Bansilal Ramnath Agarwal Charitable Trust's

Vishwakarma Institute of Information Technology

PROJECT REPORT

Data Science

GROUP MEMBERS:

Name	Roll No.	GR No.	Batch
Shreyas Asutkar	324005	21810004	Comp D1
Shubham Bhopale	324013	21810260	Comp D1
Nikit Gokhe	324022	21810522	Comp D1
Nikhil Tayade	324053	21810734	Comp D3

Index

Aim:

To develop Mini Project on data Analysis: Identify problem statement. Use Semi or unstructured data set. Define 3 to 4 objectives. Perform 1. Data Interpretation, 2. Data pre-processing, 3. Data Modelling (perform both Descriptive and Predictive analysis, also perform Prescriptive Analysis (if required and fits for the data set)), and 4.data visualization.

Problem Statement:

We have a data which classified if patients have heart disease or not according to features in it. We will try to use this data to create a model which tries predict if a patient has this disease or not.

Objectives:

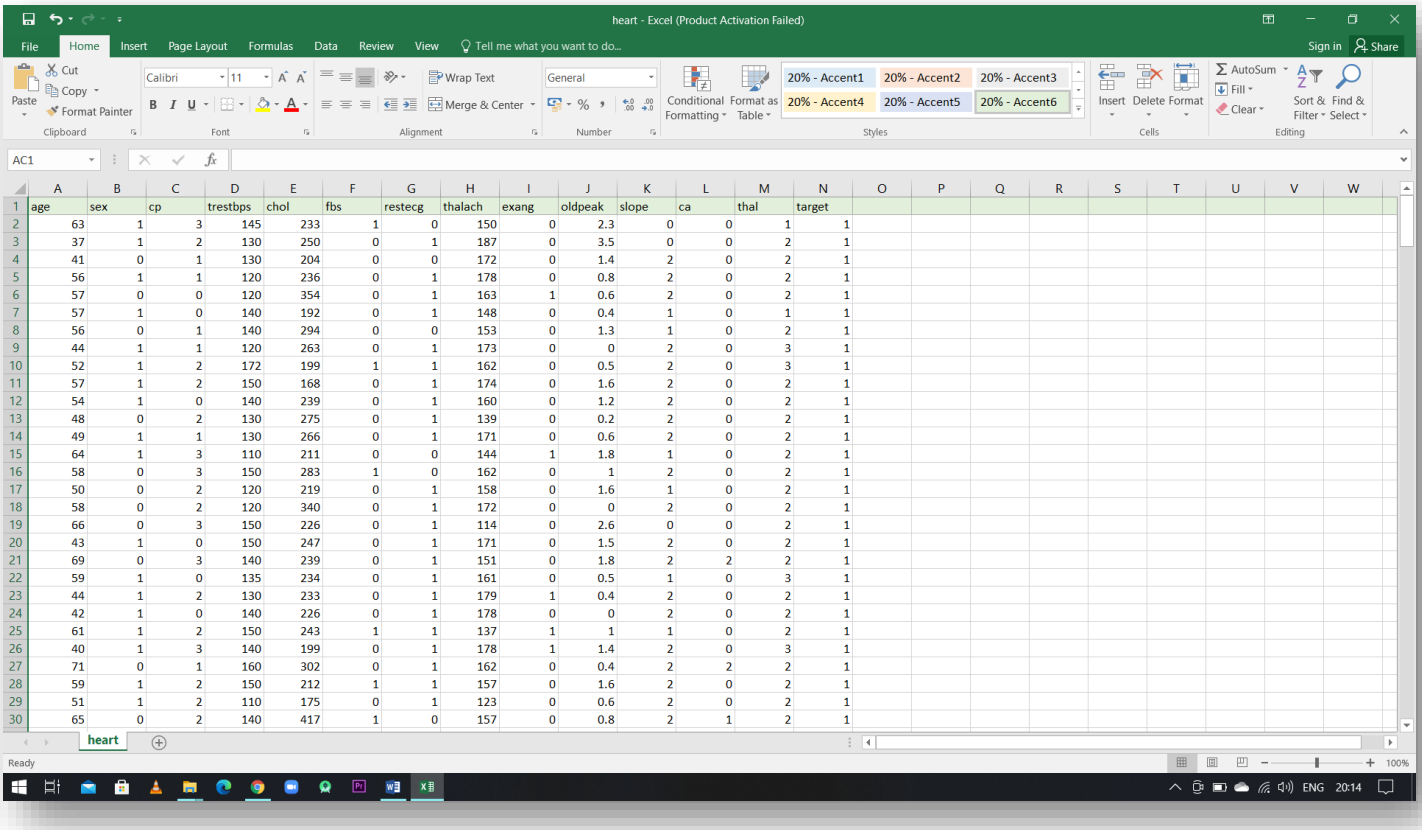
- Understand Dataset: Read data from dataset, describe attributed of data, checking data types of each column, counting unique values of data etc.
- Perform Data cleaning, Data integration, Data transformation and make dataset perfectly ready to use for analysis
  (if required)
- Data Visualization- create different visuals for each feature in the dataset to illustrate the feature distributions and for clear understanding of dataset.
- Using logistic regression (classification) algorithm, we will create a model which tries predict if a patient has this disease or not.

Dataset:

Data contains:

- Age - age in years
- Sex - (1 = male; 0 = female)
- cp - chest pain type
- trestbps - resting blood pressure (in mm Hg on admission to the hospital)
- chol - serum cholesterol in mg/dl
- fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg - resting electrocardiographic results
- thalach - maximum heart rate achieved
- exang - exercise induced angina (1 = yes; 0 = no)
- oldpeak - ST depression induced by exercise relative to rest
- slope - the slope of the peak exercise ST segment
- ca - number of major vessels (0-3) coloured by fluoroscopy
- thal - 3 = normal; 6 = fixed defect; 7 = reversible defect
- target - have disease or not (1=yes, 0=no)

For Dataset click here



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
2	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
3	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
4	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
5	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
6	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
7	57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
8	56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
9	44	1	1	120	263	0	1	173	0	0	2	0	3	1
10	52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
11	57	1	2	150	168	0	1	174	0	1.6	2	0	2	1
12	54	1	0	140	239	0	1	160	0	1.2	2	0	2	1
13	48	0	2	130	275	0	1	139	0	0.2	2	0	2	1
14	49	1	1	130	266	0	1	171	0	0.6	2	0	2	1
15	64	1	3	110	211	0	0	144	1	1.8	1	0	2	1
16	58	0	3	150	283	1	0	162	0	1	2	0	2	1
17	50	0	2	120	219	0	1	158	0	1.6	1	0	2	1
18	58	0	2	120	340	0	1	172	0	0	2	0	2	1
19	66	0	3	150	226	0	1	114	0	2.6	0	0	2	1
20	43	1	0	150	247	0	1	171	0	1.5	2	0	2	1
21	69	0	3	140	239	0	1	151	0	1.8	2	2	2	1
22	59	1	0	135	234	0	1	161	0	0.5	1	0	3	1
23	44	1	2	130	233	0	1	179	1	0.4	2	0	2	1
24	42	1	0	140	226	0	1	178	0	0	2	0	2	1
25	61	1	2	150	243	1	1	137	1	1	1	0	2	1
26	40	1	3	140	199	0	1	178	1	1.4	2	0	3	1
27	71	0	1	160	302	0	1	162	0	0.4	2	2	2	1
28	59	1	2	150	212	1	1	157	0	1.6	2	0	2	1
29	51	1	2	110	175	0	1	123	0	0.6	2	0	2	1
30	65	0	2	140	417	1	0	157	0	0.8	2	1	2	1

Theory:

In statistics, the logistic model (or logit model) is used to model the probability of a certain class or event existing such as pass/fail, win/lose, alive/dead or healthy/sick. This can be extended to model several classes of events such as determining whether an image contains a cat, dog, lion, etc. Each object being detected in the image would be assigned a probability between 0 and 1, with a sum of one.

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression[1] (or logit regression) is estimating the parameters of a logistic model (a form of binary regression). Mathematically, a binary logistic model has a dependent variable with two possible values, such as pass/fail which is represented by an indicator variable, where the two values are labeled "0" and "1".

In the logistic model, the log-odds (the logarithm of the odds) for the value labeled "1" is a linear combination of one or more independent variables ("predictors"); the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labeled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labeling; the function that converts log-odds to probability is the logistic function, hence the name.

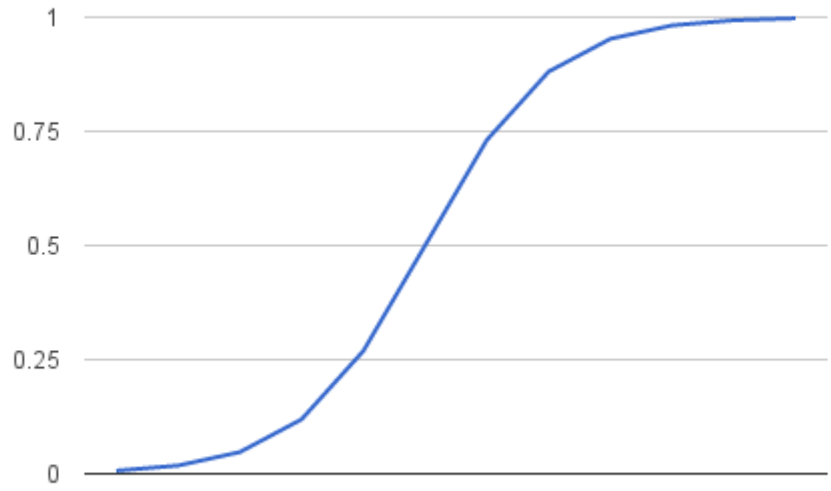
Algorithm (Logistic Regression):

Logistic Function

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It's an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value)



Representation Used for Logistic Regression

Logistic regression uses an equation as the representation, very much like linear regression.

Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

Below is an example logistic regression equation:

$$y = e^{(b0 + b1*x)} / (1 + e^{(b0 + b1*x)})$$

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data.

Logistic Regression Predicts Probabilities (Technical Interlude)

Logistic regression models the probability of the default class (e.g. the first class).

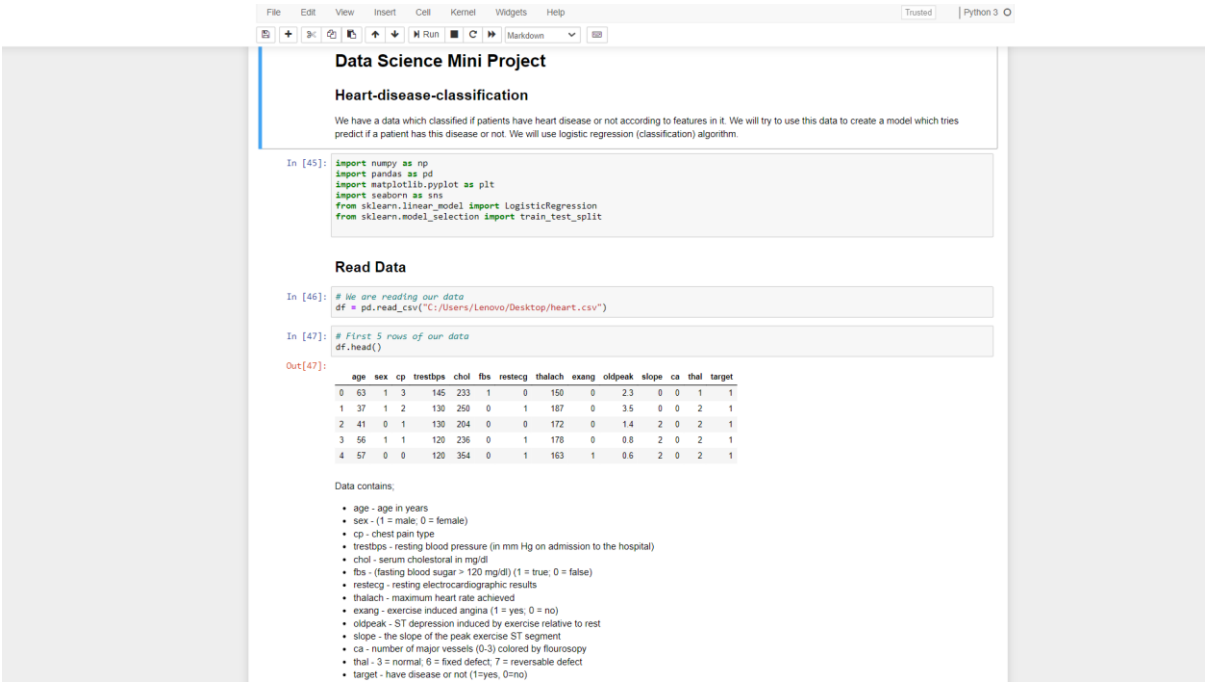
For example, if we are modeling people's sex as male or female from their height, then the first class could be male and the logistic regression model could be written as the probability of male given a person's height, or more formally:

P(sex=male|height)

Learning the Logistic Regression Model

The coefficients (Beta values b) of the logistic regression algorithm must be estimated from your training data. The best coefficients would result in a model that would predict a value very close to 1 (e.g. male) for the default class and a value very close to 0 (e.g. female) for the other class. The intuition for maximum-likelihood for logistic regression is that a search procedure seeks values for the coefficients (Beta values) that minimize the error in the probabilities predicted by the model to those in the data (e.g. probability of 1 if the data is the primary class).

Code and Output:



File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Data Science Mini Project

Heart-disease-classification

We have a data which classified if patients have heart disease or not according to features in it. We will try to use this data to create a model which tries predict if a patient has this disease or not. We will use logistic regression (classification) algorithm.

In [45]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
```

Read Data

In [46]:
```
# We are reading our data
df = pd.read_csv("C:/Users/Lenovo/Desktop/heart.csv")
```

In [47]:
```
# First 5 rows of our data
df.head()
```

Out[47]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

Data contains;

- age - age in years
- sex - (1 = male; 0 = female)
- cp - chest pain type
- trestbps - resting blood pressure (in mm Hg on admission to the hospital)
- chol - serum cholestoral in mg/dl
- fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
- restecg - resting electrocardiographic results
- thalach - maximum heart rate achieved
- exang - exercise induced angina (1 = yes; 0 = no)
- oldpeak - ST depression induced by exercise relative to rest
- slope - the slope of the peak exercise ST segment
- ca - number of major vessels (0-3) colored by flourosopy
- thal - 3 = normal; 6 = fixed defect; 7 = reversable defect
- target - have disease or not (1=yes, 0=no)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

In [59]:
```
#Given Dataset does not contain any null values
```

In [65]:
```
df.std()
```

Out[65]:
```
age          9.082101
sex          0.466011
cp           1.032052
trestbps    17.538143
chol        51.830751
fbs          0.356198
restecg      0.525860
thalach     22.905161
exang        0.469794
oldpeak      1.161075
slope        0.616226
ca           1.022606
thal         0.612277
target       0.498835
dtype: float64
```

In [66]:
```
df.var()
```

Out[66]:
```
age          82.484558
sex           0.217166
cp            1.065132
trestbps    307.586453
chol       2686.426748
fbs           0.126877
restecg       0.276528
thalach     524.646406
exang         0.220707
oldpeak       1.348095
slope         0.379735
ca            1.045724
thal          0.374883
target        0.248836
dtype: float64
```

In [63]:
```
df.target.value_counts()
```

Out[63]:
```
1    165
0    138
Name: target, dtype: int64
```

Descriptive Analysis and Visualizations

In [6]:
```
sns.countplot(x="target", data=df, palette="bwr")
plt.show()
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Data Exploration

In [54]:
```
df.describe()
```

Out[54]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	
count	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.000000	303.0
mean	54.366337	0.683168	0.966997	131.623762	246.264026	0.148515	0.528053	149.646865	0.326733	1.039604	1.399340	0.729373	2.3
std	9.082101	0.466011	1.032052	17.538143	51.830751	0.356198	0.525860	22.905161	0.469794	1.161075	0.616226	1.022606	0.6
min	29.000000	0.000000	0.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	47.500000	0.000000	0.000000	120.000000	211.000000	0.000000	0.000000	133.500000	0.000000	0.000000	1.000000	0.000000	2.0
50%	55.000000	1.000000	1.000000	130.000000	240.000000	0.000000	1.000000	153.000000	0.000000	0.800000	1.000000	0.000000	2.0
75%	61.000000	1.000000	2.000000	140.000000	274.500000	0.000000	1.000000	166.000000	1.000000	1.600000	2.000000	1.000000	3.0
max	77.000000	1.000000	3.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.0

In [55]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
age         303 non-null int64
sex         303 non-null int64
cp          303 non-null int64
trestbps    303 non-null int64
chol        303 non-null int64
fbs         303 non-null int64
restecg     303 non-null int64
thalach     303 non-null int64
exang       303 non-null int64
oldpeak     303 non-null float64
slope       303 non-null int64
ca          303 non-null int64
thal        303 non-null int64
target      303 non-null int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

In [56]:
```
df.isnull().sum()
```

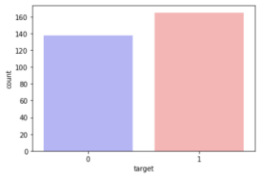
Out[56]:
```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Descriptive Analysis and Visualizations

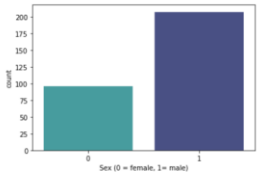
In [6]:
```
sns.countplot(x="target", data=df, palette="bwr")
plt.show()
```



In [7]:
```
countNoDisease = len(df[df.target == 0])
countHaveDisease = len(df[df.target == 1])
print("Percentage of Patients Haven't Heart Disease: {:.2f}%".format((countNoDisease / (len(df.target))*100)))
print("Percentage of Patients Have Heart Disease: {:.2f}%".format((countHaveDisease / (len(df.target))*100)))
```

```
Percentage of Patients Haven't Heart Disease: 45.54%
Percentage of Patients Have Heart Disease: 54.46%
```

In [8]:
```
sns.countplot(x='sex', data=df, palette="mako_r")
plt.xlabel("Sex (0 = female, 1= male)")
plt.show()
```

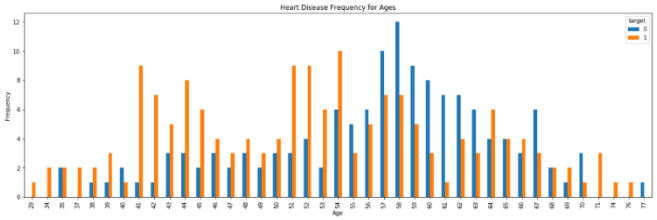


In [9]:
```
countFemale = len(df[df.sex == 0])
countMale = len(df[df.sex == 1])
print("Percentage of Female Patients: {:.2f}%".format((countFemale / (len(df.sex))*100)))
print("Percentage of Male Patients: {:.2f}%".format((countMale / (len(df.sex))*100)))
```

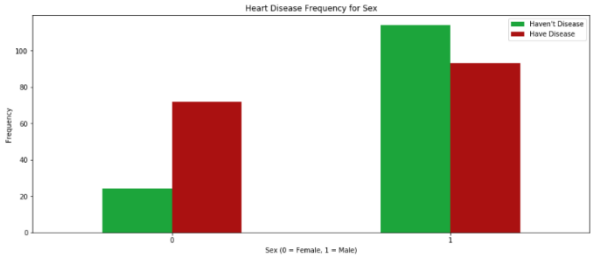
File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

```
plt.savefig('heartDiseaseAndAges.png')
plt.show()
```

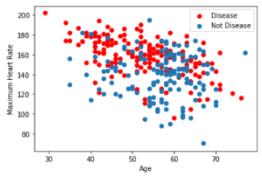


In [12]:
```
pd.crosstab(df.sex,df.target).plot(kind="bar",figsize=(15,6),color=['#1CA53B','#AA1111' ])
plt.title('Heart Disease Frequency for Sex')
plt.xlabel('Sex (0 = Female, 1 = Male)')
plt.xticks(rotation=0)
plt.legend(["Haven't Disease", "Have Disease"])
plt.ylabel('Frequency')
plt.show()
```

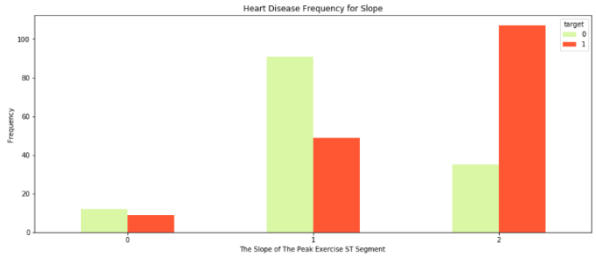


In [13]:
```
plt.scatter(x=df.age[df.target==1], y=df.thalach[(df.target==1)], c="red")
```

```
plt.xlabel("Age")
plt.ylabel("Maximum Heart Rate")
plt.show()
```

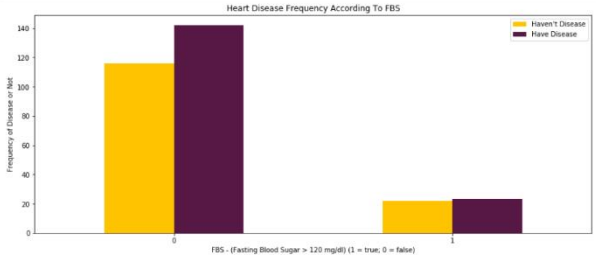


In [14]:
```
pd.crosstab(df.slope,df.target).plot(kind="bar",figsize=(15,6),color=['#DAF7A6','#FF5733' ])
plt.title('Heart Disease Frequency for Slope')
plt.xlabel('The Slope of The Peak Exercise ST Segment ')
plt.xticks(rotation = 0)
plt.ylabel('Frequency')
plt.show()
```

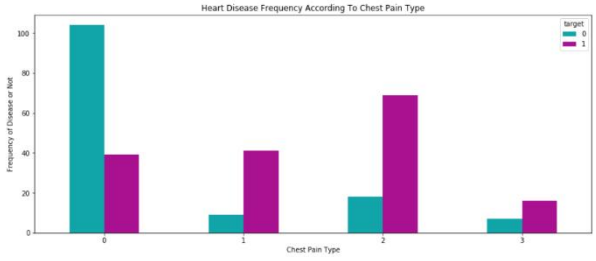


In [15]:
```
pd.crosstab(df.fbs,df.target).plot(kind="bar",figsize=(15,6),color=['#FFC300','#581845' ])
plt.title('Heart Disease Frequency According To FBS')
plt.xlabel('FBS - (Fasting Blood Sugar > 120 mg/dl) (1 = true; 0 = false)')
plt.xticks(rotation = 0)
plt.legend(["Haven't Disease", "Have Disease"])
```

```
plt.show()
```



In [16]:
```
pd.crosstab(df.cp,df.target).plot(kind="bar",figsize=(15,6),color=['#11A5AA','#AA1190' ])
plt.title('Heart Disease Frequency According To Chest Pain Type')
plt.xlabel('Chest Pain Type')
plt.xticks(rotation = 0)
plt.ylabel('Frequency of Disease or Not')
plt.show()
```



File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Creating Model for Logistic Regression

We can use sklearn library or we can write functions ourselves. Let's them both. Firstly we will write our functions after that we'll use sklearn library to calculate score.

In [20]:
```
y = df.target.values
x_data = df.drop(['target'], axis = 1)
```

In [21]:
```
# Normalize
x = (x_data - np.min(x_data)) / (np.max(x_data) - np.min(x_data)).values
```

We will split our data. 80% of our data will be train data and 20% of it will be test data.

In [22]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size = 0.2,random_state=0)
```

In [23]:
```
#transpose matrices
x_train = x_train.T
y_train = y_train.T
x_test = x_test.T
y_test = y_test.T
```

Let's say weight = 0.01 and bias = 0.0

Sklearn Logistic Regression

In [48]:
```
accuracies = {}

lr = LogisticRegression()
lr.fit(x_train.T,y_train.T)
acc = lr.score(x_test.T,y_test.T)*100

accuracies['Logistic Regression'] = acc
print("Test Accuracy {:.2f}%".format(acc))
```

```
Test Accuracy 86.89%
```

```
C:\Users\Lenovo\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be changed
to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

Logistic Model works with 86.89accuracy.

Naive Bayes Algorithm

In [49]:
```
from sklearn.naive_bayes import GaussianNB
nb = GaussianNB()
```

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Markdown

Creating Dummy Variables

Since 'cp', 'thal' and 'slope' are categorical variables we'll turn them into dummy variables.

In [17]:
```
a = pd.get_dummies(df['cp'], prefix = "cp")
b = pd.get_dummies(df['thal'], prefix = "thal")
c = pd.get_dummies(df['slope'], prefix = "slope")
```

In [18]:
```
frames = [df, a, b, c]
df = pd.concat(frames, axis = 1)
df.head()
```

Out[18]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	...	cp_1	cp_2	cp_3	thal_0	thal_1	thal_2	thal_3	slope_0	slope_1	slope_2
0	63	1	3	145	233	1	0	150	0	2.3	...	0	0	1	0	1	0	0	1	0	0
1	37	1	2	130	250	0	1	187	0	3.5	...	0	1	0	0	0	1	0	1	0	0
2	41	0	1	130	204	0	0	172	0	1.4	...	1	0	0	0	0	1	0	0	0	1
3	56	1	1	120	236	0	1	178	0	0.8	...	1	0	0	0	0	1	0	0	0	1
4	57	0	0	120	354	0	1	163	1	0.6	...	0	0	0	0	0	1	0	0	0	1

5 rows × 25 columns

In [19]:
```
df = df.drop(columns = ['cp', 'thal', 'slope'])
df.head()
```

Out[19]:

	age	sex	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	ca	...	cp_1	cp_2	cp_3	thal_0	thal_1	thal_2	thal_3	slope_0	slope_1	slope_2
0	63	1	145	233	1	0	150	0	2.3	0	...	0	0	1	0	1	0	0	1	0	0
1	37	1	130	250	0	1	187	0	3.5	0	...	0	1	0	0	0	1	0	1	0	0
2	41	0	130	204	0	0	172	0	1.4	0	...	1	0	0	0	0	1	0	0	0	1
3	56	1	120	236	0	1	178	0	0.8	0	...	1	0	0	0	0	1	0	0	0	1
4	57	0	120	354	0	1	163	1	0.6	0	...	0	0	0	0	0	1	0	0	0	1

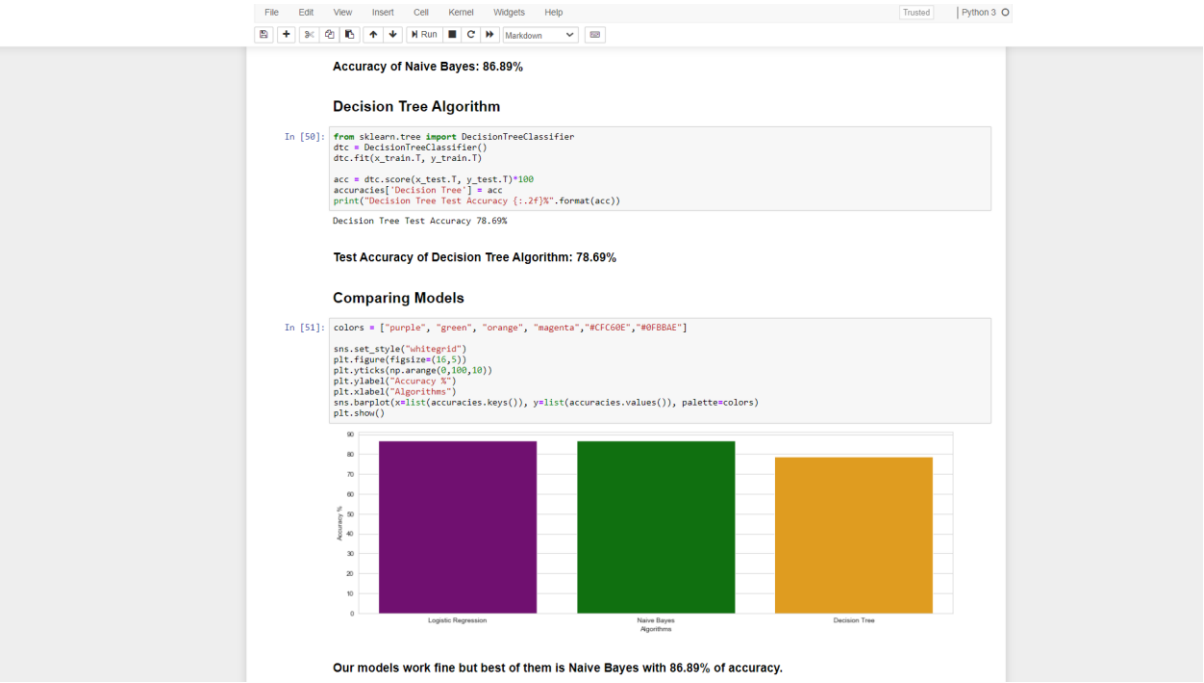
5 rows × 22 columns

Creating Model for Logistic Regression

We can use sklearn library or we can write functions ourselves. Let's them both. Firstly we will write our functions after that we'll use sklearn library to calculate score.

In [20]:
```
y = df.target.values
x_data = df.drop(['target'], axis = 1)
```

In [21]:
```
# Normalize
x = (x_data - np.min(x_data)) / (np.max(x_data) - np.min(x_data)).values
```



Inferences & Conclusion:

- Data needed dummy variable creation as it consisted of categorical variables.
- Target Variable was binary as it consisted of two classes that is have heart disease and don't have heart disease.
- Logistic Regression was applied by splitting the dataset into training and testing and 86.89% accuracy was achieved.
- Inorder to verify the accuracy two more algorithms were applied which are Decisoin Tree and Naïve Bayes.
- Naïve Bayes model had same accuracy as Logistic Regression model.
- Given the attributes it can predicted whether the patient will have heart disease.