value must be independent of every other data value. There must be no collinearity between the independent variables of the dataset.

Since this investigation is a statistical prediction of getting coronary heart disease, there are various extensions to this investigation of CHD prediction.

1. To what extent does the bifurcation angle due to plaque build-up in the blood vessels, affect the likeliness of getting a Coronary Heart Disease?
2. To determine the relationship between the blood flow and the likeliness of getting CHD due to plaque build-up using calculus.

For such a prevalent disease like the coronary heart disease, through this investigation of statistical analysis, I was able to determine the most important factors that affect the likeliness of getting CHD which will allow people to gain insights about lifestyle choices that can allow people to prevent themselves from getting CHD.

Bibliography

Admin. “Coronary Heart Disease.” *Health Knowledge*, 27 June 2010, https://www.healthknowledge.org.uk/public-health-textbook/disease-causation-diagnostic/2b-epidemiology-diseases-phs/chronic-diseases/coronary-heart-disease#:~:text=Coronary%20heart%20disease%20is%20now,million%20deaths%20each%20year4.

Amay, Omode. “Compare · Amayomode/Heart-Disease-Risk-Prediction.” *GitHub*, 2020, https://github.com/amayomode/Heart-Disease-Risk-Prediction/compare.

Arc. “Derivative of the Sigmoid Function.” *Medium*, Towards Data Science, 17 July 2018, https://towardsdatascience.com/derivative-of-the-sigmoid-function-536880cf918e.

Brooks, Steve. “Odds Ratio - Confidence Interval.” *Select Statistical Consultants*, 9 Apr. 2020, https://select-statistics.co.uk/calculators/confidence-interval-calculator-odds-ratio/#:~:text=An%20odds%20ratio%20is%20a,or%20absence%20of%20two%20properties.&text=The%20value%20of%20the%20odds,uncertainty%20associated%20with%20that%20ratio.

Brownlee, Jason. “What Is a Confusion Matrix in Machine Learning.” *Machine Learning Mastery*, 14 Aug. 2020, https://machinelearningmastery.com/confusion-matrix-machine-learning/#:~:text=A%20confusion%20matrix%20is%20a%20summary%20of%20prediction%20results%20on,in%20which%20your%20classification%20model.

Emporium, Code. “Logistic Regression - the Math You Should Know!” *YouTube*, YouTube, 18 Jan. 2018, https://www.youtube.com/watch?v=YMJtsYIp4kg&t=489s.

Grover, Khushnuma. “Advantages and Disadvantages of Logistic Regression.” *OpenGenus IQ: Computing Expertise & Legacy*, OpenGenus IQ: Computing Expertise & Legacy, 23 June 2020, https://iq.opengenus.org/advantages-and-disadvantages-of-logistic-regression/.

Gupta, Sparsh. “What Makes Logistic Regression a Classification Algorithm?” *Medium*, Towards Data Science, 17 July 2020, https://towardsdatascience.com/what-makes-logistic-regression-a-classification-algorithm-35018497b63f.

Gupta, Sparsh. “What Makes Logistic Regression a Classification Algorithm?” *Medium*, Towards Data Science, 17 July 2020, https://towardsdatascience.com/what-makes-logistic-regression-a-classification-algorithm-35018497b63f.

“Home.” *Framingham Heart Study*, 1 Oct. 2021, https://framinghamheartstudy.org/.

Hornung, Dirk. “Binary Classification with Logistic Regression.” *Medium*, Towards Data Science, 27 Nov. 2019, https://towardsdatascience.com/binary-classification-with-logistic-regression-31b5a25693c4.

Huilgol, Purva. “Accuracy vs. F1-Score.” *Medium*, Analytics Vidhya, 24 Aug. 2019, https://medium.com/analytics-vidhya/accuracy-vs-f1-score-6258237beca2.

Joby, Amal. “What Is Logistic Regression? Learn When to Use It.” *Learn Hub*, 29 July 2021, https://learn.g2.com/logistic-regression.

“Logistic Regression - CMU Statistics.” *Logistic Regression*, 2015, https://www.stat.cmu.edu/~cshalizi/uADA/12/lectures/ch12.pdf.

Mohammad, Rami Mustafa. “(PDF) Prediction of Coronary Heart Disease Using Machine ...” *Prediction of Coronary Heart Disease Using Machine Learning: An Experimental Analysis*, July 2019, https://www.researchgate.net/publication/335094208_Prediction_of_Coronary_Heart_Disease_using_Machine_Learning_An_Experimental_Analysis.

“Newton Raphson Method.” *Brilliant Math & Science Wiki*, https://brilliant.org/wiki/newton-raphson-method/.

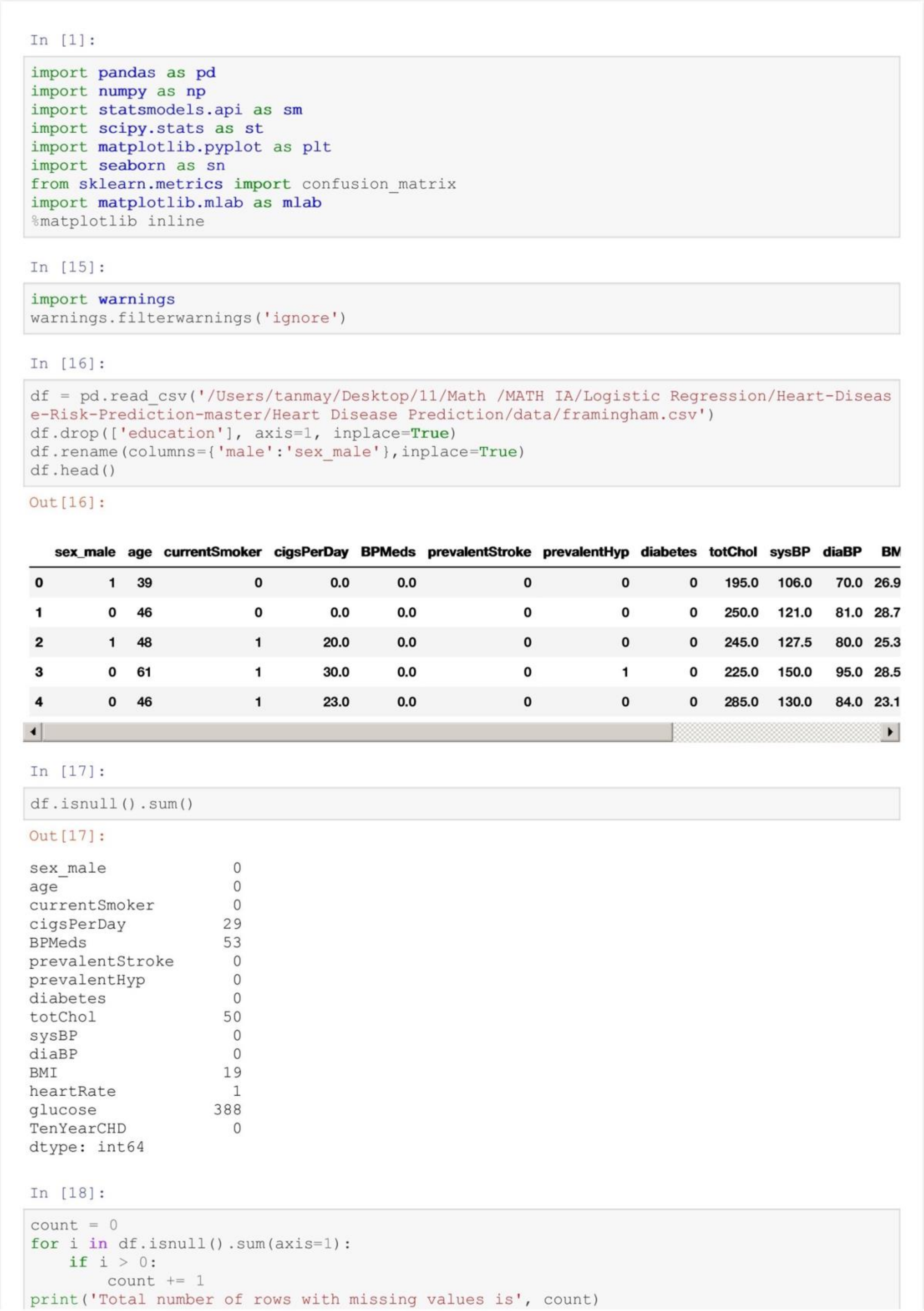
Nissa, Nuzulul Khairu. “How to Predict Coronary Heart Disease Risk Using Logistic Regression?” *Medium*, Analytics Vidhya, 21 Apr. 2021, https://medium.com/analytics-vidhya/how-to-predict-coronary-heart-disease-risk-using-logistic-regression-c069ab95cbec.

“Transcendental Equation.” *THERMOPEDIA*, https://www.thermopedia.com/content/1202/.

“Visualization with Python¶.” *Matplotlib*, https://matplotlib.org/.

Zumel, Nina, et al. “The Simpler Derivation of Logistic Regression.” *Win Vector LLC*, 14 Sept. 2011, https://win-vector.com/2011/09/14/the-simpler-derivation-of-logistic-regression/#:~:text=We%20assume%20that%20the%20case,%E2%80%9D)%20is%20coded%20to%200.&text=Here%2C%20we%20add%20the%20constant,%2C%20the%20name%20logistic%20regression.

Appendix-1: Source code for Machine learning model



In [1]:

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
import scipy.stats as st
import matplotlib.pyplot as plt
import seaborn as sn
from sklearn.metrics import confusion_matrix
import matplotlib.mlab as mlab
%matplotlib inline
```

In [15]:

```
import warnings
warnings.filterwarnings('ignore')
```

In [16]:

```
df = pd.read_csv('/Users/tanmay/Desktop/11/Math /MATH IA/Logistic Regression/Heart-Diseas
e-Risk-Prediction-master/Heart Disease Prediction/data/framingham.csv')
df.drop(['education'], axis=1, inplace=True)
df.rename(columns={'male':'sex_male'},inplace=True)
df.head()
```

Out[16]:

	sex_male	age	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BM
0	1	39	0	0.0	0.0	0	0	0	195.0	106.0	70.0	26.9
1	0	46	0	0.0	0.0	0	0	0	250.0	121.0	81.0	28.7
2	1	48	1	20.0	0.0	0	0	0	245.0	127.5	80.0	25.3
3	0	61	1	30.0	0.0	0	1	0	225.0	150.0	95.0	28.5
4	0	46	1	23.0	0.0	0	0	0	285.0	130.0	84.0	23.1

In [17]:

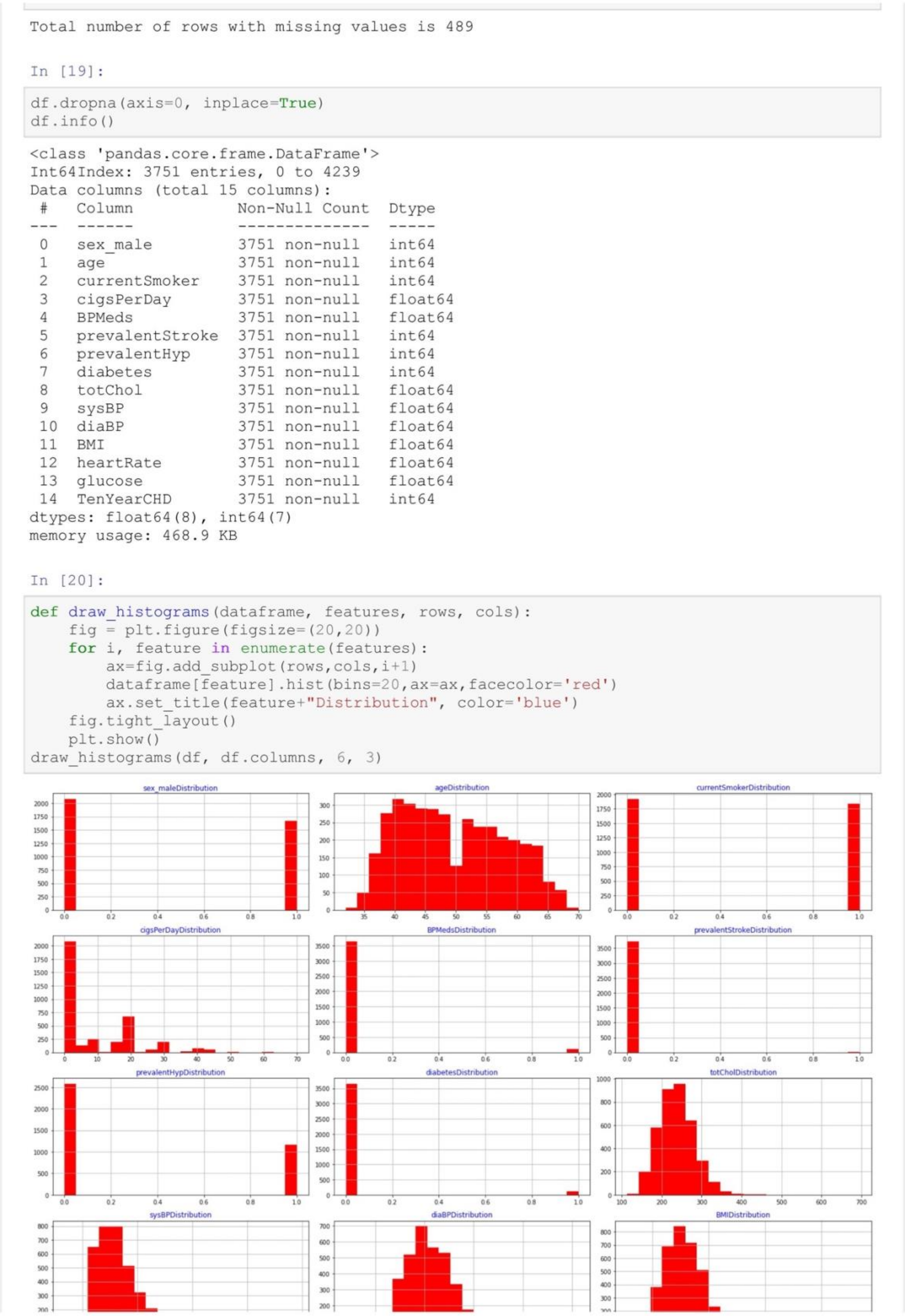
```
df.isnull().sum()
```

Out[17]:

```
sex_male             0
age                  0
currentSmoker        0
cigsPerDay          29
BPMeds              53
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             50
sysBP                0
diaBP                0
BMI                 19
heartRate            1
glucose            388
TenYearCHD           0
dtype: int64
```

In [18]:

```
count = 0
for i in df.isnull().sum(axis=1):
    if i > 0:
        count += 1
print('Total number of rows with missing values is', count)
```



```
Total number of rows with missing values is 489
```

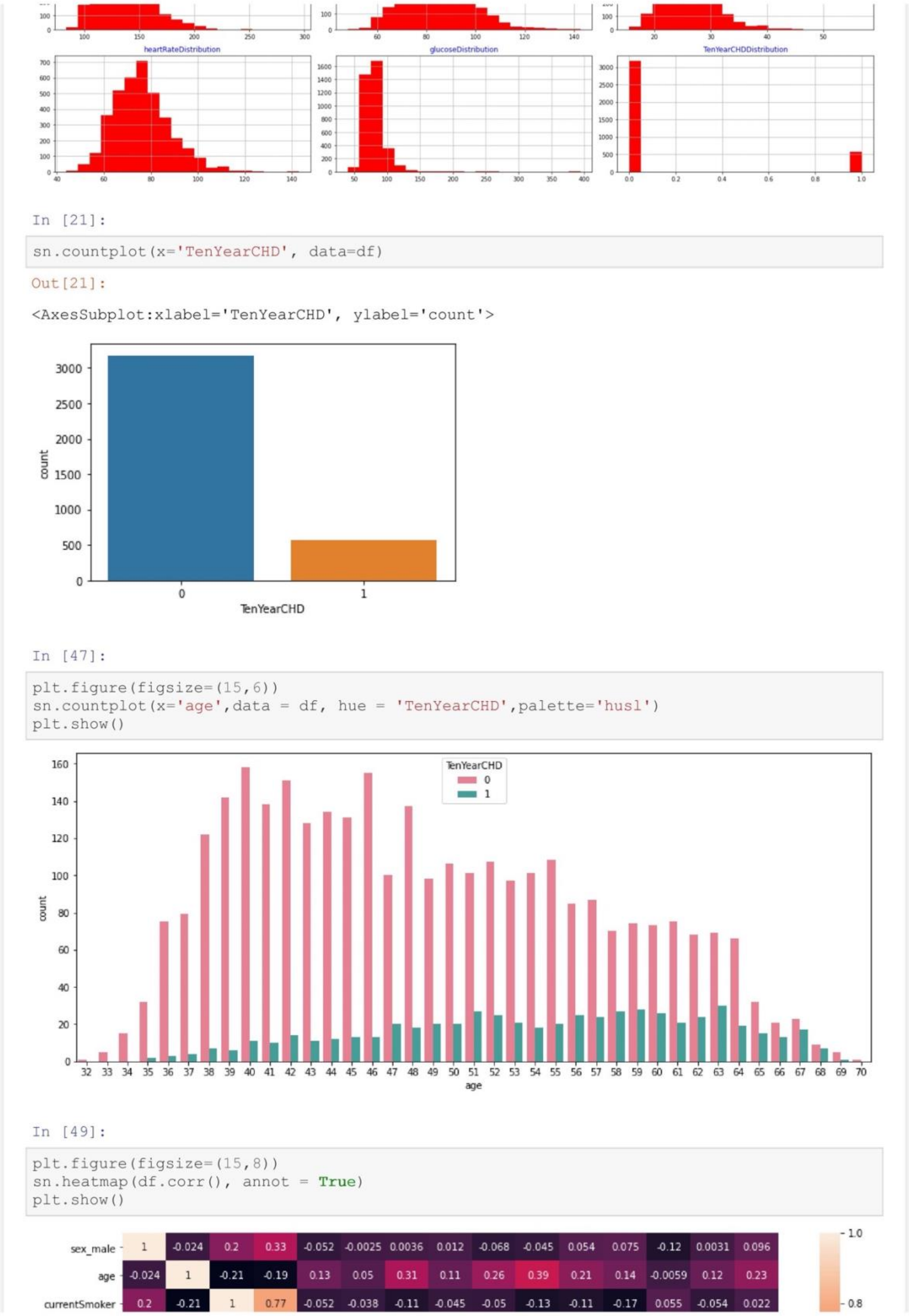
In [19]:

```
df.dropna(axis=0, inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3751 entries, 0 to 4239
Data columns (total 15 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   sex_male         3751 non-null   int64  
 1   age              3751 non-null   int64  
 2   currentSmoker    3751 non-null   int64  
 3   cigsPerDay       3751 non-null   float64
 4   BPMeds           3751 non-null   float64
 5   prevalentStroke  3751 non-null   int64  
 6   prevalentHyp     3751 non-null   int64  
 7   diabetes         3751 non-null   int64  
 8   totChol          3751 non-null   float64
 9   sysBP            3751 non-null   float64
 10  diaBP            3751 non-null   float64
 11  BMI              3751 non-null   float64
 12  heartRate        3751 non-null   float64
 13  glucose          3751 non-null   float64
 14  TenYearCHD       3751 non-null   int64  
dtypes: float64(8), int64(7)
memory usage: 468.9 KB
```

In [20]:

```
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='red')
        ax.set_title(feature+"Distribution", color='blue')
    fig.tight_layout()
    plt.show()
draw_histograms(df, df.columns, 6, 3)
```



In [21]:

```
sn.countplot(x='TenYearCHD', data=df)
```

Out[21]:

```
<AxesSubplot:xlabel='TenYearCHD', ylabel='count'>
```

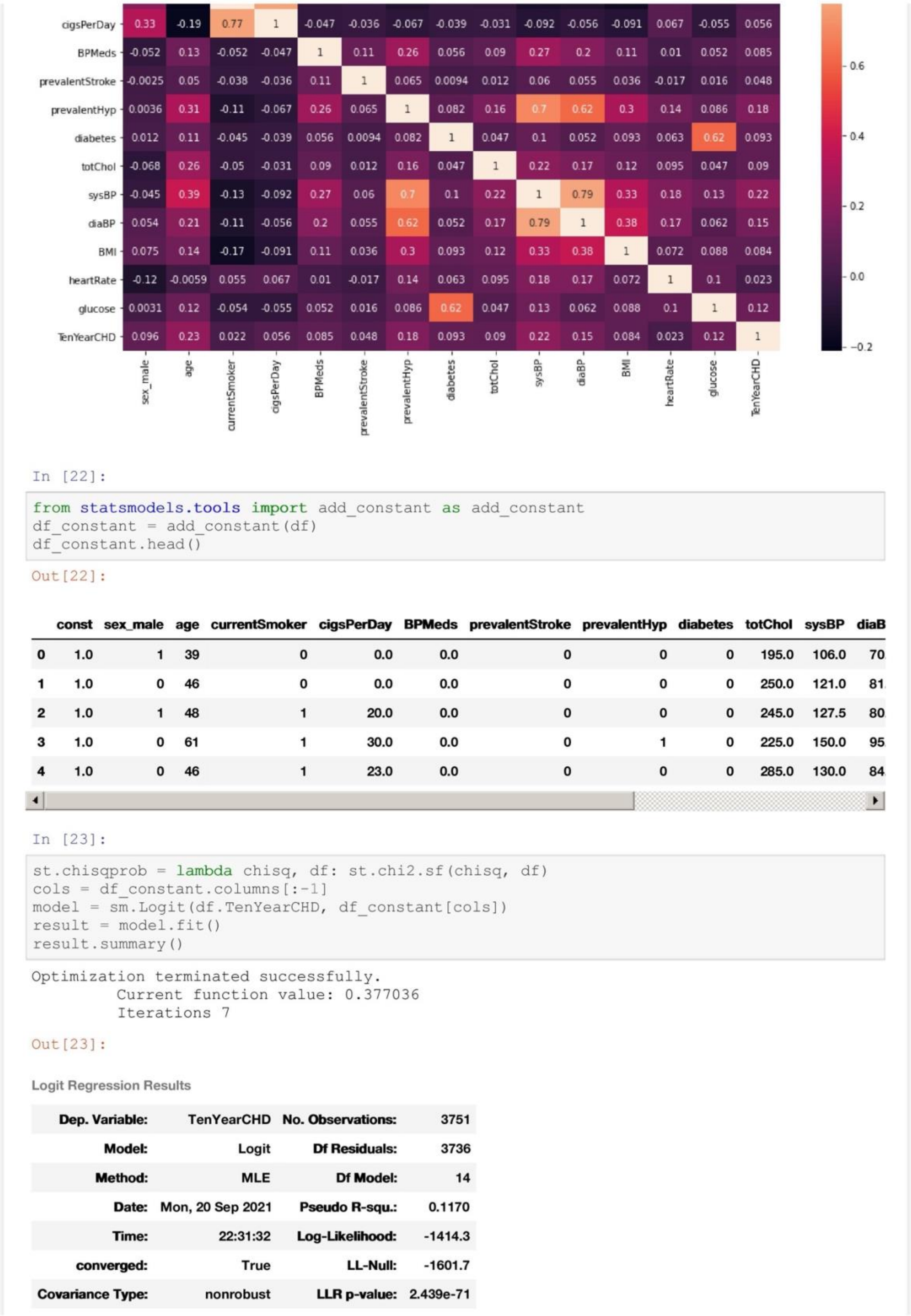
In [47]:

```
plt.figure(figsize=(15,6))
sn.countplot(x='age',data = df, hue = 'TenYearCHD',palette='husl')
plt.show()
```

In [49]:

```
plt.figure(figsize=(15,8))
sn.heatmap(df.corr(), annot = True)
plt.show()
```

sex_male 1 -0.024 0.2 0.33 -0.052 -0.0025 0.0036 0.012 -0.068 -0.045 0.054 0.075 -0.12 0.0031 0.096
age -0.024 1 -0.21 -0.19 0.13 0.05 0.31 0.11 0.26 0.39 0.21 0.14 -0.0059 0.12 0.23
currentSmoker 0.2 -0.21 1 0.77 -0.052 -0.038 -0.11 -0.045 -0.05 -0.13 -0.11 -0.17 0.055 -0.054 0.022
1.0
0.8



	sex_male	age	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
cigsPerDay	0.33	-0.19	0.77	1	-0.047	-0.036	-0.067	-0.039	-0.031	-0.092	-0.056	-0.091	0.067	-0.055	0.056
BPMeds	-0.052	0.13	-0.052	-0.047	1	0.11	0.26	0.056	0.09	0.27	0.2	0.11	0.01	0.052	0.085
prevalentStroke	-0.0025	0.05	-0.038	-0.036	0.11	1	0.065	0.0094	0.012	0.06	0.055	0.036	-0.017	0.016	0.048
prevalentHyp	0.0036	0.31	-0.11	-0.067	0.26	0.065	1	0.082	0.16	0.7	0.62	0.3	0.14	0.086	0.18
diabetes	0.012	0.11	-0.045	-0.039	0.056	0.0094	0.082	1	0.047	0.1	0.052	0.093	0.063	0.62	0.093
totChol	-0.068	0.26	-0.05	-0.031	0.09	0.012	0.16	0.047	1	0.22	0.17	0.12	0.095	0.047	0.09
sysBP	-0.045	0.39	-0.13	-0.092	0.27	0.06	0.7	0.1	0.22	1	0.79	0.33	0.18	0.13	0.22
diaBP	0.054	0.21	-0.11	-0.056	0.2	0.055	0.62	0.052	0.17	0.79	1	0.38	0.17	0.062	0.15
BMI	0.075	0.14	-0.17	-0.091	0.11	0.036	0.3	0.093	0.12	0.33	0.38	1	0.072	0.088	0.084
heartRate	-0.12	-0.0059	0.055	0.067	0.01	-0.017	0.14	0.063	0.095	0.18	0.17	0.072	1	0.1	0.023
glucose	0.0031	0.12	-0.054	-0.055	0.052	0.016	0.086	0.62	0.047	0.13	0.062	0.088	0.1	1	0.12
TenYearCHD	0.096	0.23	0.022	0.056	0.085	0.048	0.18	0.093	0.09	0.22	0.15	0.084	0.023	0.12	1

In [22]:

```
from statsmodels.tools import add_constant as add_constant
df_constant = add_constant(df)
df_constant.head()
```

Out[22]:

	const	sex_male	age	currentSmoker	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaB
0	1.0	1	39	0	0.0	0.0	0	0	0	195.0	106.0	70
1	1.0	0	46	0	0.0	0.0	0	0	0	250.0	121.0	81.
2	1.0	1	48	1	20.0	0.0	0	0	0	245.0	127.5	80.
3	1.0	0	61	1	30.0	0.0	0	1	0	225.0	150.0	95.
4	1.0	0	46	1	23.0	0.0	0	0	0	285.0	130.0	84.

In [23]:

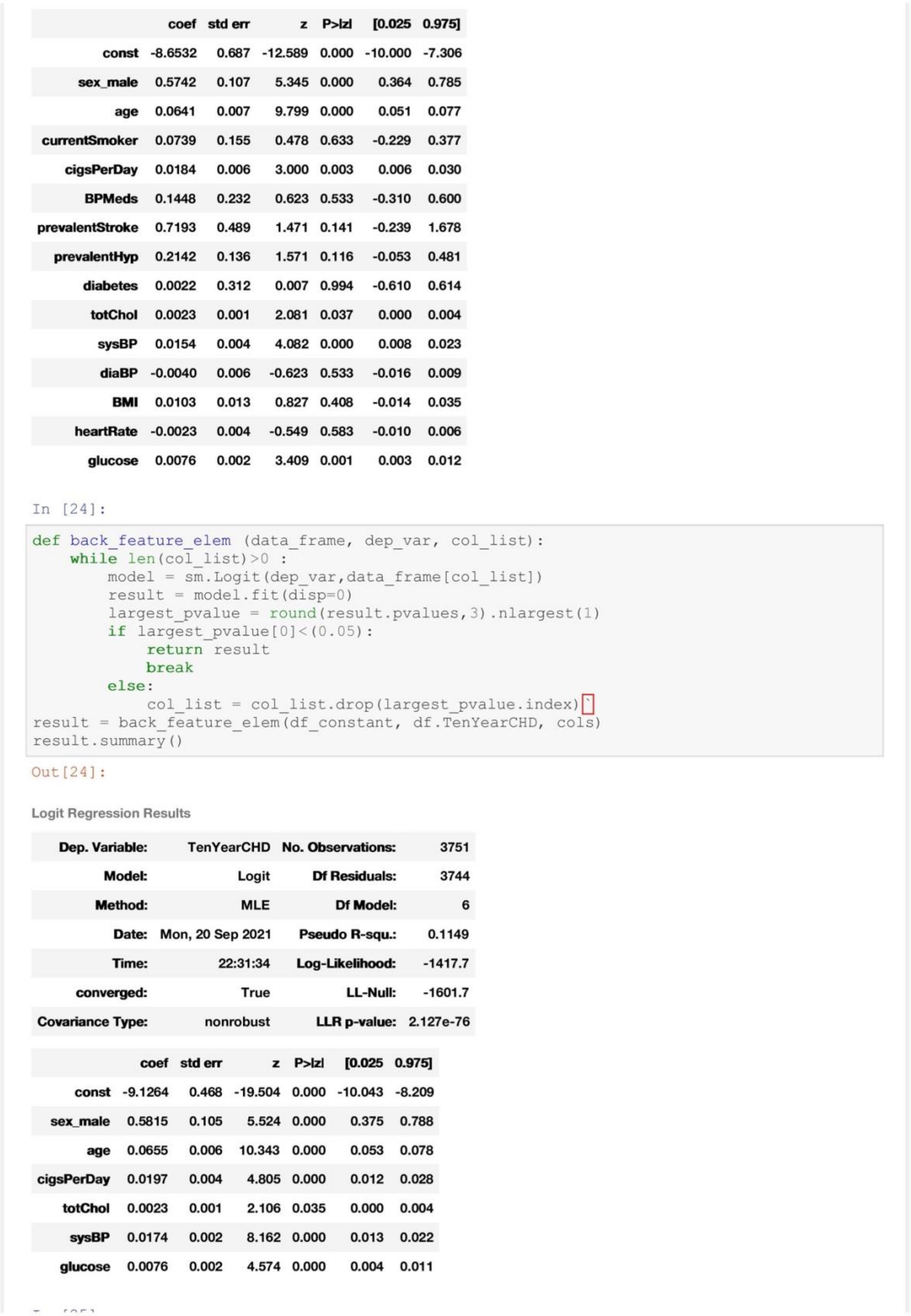
```
st.chisqprob = lambda chisq, df: st.chi2.sf(chisq, df)
cols = df_constant.columns[:-1]
model = sm.Logit(df.TenYearCHD, df_constant[cols])
result = model.fit()
result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.377036
         Iterations 7
```

Out[23]:

Logit Regression Results

Dep. Variable:	TenYearCHD	**No. Observations:**	3751
Model:	Logit	**Df Residuals:**	3736
Method:	MLE	**Df Model:**	14
Date:	Mon, 20 Sep 2021	**Pseudo R-squ.:**	0.1170
Time:	22:31:32	**Log-Likelihood:**	-1414.3
converged:	True	**LL-Null:**	-1601.7
Covariance Type:	nonrobust	**LLR p-value:**	2.439e-71



	coef	std err	z	P>|z|	[0.025	0.975]
const	-8.6532	0.687	-12.589	0.000	-10.000	-7.306
sex_male	0.5742	0.107	5.345	0.000	0.364	0.785
age	0.0641	0.007	9.799	0.000	0.051	0.077
currentSmoker	0.0739	0.155	0.478	0.633	-0.229	0.377
cigsPerDay	0.0184	0.006	3.000	0.003	0.006	0.030
BPMeds	0.1448	0.232	0.623	0.533	-0.310	0.600
prevalentStroke	0.7193	0.489	1.471	0.141	-0.239	1.678
prevalentHyp	0.2142	0.136	1.571	0.116	-0.053	0.481
diabetes	0.0022	0.312	0.007	0.994	-0.610	0.614
totChol	0.0023	0.001	2.081	0.037	0.000	0.004
sysBP	0.0154	0.004	4.082	0.000	0.008	0.023
diaBP	-0.0040	0.006	-0.623	0.533	-0.016	0.009
BMI	0.0103	0.013	0.827	0.408	-0.014	0.035
heartRate	-0.0023	0.004	-0.549	0.583	-0.010	0.006
glucose	0.0076	0.002	3.409	0.001	0.003	0.012

In [24]:

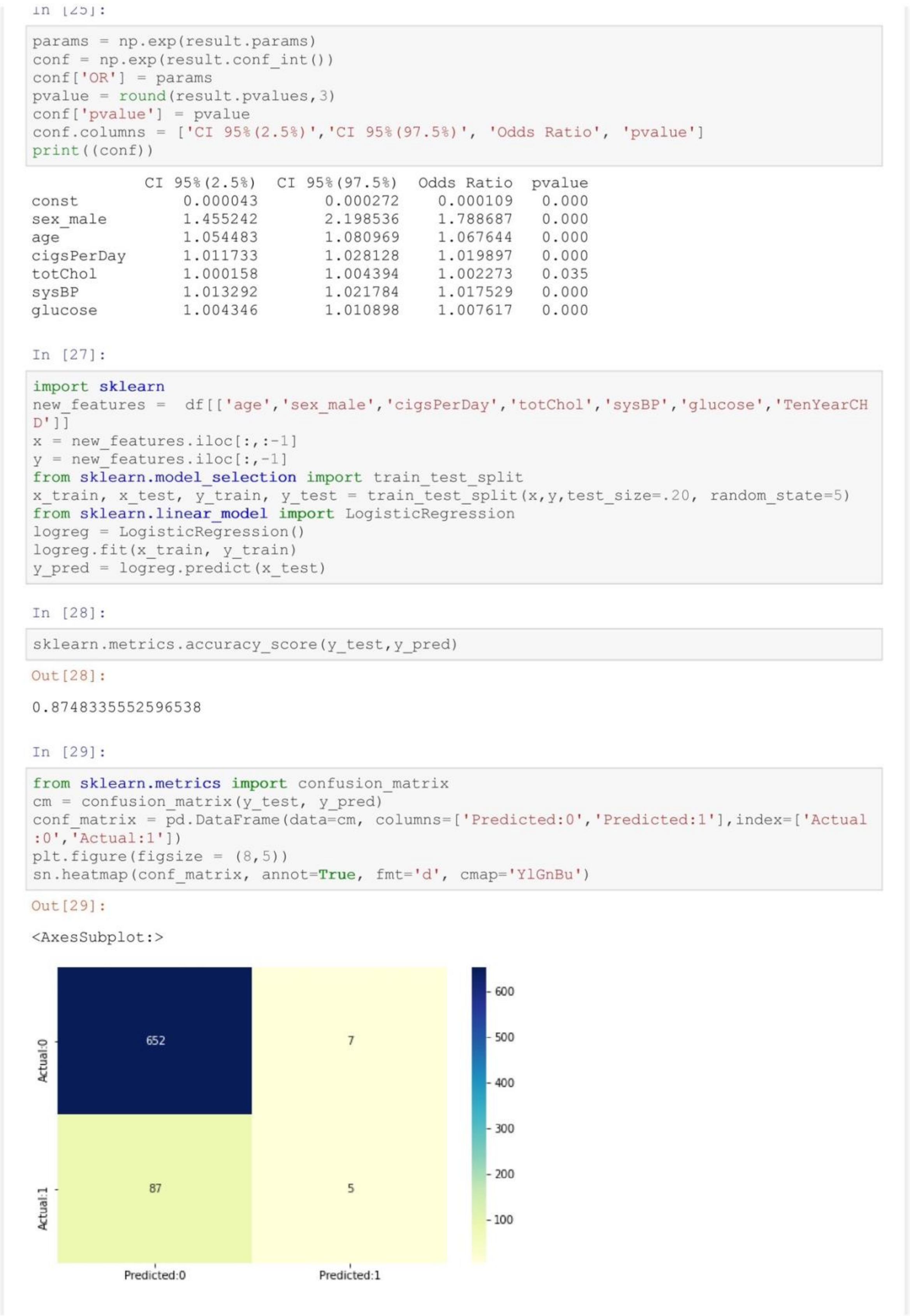
```
def back_feature_elem (data_frame, dep_var, col_list):
    while len(col_list)>0 :
        model = sm.Logit(dep_var,data_frame[col_list])
        result = model.fit(disp=0)
        largest_pvalue = round(result.pvalues,3).nlargest(1)
        if largest_pvalue[0]<(0.05):
            return result
            break
        else:
            col_list = col_list.drop(largest_pvalue.index)`
result = back_feature_elem(df_constant, df.TenYearCHD, cols)
result.summary()
```

Out[24]:

Logit Regression Results

Dep. Variable:	TenYearCHD	No. Observations:	3751
Model:	Logit	Df Residuals:	3744
Method:	MLE	Df Model:	6
Date:	Mon, 20 Sep 2021	Pseudo R-squ.:	0.1149
Time:	22:31:34	Log-Likelihood:	-1417.7
converged:	True	LL-Null:	-1601.7
Covariance Type:	nonrobust	LLR p-value:	2.127e-76

	coef	std err	z	P>|z|	[0.025	0.975]
const	-9.1264	0.468	-19.504	0.000	-10.043	-8.209
sex_male	0.5815	0.105	5.524	0.000	0.375	0.788
age	0.0655	0.006	10.343	0.000	0.053	0.078
cigsPerDay	0.0197	0.004	4.805	0.000	0.012	0.028
totChol	0.0023	0.001	2.106	0.035	0.000	0.004
sysBP	0.0174	0.002	8.162	0.000	0.013	0.022
glucose	0.0076	0.002	4.574	0.000	0.004	0.011



In [25]:

```
params = np.exp(result.params)
conf = np.exp(result.conf_int())
conf['OR'] = params
pvalue = round(result.pvalues,3)
conf['pvalue'] = pvalue
conf.columns = ['CI 95%(2.5%)','CI 95%(97.5%)', 'Odds Ratio', 'pvalue']
print((conf))
```

```
           CI 95%(2.5%)  CI 95%(97.5%)  Odds Ratio  pvalue
const          0.000043       0.000272    0.000109   0.000
sex_male       1.455242       2.198536    1.788687   0.000
age            1.054483       1.080969    1.067644   0.000
cigsPerDay     1.011733       1.028128    1.019897   0.000
totChol        1.000158       1.004394    1.002273   0.035
sysBP          1.013292       1.021784    1.017529   0.000
glucose        1.004346       1.010898    1.007617   0.000
```

In [27]:

```
import sklearn
new_features =  df[['age','sex_male','cigsPerDay','totChol','sysBP','glucose','TenYearCH
D']]
x = new_features.iloc[:,:-1]
y = new_features.iloc[:,-1]
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=.20, random_state=5)
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
y_pred = logreg.predict(x_test)
```

In [28]:

```
sklearn.metrics.accuracy_score(y_test,y_pred)
```

Out[28]:

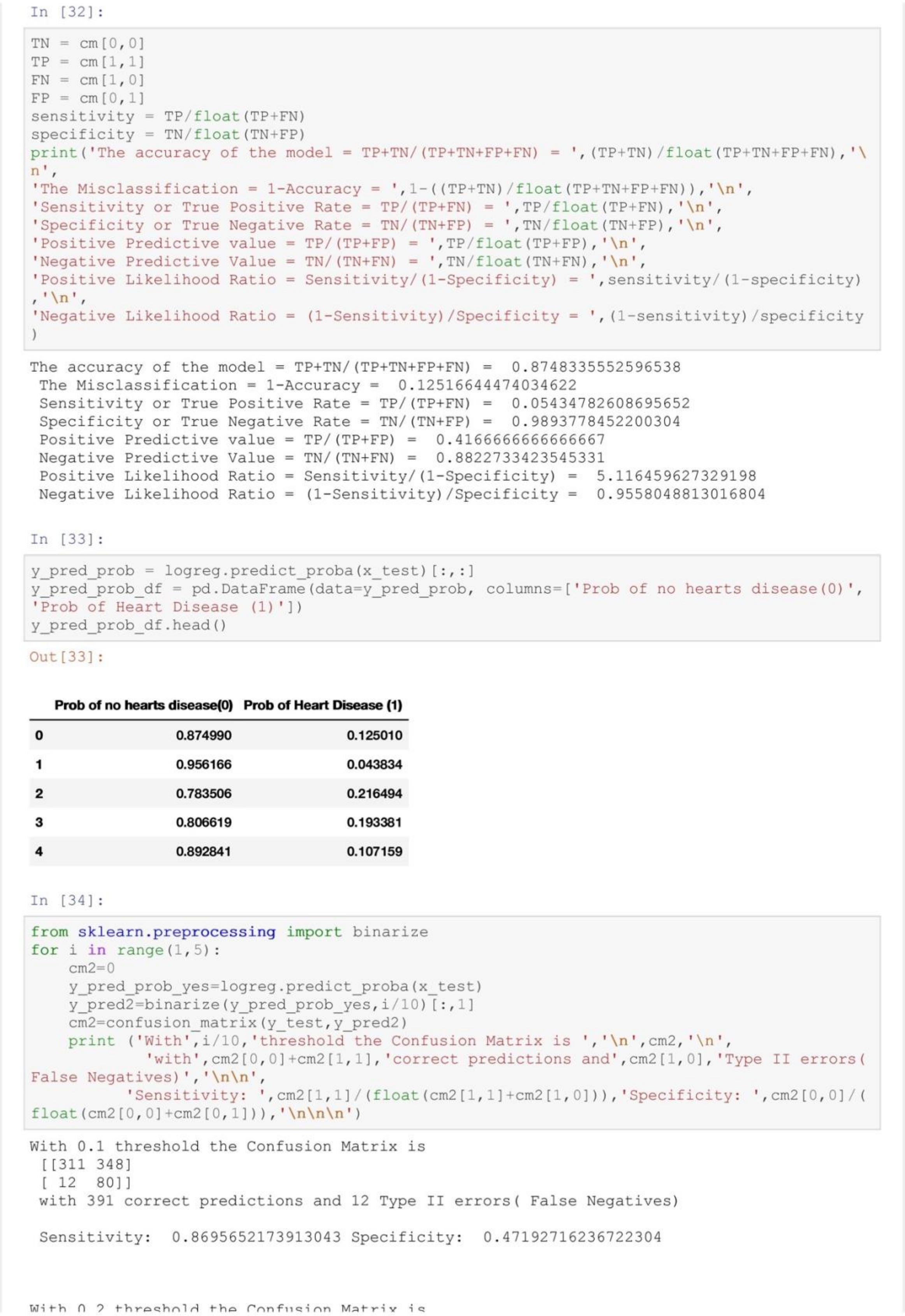
```
0.8748335552596538
```

In [29]:

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
conf_matrix = pd.DataFrame(data=cm, columns=['Predicted:0','Predicted:1'],index=['Actual
:0','Actual:1'])
plt.figure(figsize = (8,5))
sn.heatmap(conf_matrix, annot=True, fmt='d', cmap='YlGnBu')
```

Out[29]:

```
<AxesSubplot:>
```



In [32]:

```
TN = cm[0,0]
TP = cm[1,1]
FN = cm[1,0]
FP = cm[0,1]
sensitivity = TP/float(TP+FN)
specificity = TN/float(TN+FP)
print('The accuracy of the model = TP+TN/(TP+TN+FP+FN) = ',(TP+TN)/float(TP+TN+FP+FN),'\
n',
'The Misclassification = 1-Accuracy = ',1-((TP+TN)/float(TP+TN+FP+FN)),'\n',
'Sensitivity or True Positive Rate = TP/(TP+FN) = ',TP/float(TP+FN),'\n',
'Specificity or True Negative Rate = TN/(TN+FP) = ',TN/float(TN+FP),'\n',
'Positive Predictive value = TP/(TP+FP) = ',TP/float(TP+FP),'\n',
'Negative Predictive Value = TN/(TN+FN) = ',TN/float(TN+FN),'\n',
'Positive Likelihood Ratio = Sensitivity/(1-Specificity) = ',sensitivity/(1-specificity)
,'\n',
'Negative Likelihood Ratio = (1-Sensitivity)/Specificity = ',(1-sensitivity)/specificity
)
```

```
The accuracy of the model = TP+TN/(TP+TN+FP+FN) =  0.8748335552596538
 The Misclassification = 1-Accuracy =  0.12516644474034622
 Sensitivity or True Positive Rate = TP/(TP+FN) =  0.05434782608695652
 Specificity or True Negative Rate = TN/(TN+FP) =  0.9893778452200304
 Positive Predictive value = TP/(TP+FP) =  0.4166666666666667
 Negative Predictive Value = TN/(TN+FN) =  0.8822733423545331
 Positive Likelihood Ratio = Sensitivity/(1-Specificity) =  5.116459627329198
 Negative Likelihood Ratio = (1-Sensitivity)/Specificity =  0.9558048813016804
```

In [33]:

```
y_pred_prob = logreg.predict_proba(x_test)[:,:]
y_pred_prob_df = pd.DataFrame(data=y_pred_prob, columns=['Prob of no hearts disease(0)',
'Prob of Heart Disease (1)'])
y_pred_prob_df.head()
```

Out[33]:

	Prob of no hearts disease(0)	Prob of Heart Disease (1)
0	0.874990	0.125010
1	0.956166	0.043834
2	0.783506	0.216494
3	0.806619	0.193381
4	0.892841	0.107159

In [34]:

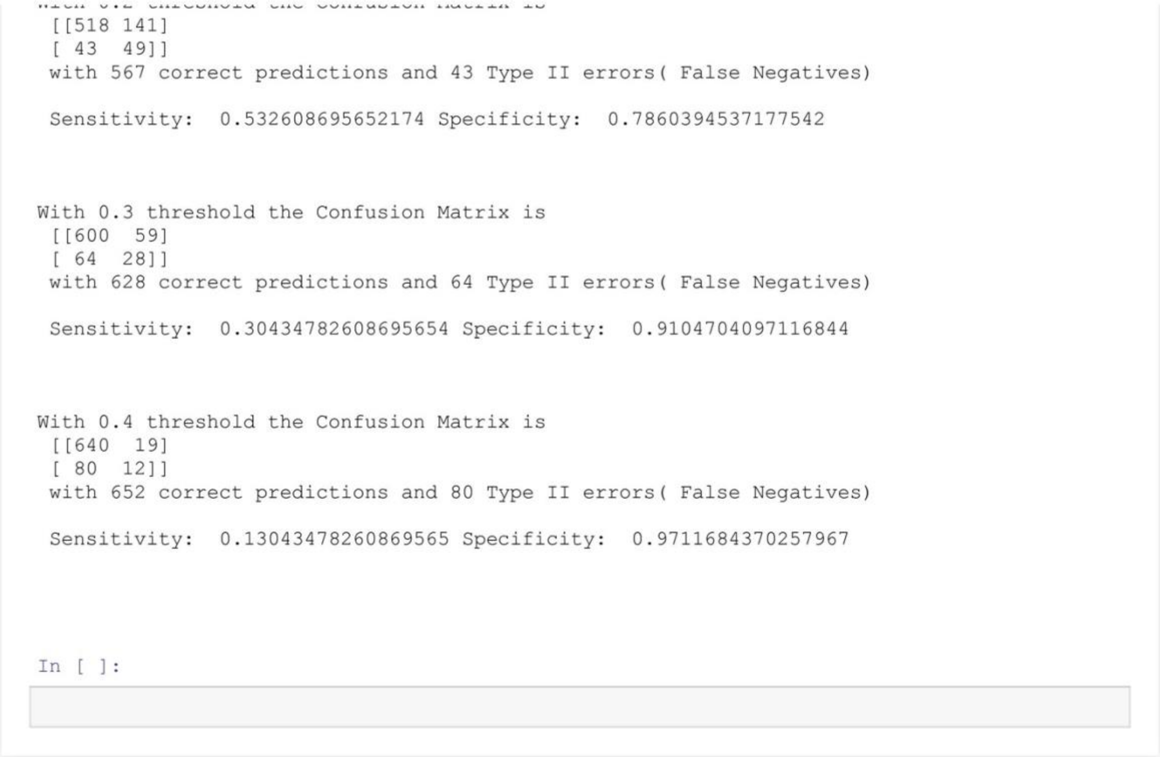
```
from sklearn.preprocessing import binarize
for i in range(1,5):
    cm2=0
    y_pred_prob_yes=logreg.predict_proba(x_test)
    y_pred2=binarize(y_pred_prob_yes,i/10)[:,1]
    cm2=confusion_matrix(y_test,y_pred2)
    print ('With',i/10,'threshold the Confusion Matrix is ','\n',cm2,'\n',
            'with',cm2[0,0]+cm2[1,1],'correct predictions and',cm2[1,0],'Type II errors(
False Negatives)','\n\n',
          'Sensitivity: ',cm2[1,1]/(float(cm2[1,1]+cm2[1,0])),'Specificity: ',cm2[0,0]/(
float(cm2[0,0]+cm2[0,1])),'\n\n\n')
```

```
With 0.1 threshold the Confusion Matrix is
 [[311 348]
 [ 12  80]]
 with 391 correct predictions and 12 Type II errors( False Negatives)

 Sensitivity:  0.8695652173913043 Specificity:  0.47192716236722304



With 0.2 threshold the Confusion Matrix is
```



```
With 0.2 threshold the Confusion Matrix is
 [[518 141]
 [ 43  49]]
 with 567 correct predictions and 43 Type II errors( False Negatives)

 Sensitivity:  0.532608695652174 Specificity:  0.7860394537177542



With 0.3 threshold the Confusion Matrix is
 [[600  59]
 [ 64  28]]
 with 628 correct predictions and 64 Type II errors( False Negatives)

 Sensitivity:  0.30434782608695654 Specificity:  0.9104704097116844



With 0.4 threshold the Confusion Matrix is
 [[640  19]
 [ 80  12]]
 with 652 correct predictions and 80 Type II errors( False Negatives)

 Sensitivity:  0.13043478260869565 Specificity:  0.9711684370257967
```

```
In [ ]:
```