

**MODEL DEVELOPMENT ON WHETHER A PERSON IS DIAGNOSED  
WITH HEART DISEASE OR NOT**

**PROJECT REPORT**

**TERNA ENGINEERING COLLEGE**

**By**

**“ANWESH BISWAS” “ID No: TU2F1920039”-EXTC/SEM5**

**“NABEELA MALLICK” “ID No: TU2F1920040”-EXTC/SEM5**

**“ KAUSTUBH LANGADE” “ID No: TUS3F181924”-CS/SEM5**



**TERNA ENGINEERING COLLEGE**

**Plot no.12, Sector-22, Opp. Nerul Railway station,**

**Phase-11, Nerul (w), Navi Mumbai 400706**

**UNIVERSITY OF MUMBAI**



**TERNA ENGINEERING COLLEGE, NERUL,**

**NAVI MUMBAI**

Academic Year 2020-21

**CERTIFICATE**

This is to certify that

<b>“ANWESH BISWAS”</b>	<b>“ID No: TU2F1920039”-EXTC/SEM5</b>
<b>“NABEELA MALLICK”</b>	<b>“ID No: TU2F1920040”-EXTC/SEM5</b>
<b>“ KAUSTUBH LANGADE”</b>	<b>“ID No: TU2F1920057”-CS/SEM5</b>

Has satisfactorily completed the requirements of the Mini Project of Subject

**“MODEL DEVELOPMENT ON WHETHER A PERSON IS DIAGNOSED  
WITH HEART DISEASE OR NOT”**

As prescribed by the **Knowledge Solutions India**

**Subject In-charge**

**HoD**

**Index**

**TABLE OF CONTENTS**

<b>Sr. No.</b>	<b>Title</b>	<b>PageNo.</b>
Chapter 1	Introduction	4
Chapter 2	Problem Statement	6
2.1	Features	6
Chapter3	Implementation	7
3.1	Flow Chart	7
3.2	Algorithms Used	8
3.3	Packages Used	8
Chapter 4	Results	9
Chapter 5	Prediction	10
Chapter 6	Analysis & Learning	13

## Chapter 1

### Introduction

The objective of this project is to develop a model & determine what variables can are required to conclude that whether a person is diagnosed with heart disease or not.

Most data scientists who do not have strong statistics background may think of logistic regression as a machine learning model. That's true. But it has been around for a long time in statistics.

Statistics is a very important part of data science and machine learning. Because it is essential for any type of exploratory data analysis or machine learning algorithm to learn the features of a dataset, how they relate to each other, how one feature affects the other features, and the overall output.

Luckily python has this amazing library that is the 'stats models' library. This library has great functionalities to understand the dataset and also we can use this library to make predictions. Statsmodels library already has models in-built that can be fitted to the data to find the correlation between the features, learn the coefficients, p-value, test-statistic, standard error, and confidence interval.

This project will explain a statistical modeling technique with an example. I will explain a logistic regression modeling for binary outcome variables here. That means the outcome variable can have only two values, 0 or 1.

- There are lot many reasons/factors which depends on the final decision that concludes that the patient is suffering/has suffered from heart disease

- Factors such as age, sex, chest pain, blood pressure, ECG, heart rate, and other related values are required for the detailed analysis

## Chapter 2

### Problem Statement

It is observed that most of the people in the age bracket of mid-30's to mid-70's suffering from heart disease and this is because of their unhealthy and unhygienic lifestyle moreover people in their age bracket do several kinds of test but don't know what practically range of blood pressure, ECG, heartbeat, cholesterol, gastric problems they should fall in or follow to stay fit and fine.

In this project, we have analyzed and figured out what are the range and order plus how to determine whether the person is diagnosed with heart disease or not

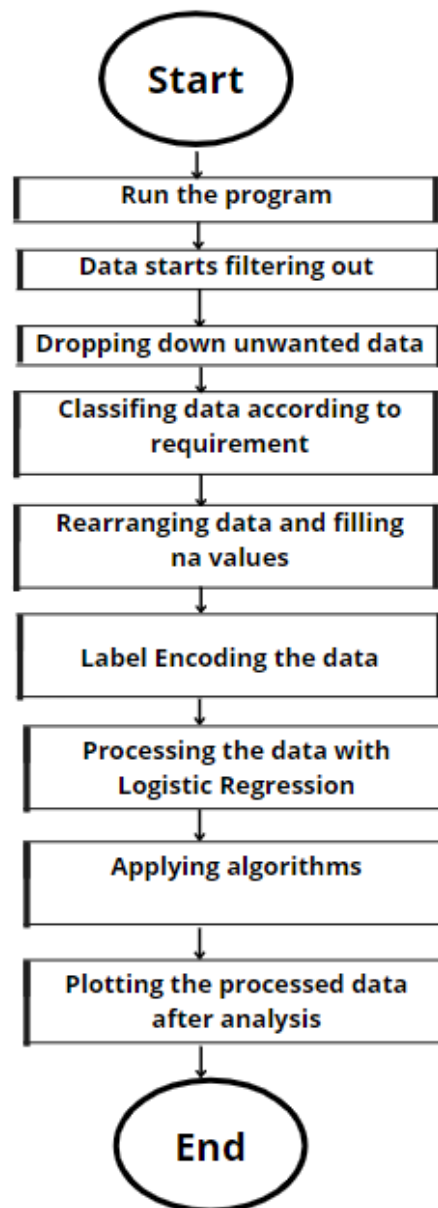
### 2.1 FEATURES

Filtered data/processed data can be visualized with the help of matplotlib & seaborn packages and with the help of sklearn we can train the entire data model after which when **MACHINE LEARNING** algorithms are applied it gives us various accuracy upon comparison it is found that logistic regression holds the highest accuracy

## Chapter 3

### Implementation

#### 3.1FLOWCHART:



### **3.2 Algorithms Used**

- Logistic Regression (LR)

### **3.3 Packages Used**

- Pandas
- NumPy
- Seaborn
- Matplotlib

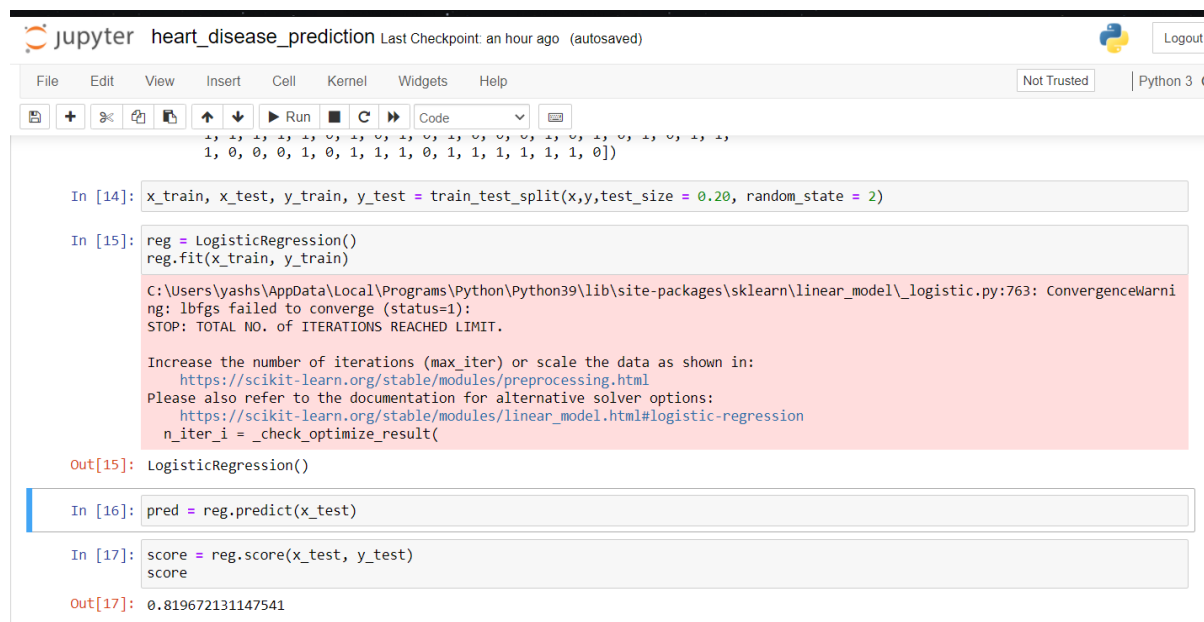


## Chapter 4

### Results

#### Logistic Regression

For this model, we simply imported the concerned module from the sklearn package and passed our data set to the concerned function from the said module. The team then generated a confusion matrix and a classification report using more functions from the sklearn package (metrics module). An accuracy percentage was also calculated using similar methods.



```
In [14]: x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.20, random_state = 2)

In [15]: reg = LogisticRegression()
reg.fit(x_train, y_train)

C:\Users\yashs\AppData\Local\Programs\Python\Python39\lib\site-packages\sklearn\linear_model\_logistic.py:763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
  https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
  https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(

Out[15]: LogisticRegression()

In [16]: pred = reg.predict(x_test)

In [17]: score = reg.score(x_test, y_test)
score

Out[17]: 0.819672131147541
```

**Logistic Regression: 81.96%**

## Chapter 5

### Prediction

Using the results from the model, we can predict if a person has heart disease or not. The models we fitted before were to explain the model parameters. For the prediction purpose, I will use all the variables in the DataFrame. Because we do not have too many variables. Let's check the correlations amongst the variables.

```
df['ChestPain'] = df.ChestPain.replace({"typical":1, "asymptomatic": 2,
'nonanginal': 3, 'nontypical':4})df['Thal'] = df.Thal.replace({'fixed': 1, 'normal': 2,
'reversable': 3})
df[['Age', 'Sex1', 'Chol', 'RestBP', 'Fbs', 'RestECG', 'Slope', 'Oldpeak', 'Ca', 'ExAng',
'ChestPain', 'Thal']].corr()
```

	Age	Chol	RestBP	Fbs	RestECG	Slope	Oldpeak	Ca	ExAng	ChestPain	Thal
Age	1.000000	0.208950	0.284946	0.118530	0.148868	0.161770	0.203805	0.362605	0.091661	-0.173124	0.061823
Chol	0.208950	1.000000	0.130120	0.009841	0.171043	-0.004062	0.046564	0.119000	0.061310	-0.015387	0.080874
RestBP	0.284946	0.130120	1.000000	0.175340	0.146560	0.117382	0.189171	0.098773	0.064762	-0.145149	0.057602
Fbs	0.118530	0.009841	0.175340	1.000000	0.069564	0.059894	0.005747	0.145478	0.025665	-0.023976	-0.007605
RestECG	0.148868	0.171043	0.146560	0.069564	1.000000	0.133946	0.114133	0.128343	0.084867	-0.157005	-0.011543
Slope	0.161770	-0.004062	0.117382	0.059894	0.133946	1.000000	0.577537	0.110119	0.257748	-0.250500	0.106051
Oldpeak	0.203805	0.046564	0.189171	0.005747	0.114133	0.577537	1.000000	0.295832	0.288223	-0.334661	0.208998
Ca	0.362605	0.119000	0.098773	0.145478	0.128343	0.110119	0.295832	1.000000	0.145570	-0.187667	0.149768
ExAng	0.091661	0.061310	0.064762	0.025665	0.084867	0.257748	0.288223	0.145570	1.000000	-0.314993	0.228084
ChestPain	-0.173124	-0.015387	-0.145149	-0.023976	-0.157005	-0.250500	-0.334661	-0.187667	-0.314993	1.000000	-0.168103
Thal	0.061823	0.080874	0.057602	-0.007605	-0.011543	0.106051	0.208998	0.149768	0.228084	-0.168103	1.000000

We can see that each variable has some correlations with other variables. I will use all the variables to get a better prediction.

```
model = sm.GLM.from_formula("AHD ~ Age + Sex1 + Chol + RestBP+ Fbs +
RestECG + Slope + Oldpeak + Ca + ExAng + ChestPain + Thal", family =
sm.families.Binomial(), data=df)
result = model.fit()
result.summary()
```

## Generalized Linear Model Regression Results

Dep. Variable:	AHD	No. Observations:	297
Model:	GLM	Df Residuals:	284
Model Family:	Binomial	Df Model:	12
Link Function:	logit	Scale:	1.0000
Method:	IRLS	Log-Likelihood:	-113.49
Date:	Mon, 03 Aug 2020	Deviance:	226.99
Time:	11:22:53	Pearson chi2:	287.
No. Iterations:	6		
Covariance Type:	nonrobust		

	coef	std err	z	P> z	[0.025	0.975]
Intercept	-9.0184	2.167	-4.162	0.000	-13.266	-4.771
Sex1[T.Male]	1.3765	0.426	3.230	0.001	0.541	2.212
Age	0.0052	0.021	0.245	0.807	-0.037	0.047
Chol	0.0041	0.004	1.160	0.246	-0.003	0.011
RestBP	0.0165	0.010	1.621	0.105	-0.003	0.036
Fbs	-1.0297	0.507	-2.031	0.042	-2.024	-0.036
RestECG	0.2011	0.173	1.164	0.244	-0.138	0.540
Slope	0.7875	0.336	2.343	0.019	0.129	1.446
Oldpeak	0.2789	0.205	1.361	0.174	-0.123	0.681
Ca	1.3120	0.247	5.301	0.000	0.827	1.797
ExAng	1.4776	0.372	3.976	0.000	0.749	2.206
ChestPain	-0.2524	0.205	-1.234	0.217	-0.653	0.149
Thal	0.9276	0.282	3.285	0.001	0.374	1.481

We can use the predict function to predict the outcome. But the predict function uses only the DataFrame. So, let's prepare a DataFrame with the variables and then use the predict function.

```
X = df[['Age', 'Sex1', 'Chol', 'RestBP', 'Fbs', 'RestECG', 'Slope', 'Oldpeak', 'Ca', 'ExAng', 'ChestPain', 'Thal']]
predicted_output = result.predict(X)
```

0	0.285203
1	0.996575
2	0.991014
3	0.537940
4	0.032250
5	0.072342
6	0.920616
7	0.180075
8	0.861155
9	0.893586
10	0.105111
11	0.115857
12	0.478468
13	0.141911
14	0.142187
15	0.135667
16	0.443162
17	0.167579
18	0.027637
19	0.086566
20	0.716429

The predicted output should be either 0 or 1. It's 1 when the output is greater than or equal to 0.5 and 0 otherwise.

```
for i in range(0, len(predicted_output)):
    predicted_output = predicted_output.replace()
    if predicted_output[i] >= 0.5:
        predicted_output = predicted_output.replace(predicted_output[i], 1)
    else:
        predicted_output = predicted_output.replace(predicted_output[i], 0)
```

Now, compare this predicted\_output to the 'AHD' column of the DataFrame which indicates the heart disease to find the accuracy:

```
accuracy = 0
for i in range(0, len(predicted_output)):
    if df['AHD'][i] == predicted_output[i]:
        accuracy += 1
accuracy/len(df)
```

The accuracy comes out to be 0.81 or 81% which is very good.

## Chapter 6

### Analysis and Learning

1. A basic logistic regression model fitting with one variable
2. Understanding odds and log-odds with examples
3. Logistic regression model fitting with two variables
4. Logistic regression model fitting with three variables
5. Visualization of the fitted model
6. Prediction

#### 6.1 Odds And Log Odds

A logistic regression model provides the ‘odds’ of an event. Remember that, ‘odds’ are the probability on a different scale. Here is the formula:

If an event has a probability of  $p$ ,

the odds of that event are  $p/(1-p)$

Odds are the transformation of probability.

**Based on this formula, if the probability is  $1/2$ , the ‘odds’ is 1.**

To understand the odds and log-odds clearly, let’s work on an example. We will use the gender variable.

**Because a categorical variable is appropriate for this.**

Check the proportion of males and females having heart disease in the dataset.

```
df["Sex1"] = df.Sex.replace({1: "Male", 0:"Female"})
c = pd.crosstab(df.Sex1, df.AHD)
c = c.apply(lambda x: x/x.sum(), axis=1)
```

AHD	0	1
Sex1		
Female	0.742268	0.257732
Male	0.446602	0.553398

Let's calculate the 'odds' of heart disease for males and females.

```
c["odds"] = c.loc[:, 1] / c.loc[:, 0]
```

smq	0.0	1.0	odds
RIAGENDRx			
Female	0.680197	0.319803	0.470162
Male	0.467453	0.532547	1.139252

The 'odds' show that the probability of a female having heart disease is substantially lower than a male(32% vs 53%) which reflects very well in the odds. Odds ratios are common to use while working with two population groups.

$c.odds.Male / c.odds.Female$

**The ratio comes out to be 3.587 which indicates a man has a 3.587 times greater chance of having heart disease.**

Remember that, an individual probability cannot be calculated from an odds ratio. Another important convention is to work with log-odds which are odds on a logarithmic scale.

Recall that the neutral point of the probability is 0.5. Using the formula for 'odds', odds for 0.5 is 1, and 'log-odds is 0 (log of 1 is 0).

*In our exercise where men have a greater chance of having heart disease, have 'odds' between 1 and infinity. At the same time, the 'odds' of women having a greater chance of having heart disease is 0 to 1.*

Here is the log odds calculation:

```
c['logodds'] = np.log(c.odds)
```

AHD	0	1	odds	logodds
Sex1				
Female	0.742268	0.257732	0.347222	-1.05779
Male	0.446602	0.553398	1.239130	0.21441

Here, the log-odds of the female population are negative which indicates that less than 50% of females have heart disease.

Log-odds of males is positive and a little more than 0 which means more than half of the males have heart disease.