**MACHINE LEARNING**

**DATASET:**

**HEART DISEASE**

**NAME: R.LOGESH**

**ROLL NO: 195214127**

**CLASS:I M.Sc-CS-A**

**SHIFT: I**

**­­­ MACHINE LEARNING PROJECT**

In this project, I have used Linear Regression Machine Learning model for **Heart disease** dataset from Kaggle website.

**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. We will use logistic regression (classification) algorithm.

**ML METHODOLOGY:**

Linear Regression is the methodology used for training and testing the dataset.Linear Regression is a method of modeling a target value based on independent predictors. This method is mostly used for forecasting and finding out cause and effect relationship between variables. Linear Regression techniques mostly differ based on the number of independent variables and the type of relationship between the independent and dependent variables.

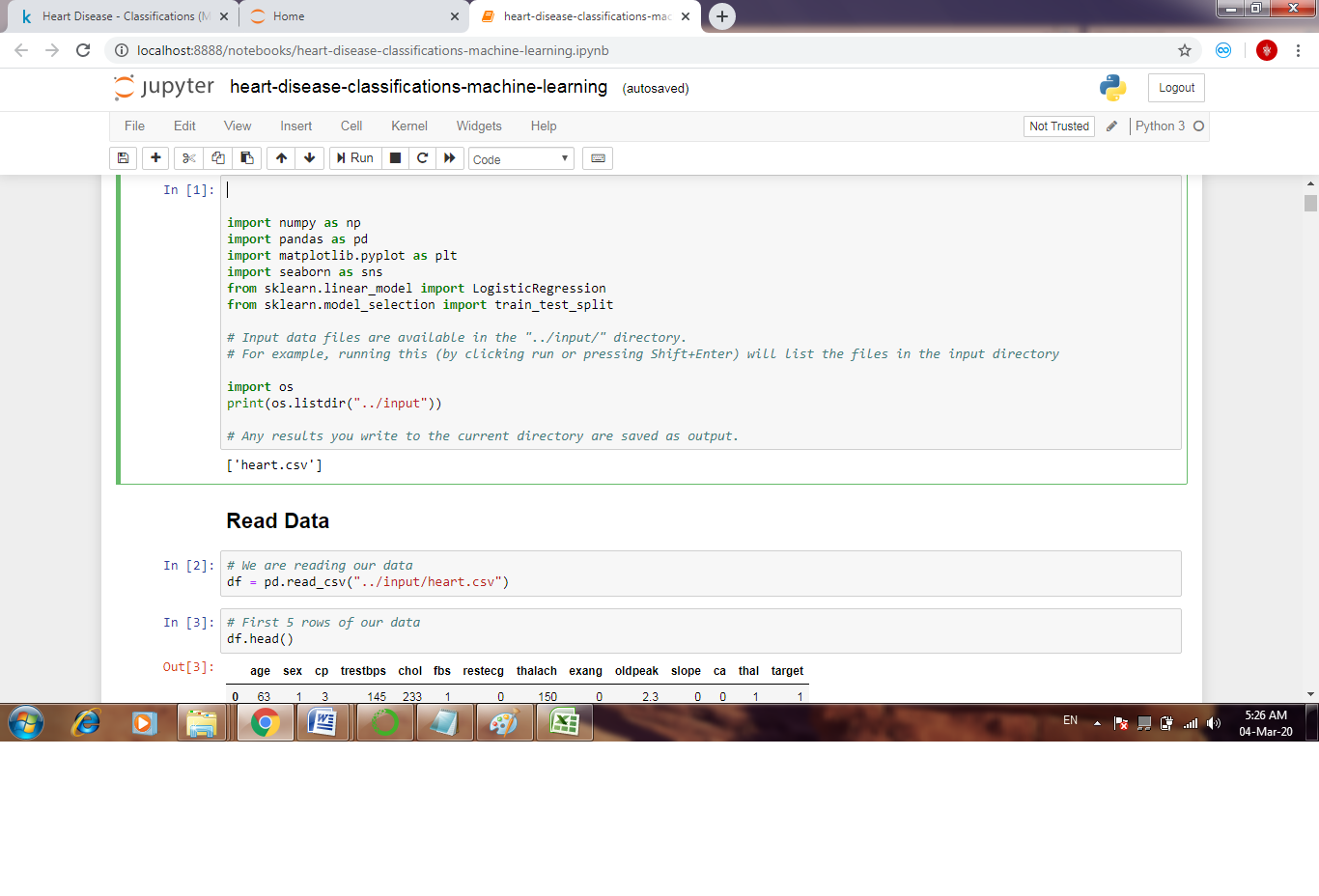
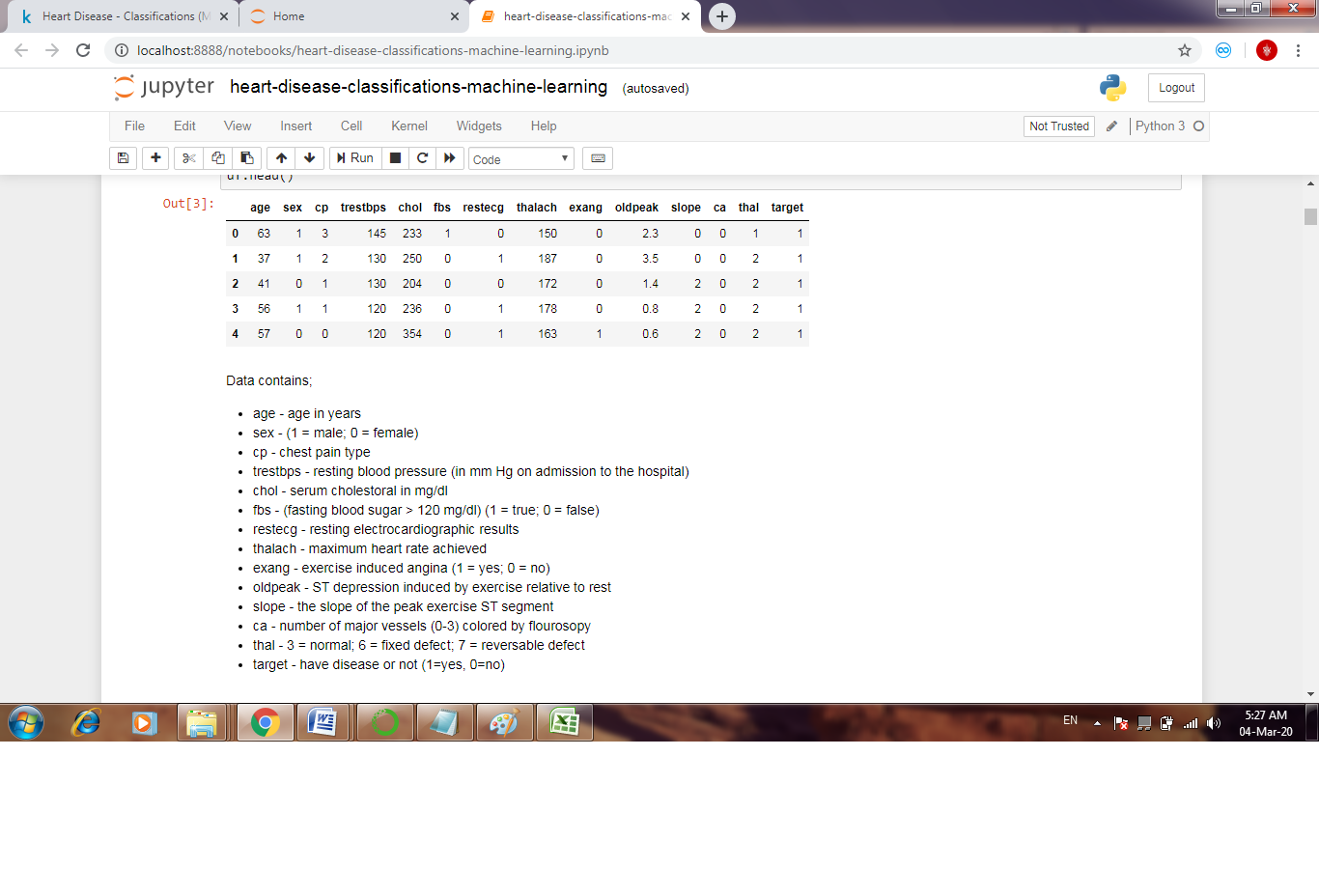
**DATASET DESCRIPTION:**

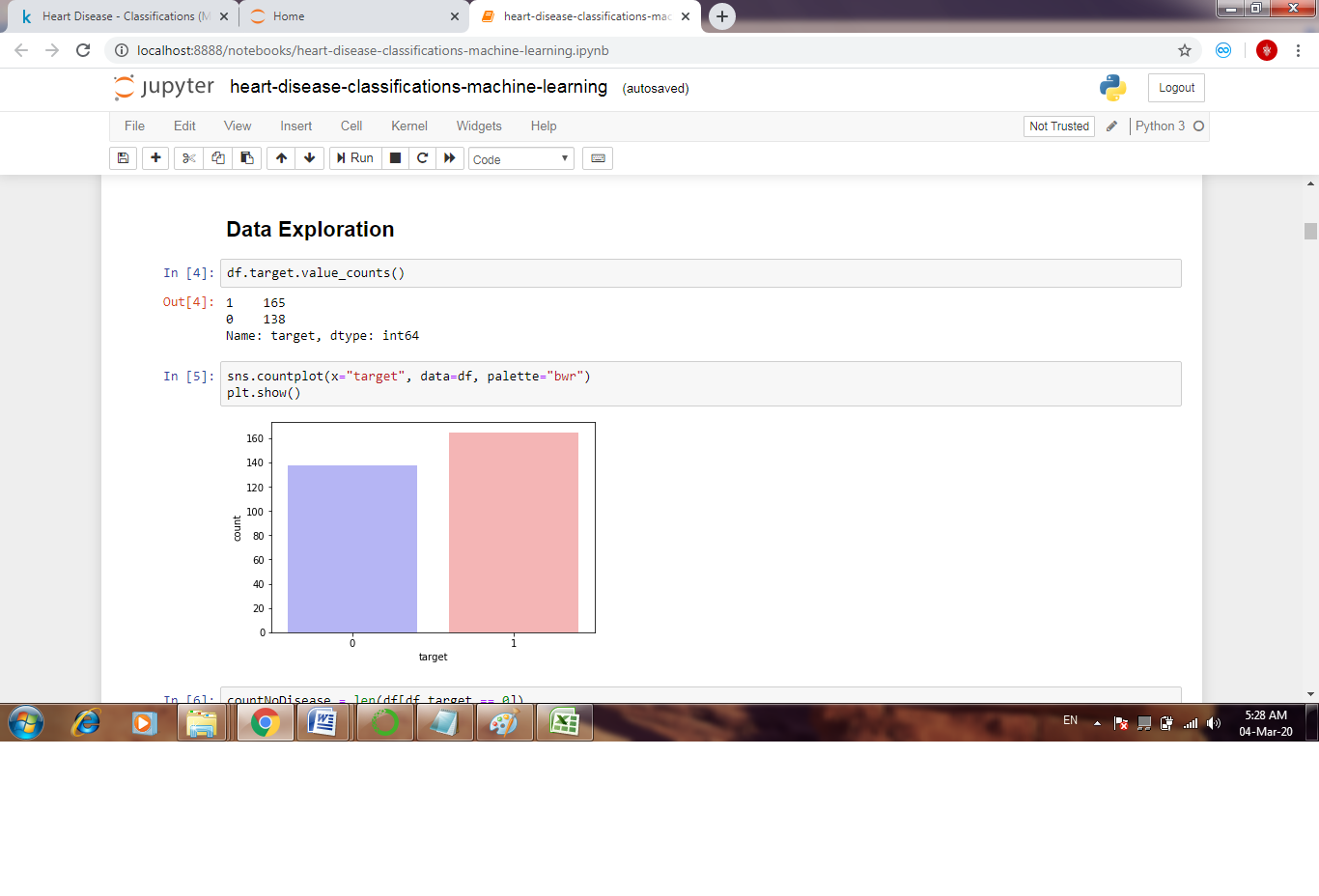
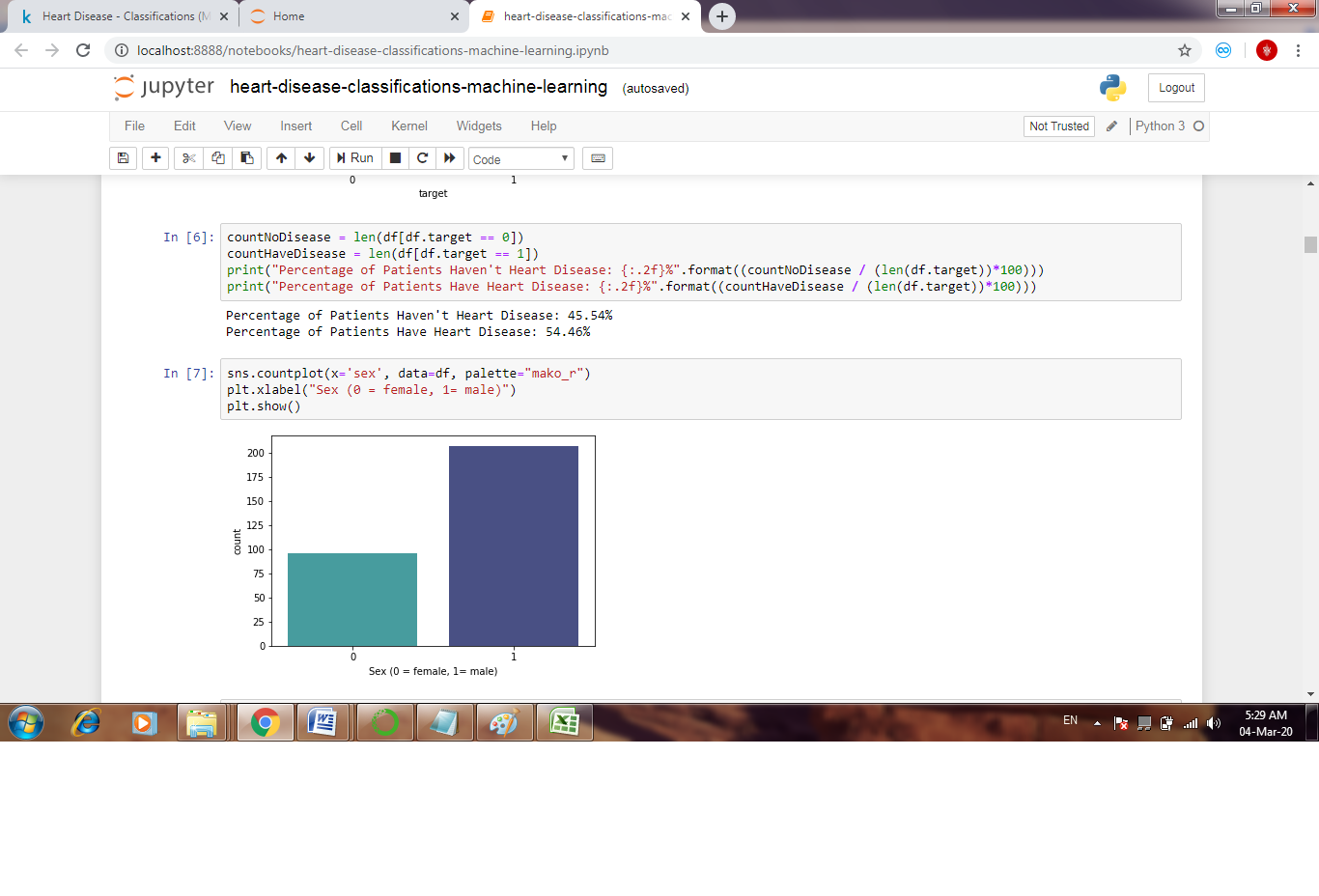
Data contains:

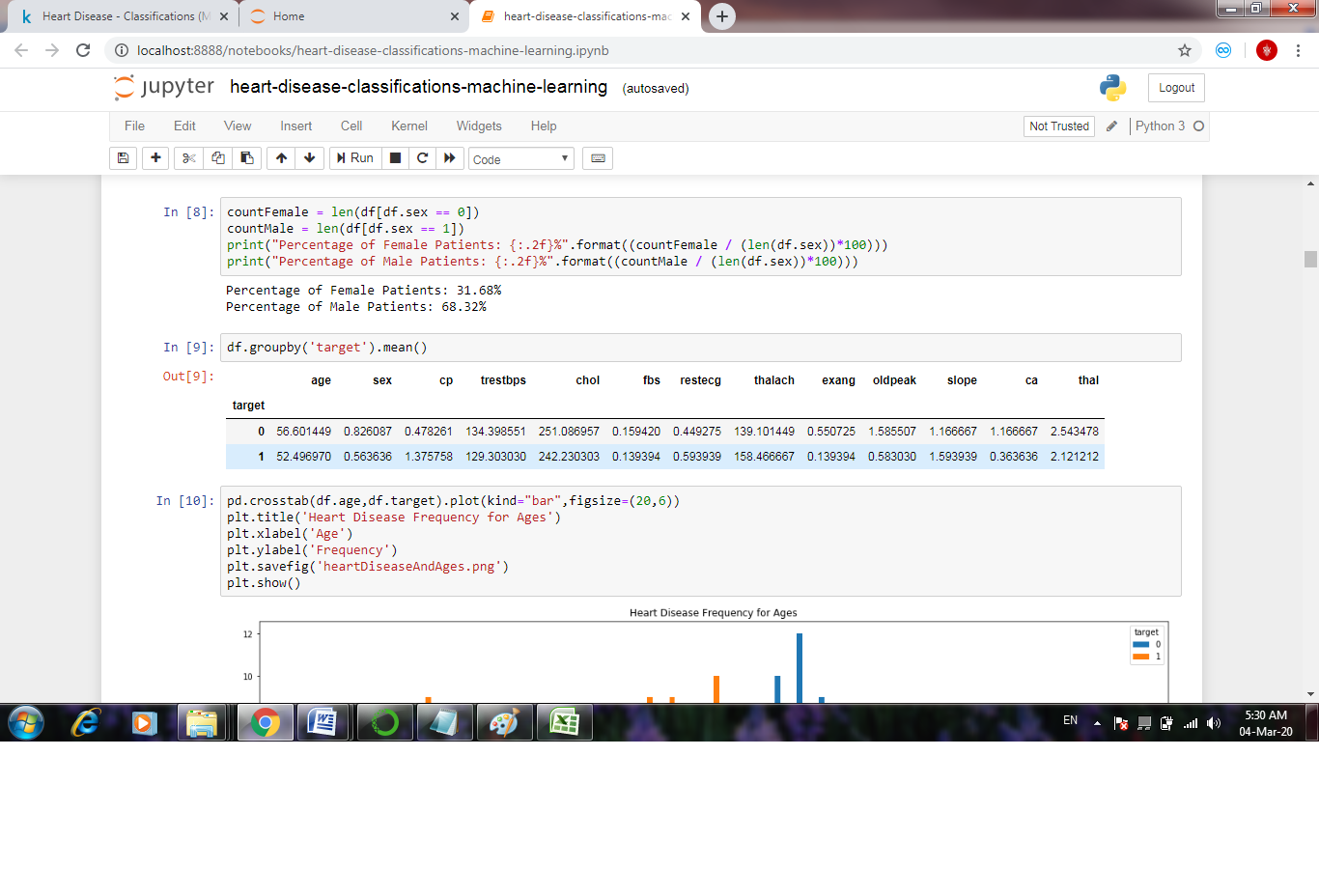
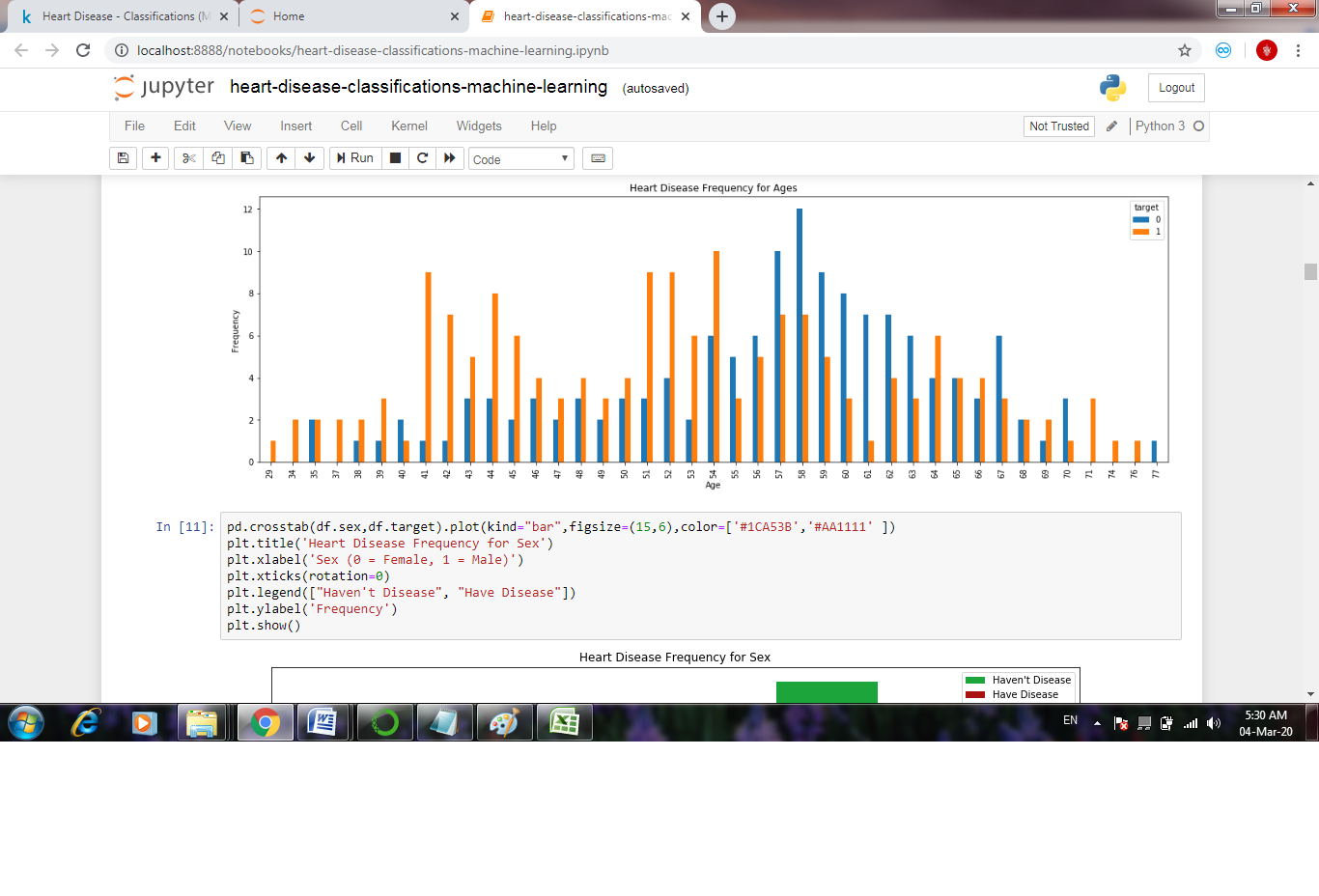
* age - age in years
* sex - (1 = male; 0 = female)
* cp - chest pain type
* trestbps - resting blood pressure (in mm Hg on admission to the hospital)
* chol - serum cholestoral in mg/dl
* fbs - (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
* restecg - resting electrocardiographic results
* 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)

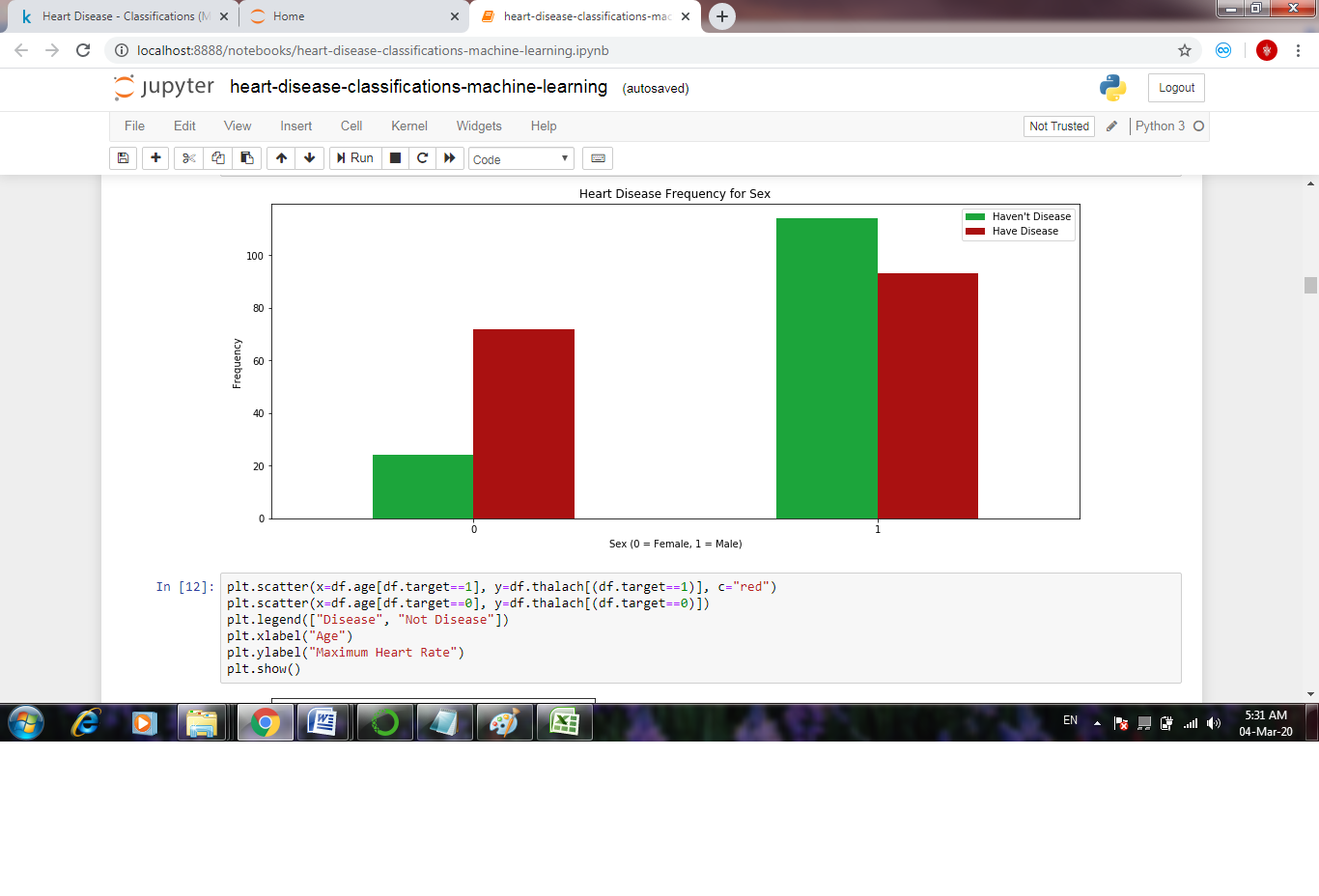
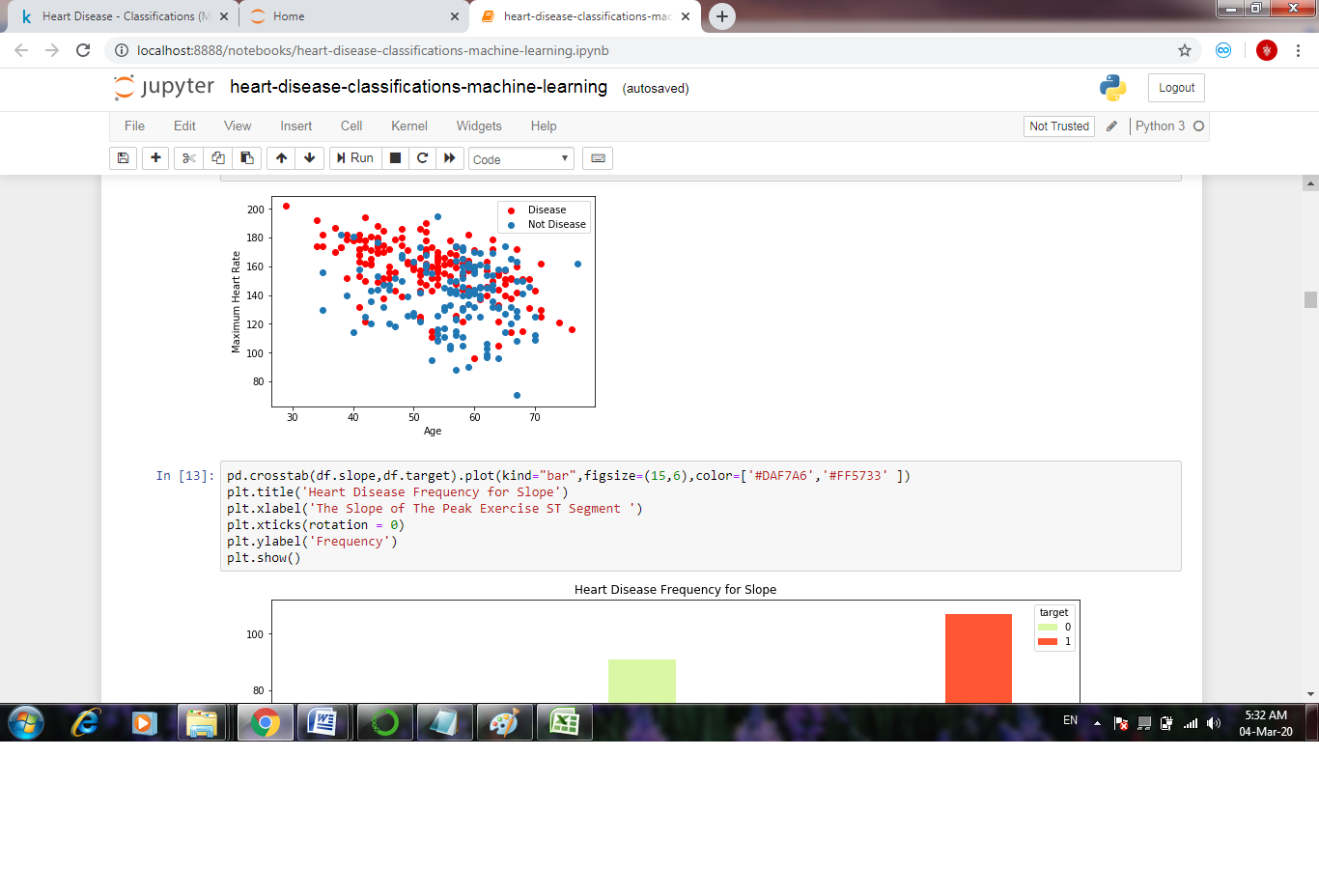
**PRE-PROCESSING:**

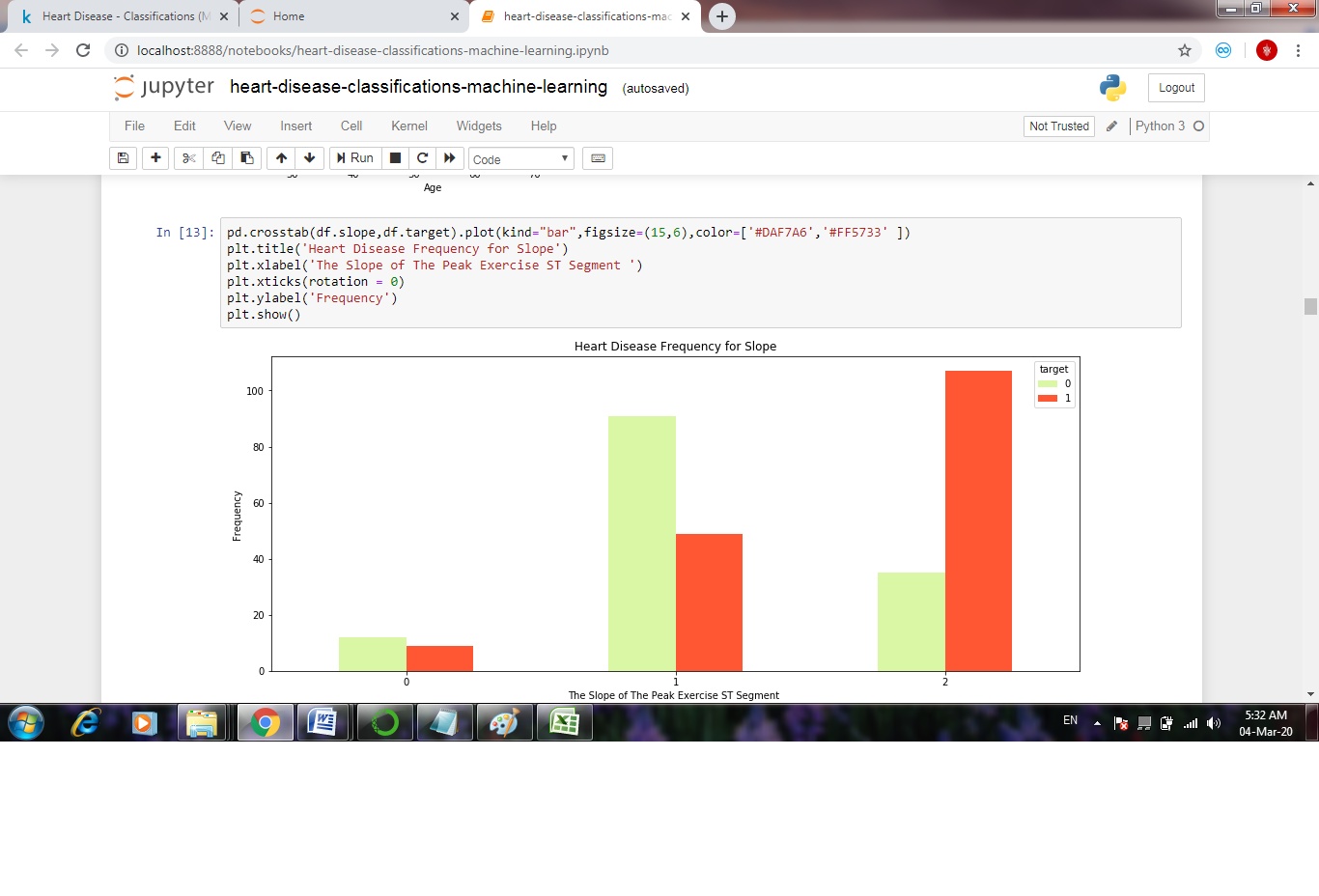
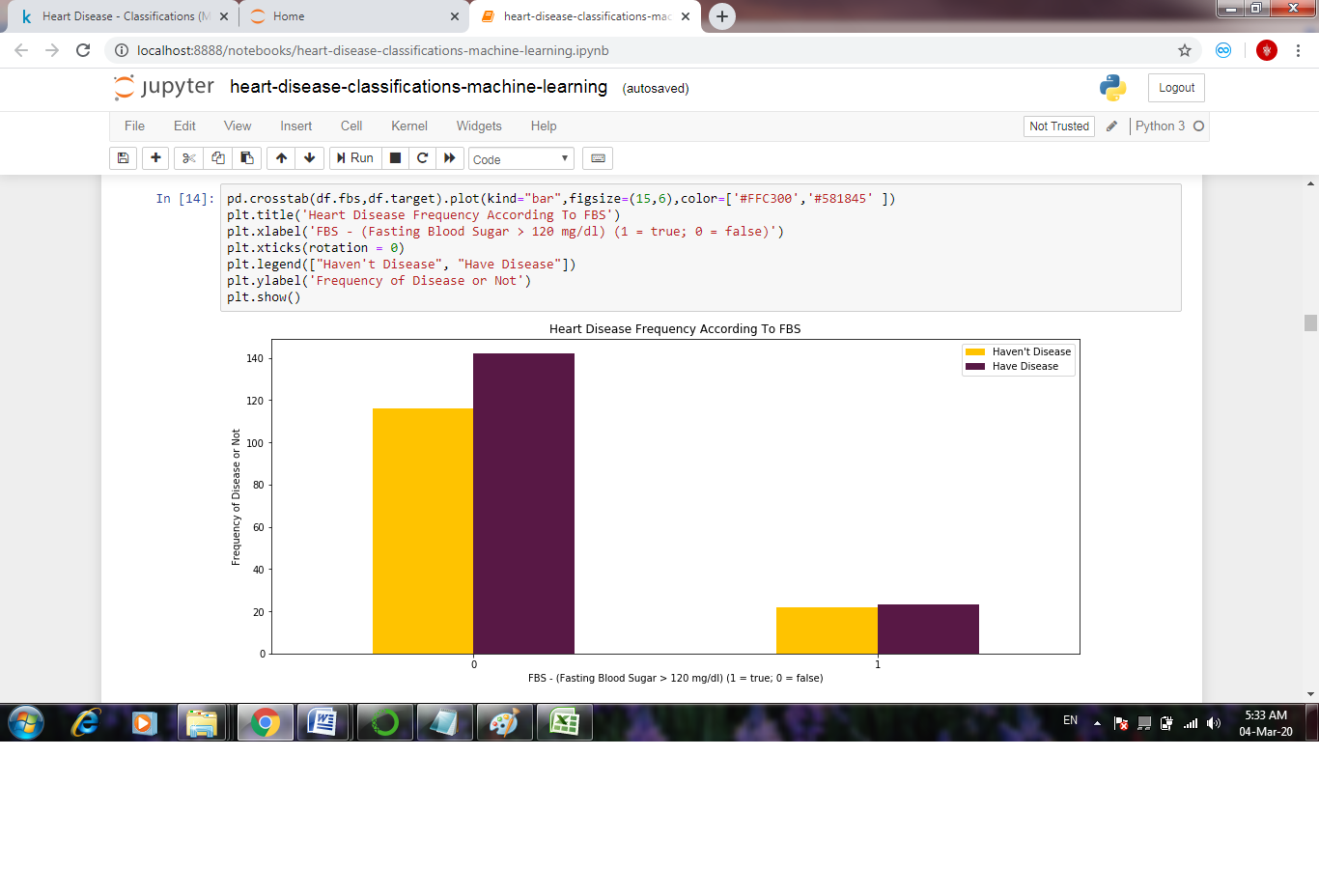
Pre-processing refers to the transformations applied to our data before feeding it to the algorithm.

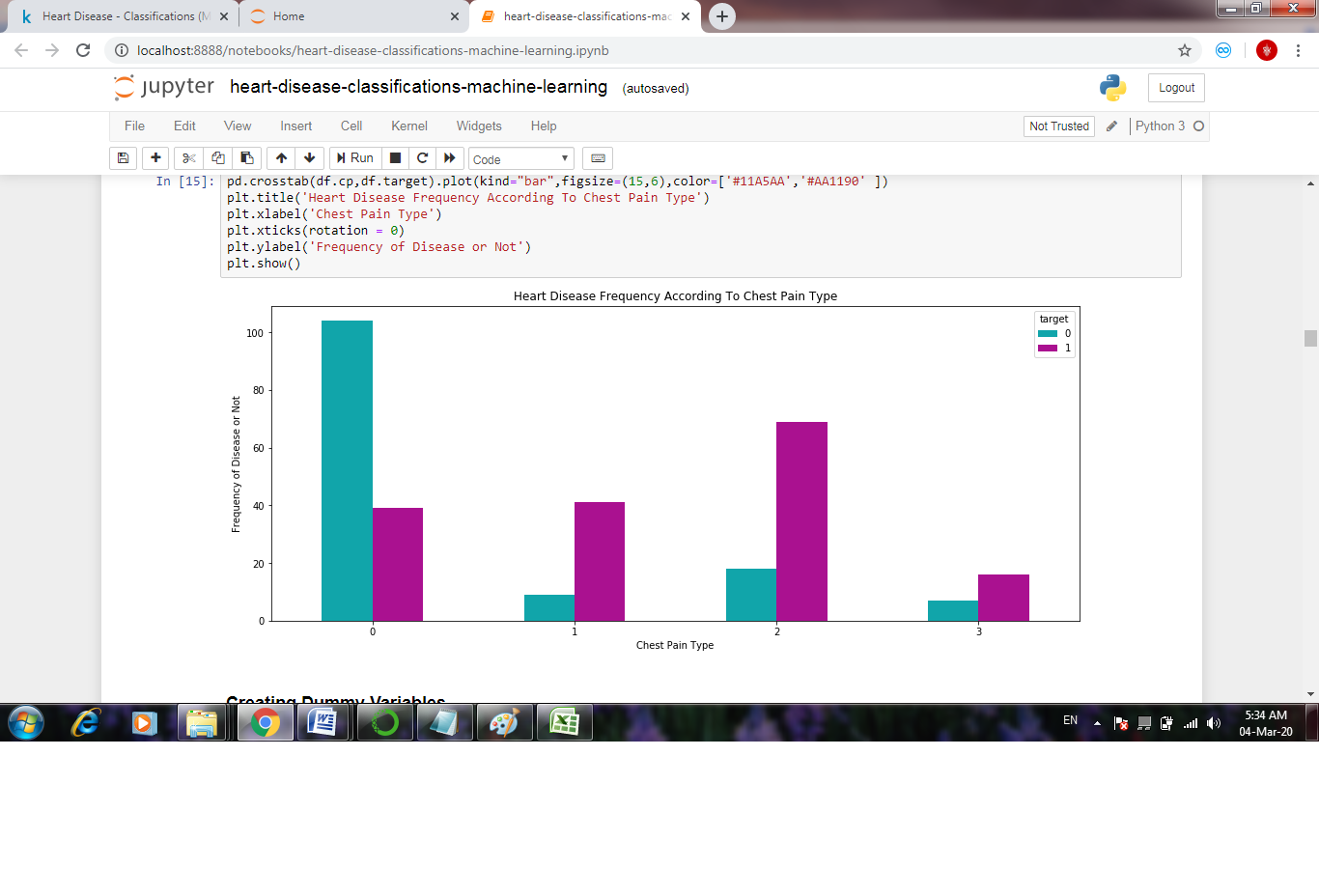
 

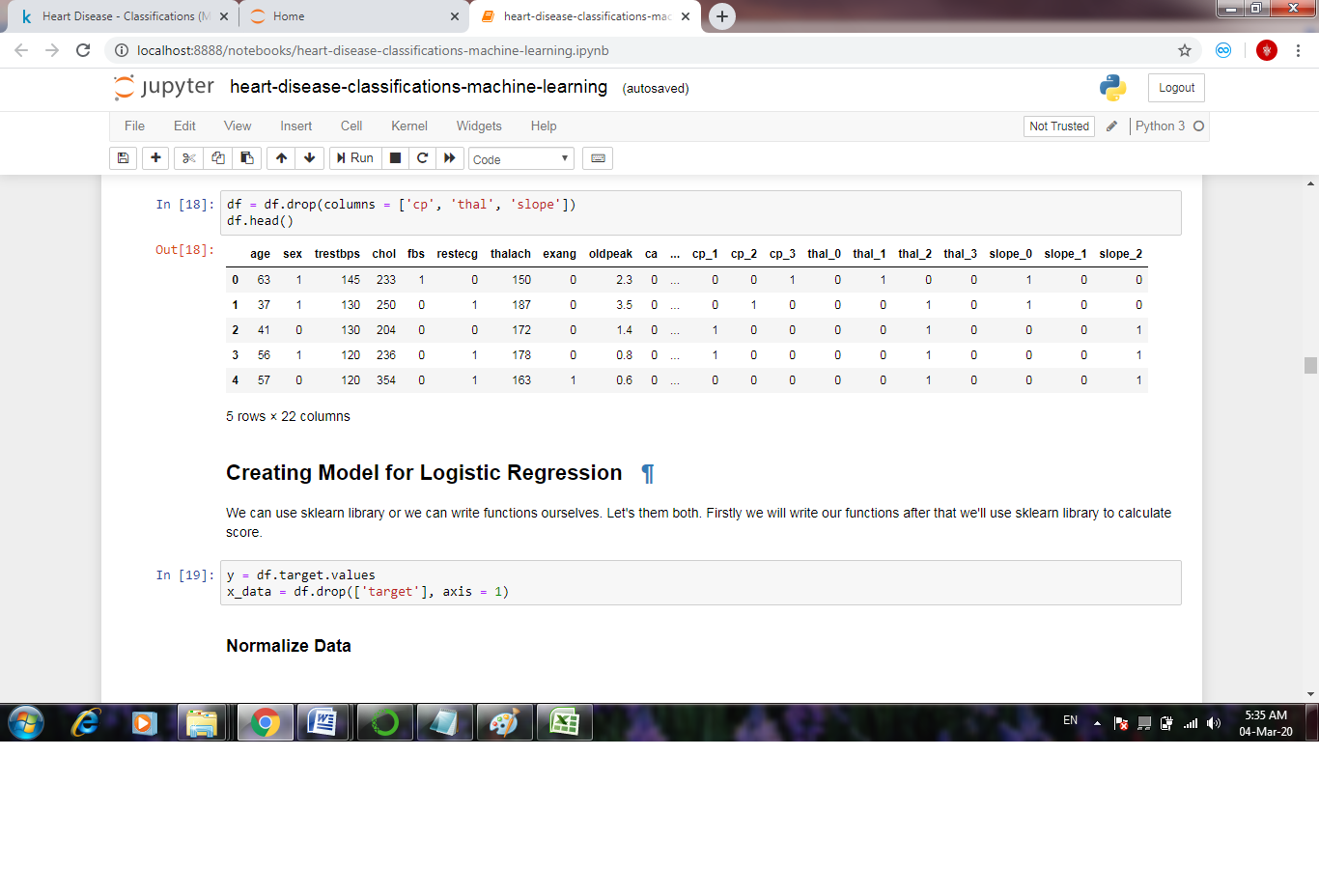
 

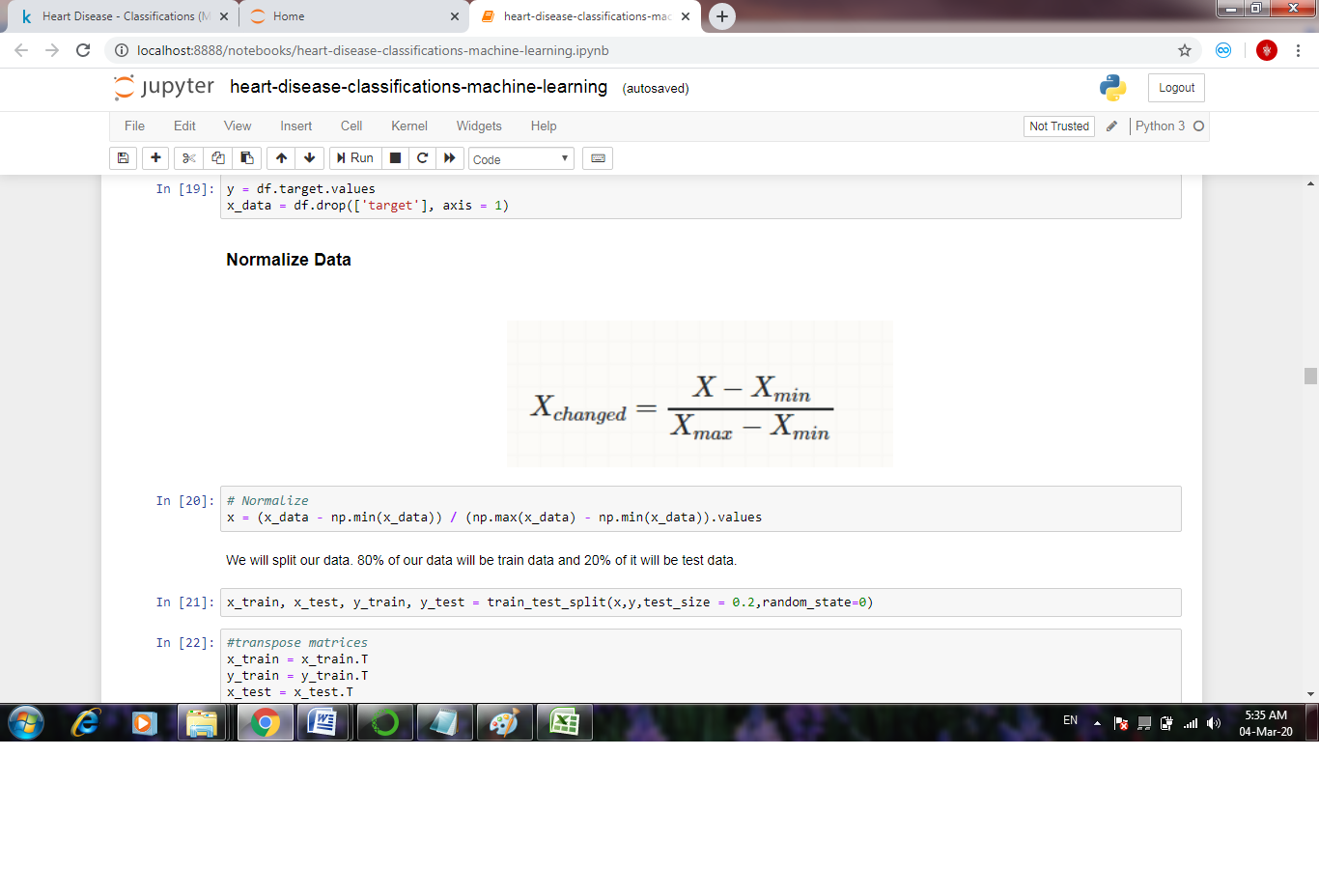
**** ****

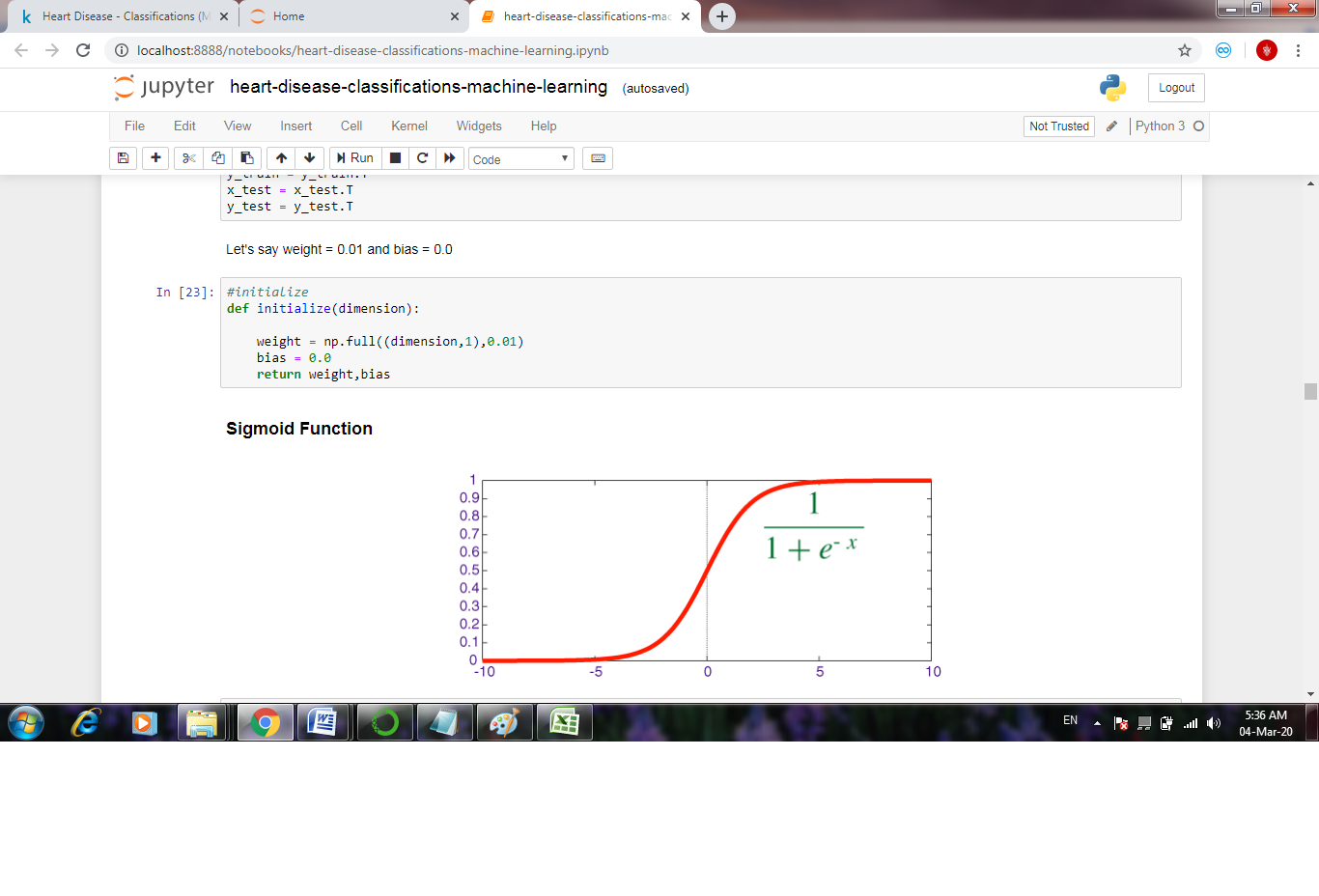
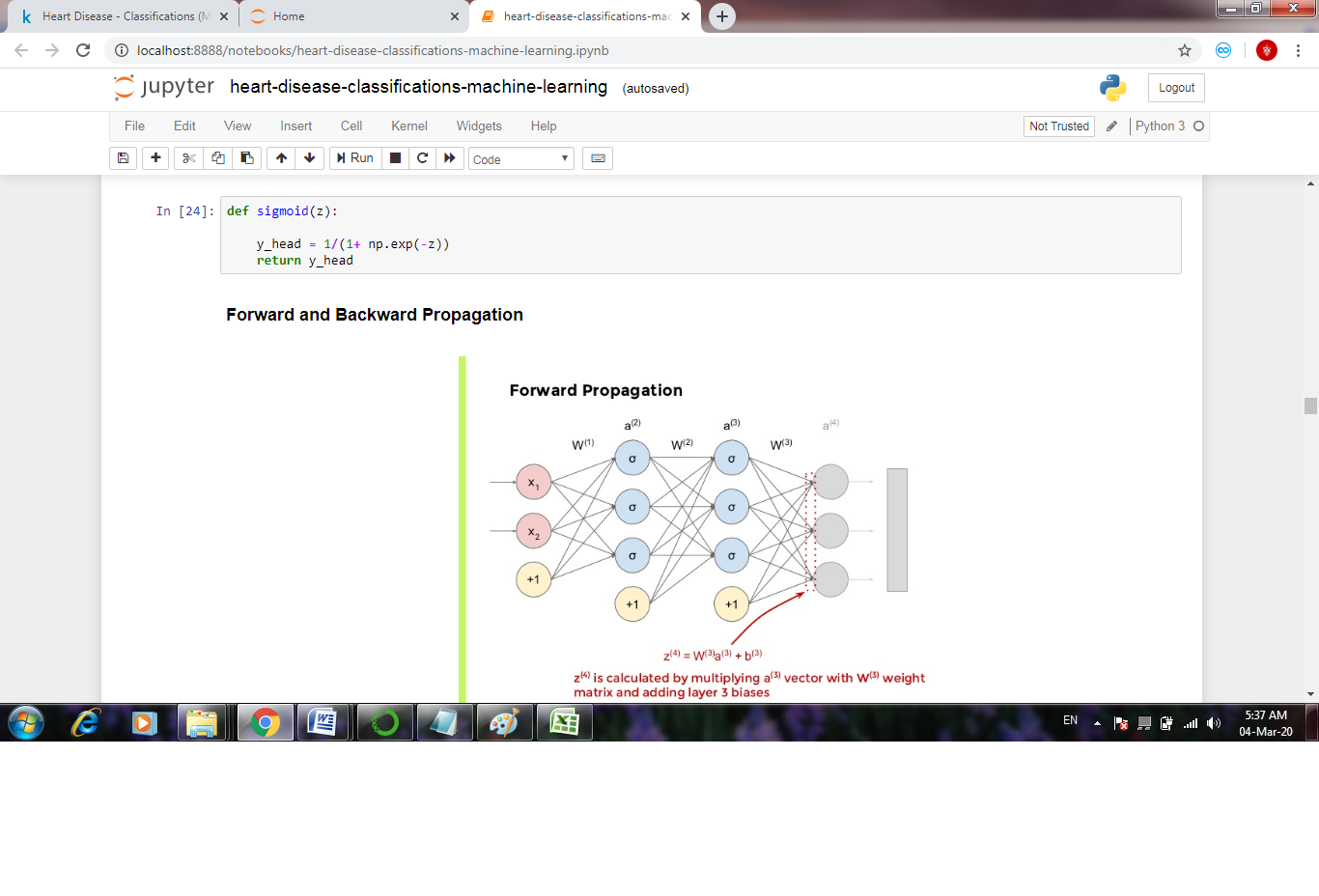
 

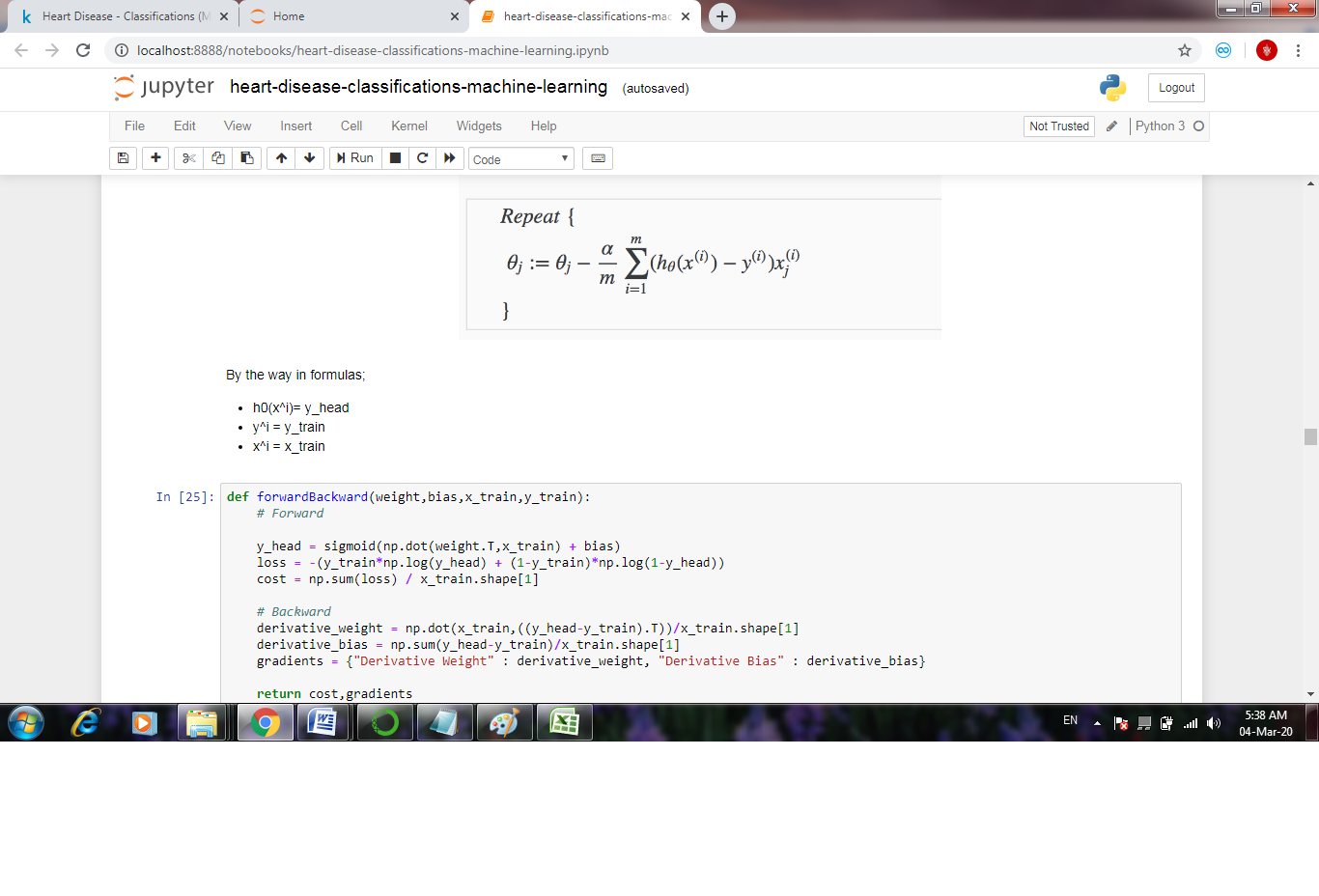
 

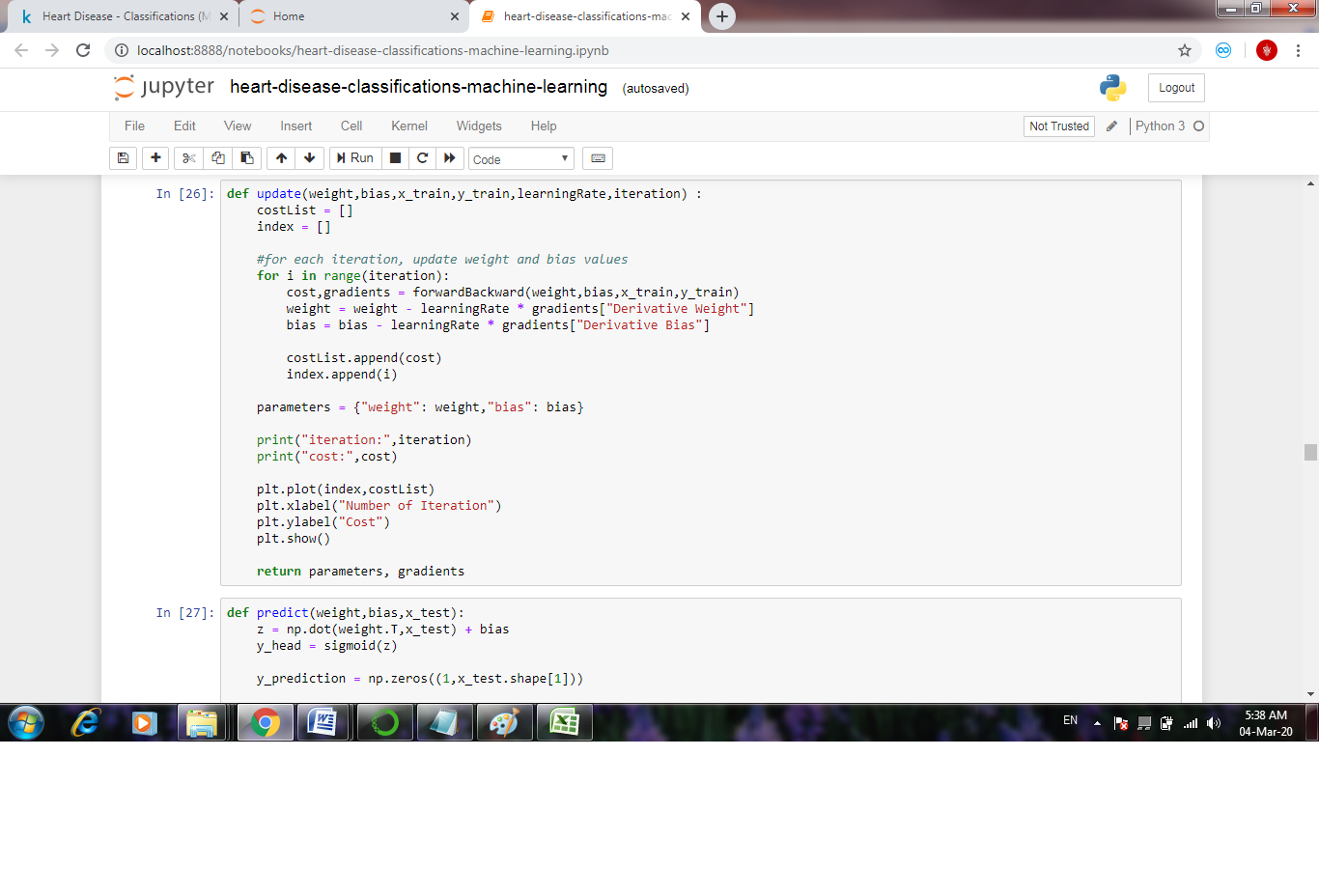
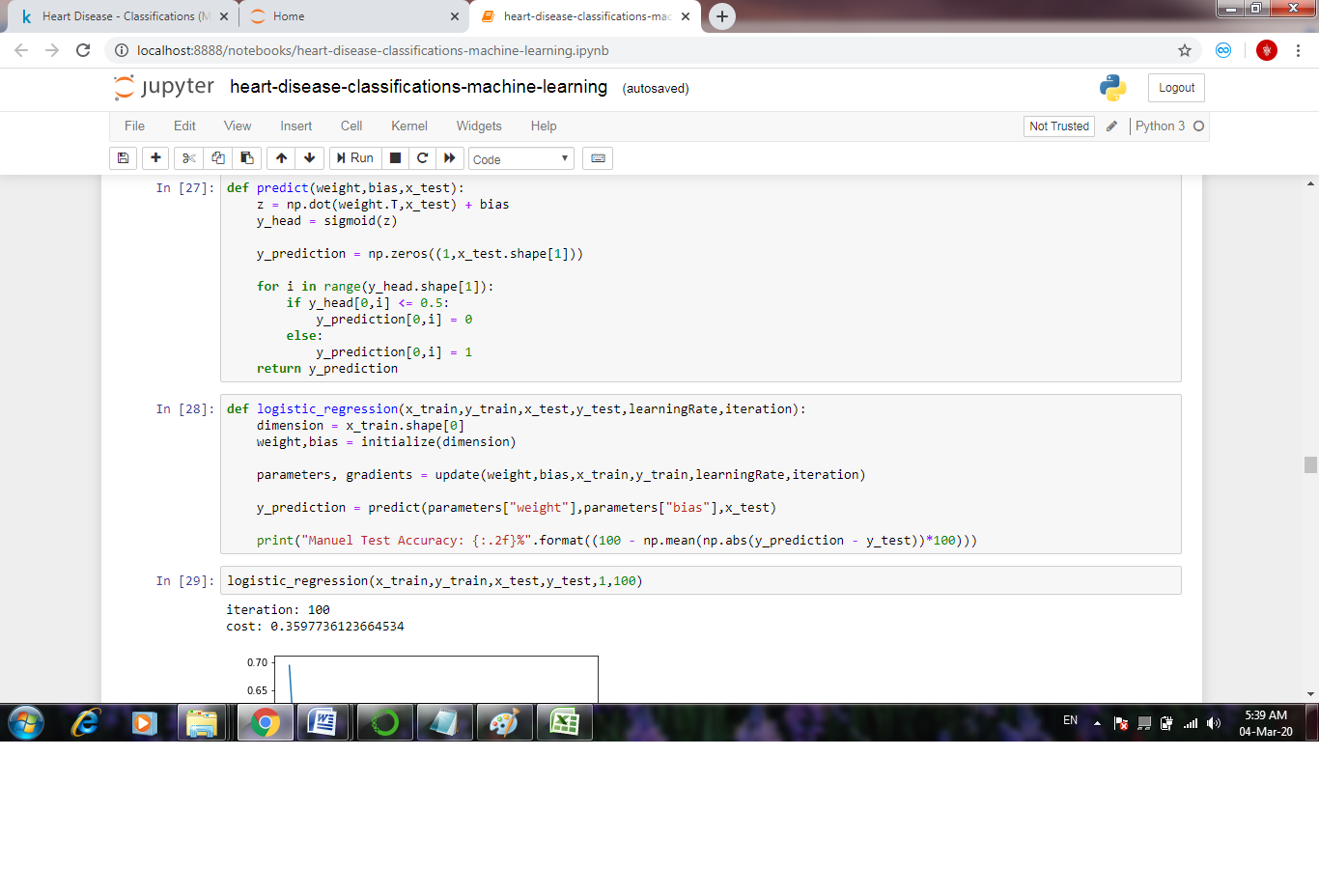
****

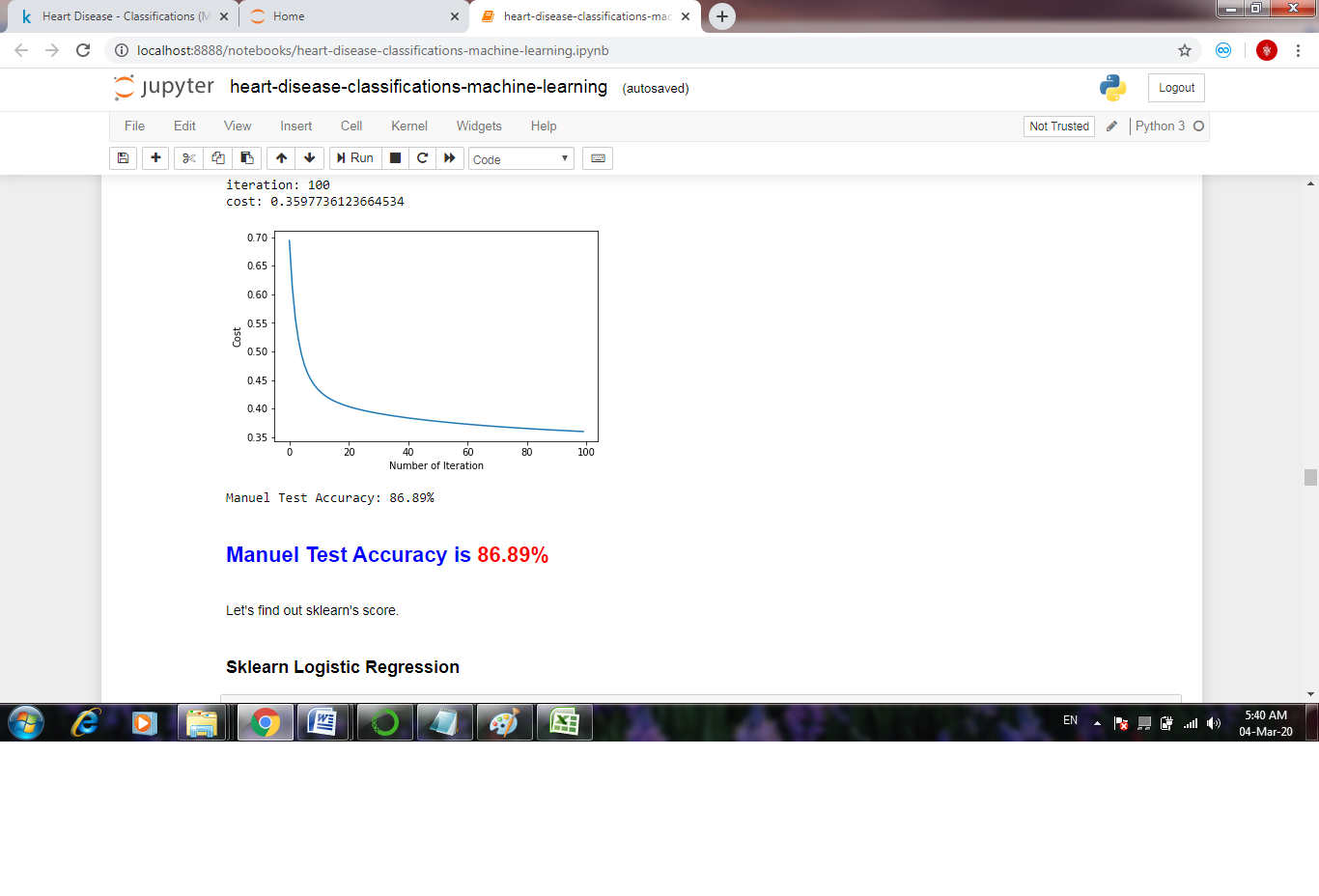
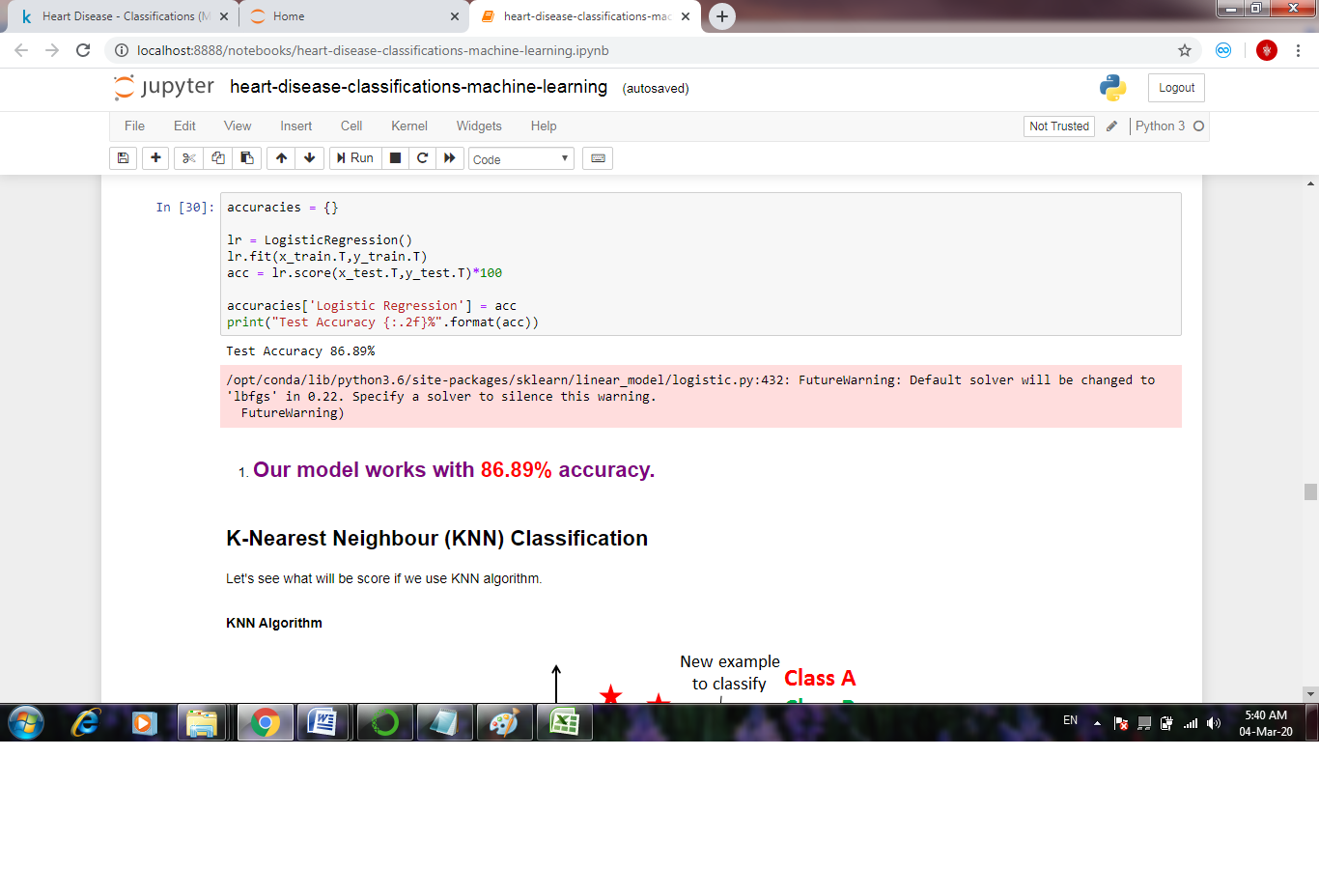
****

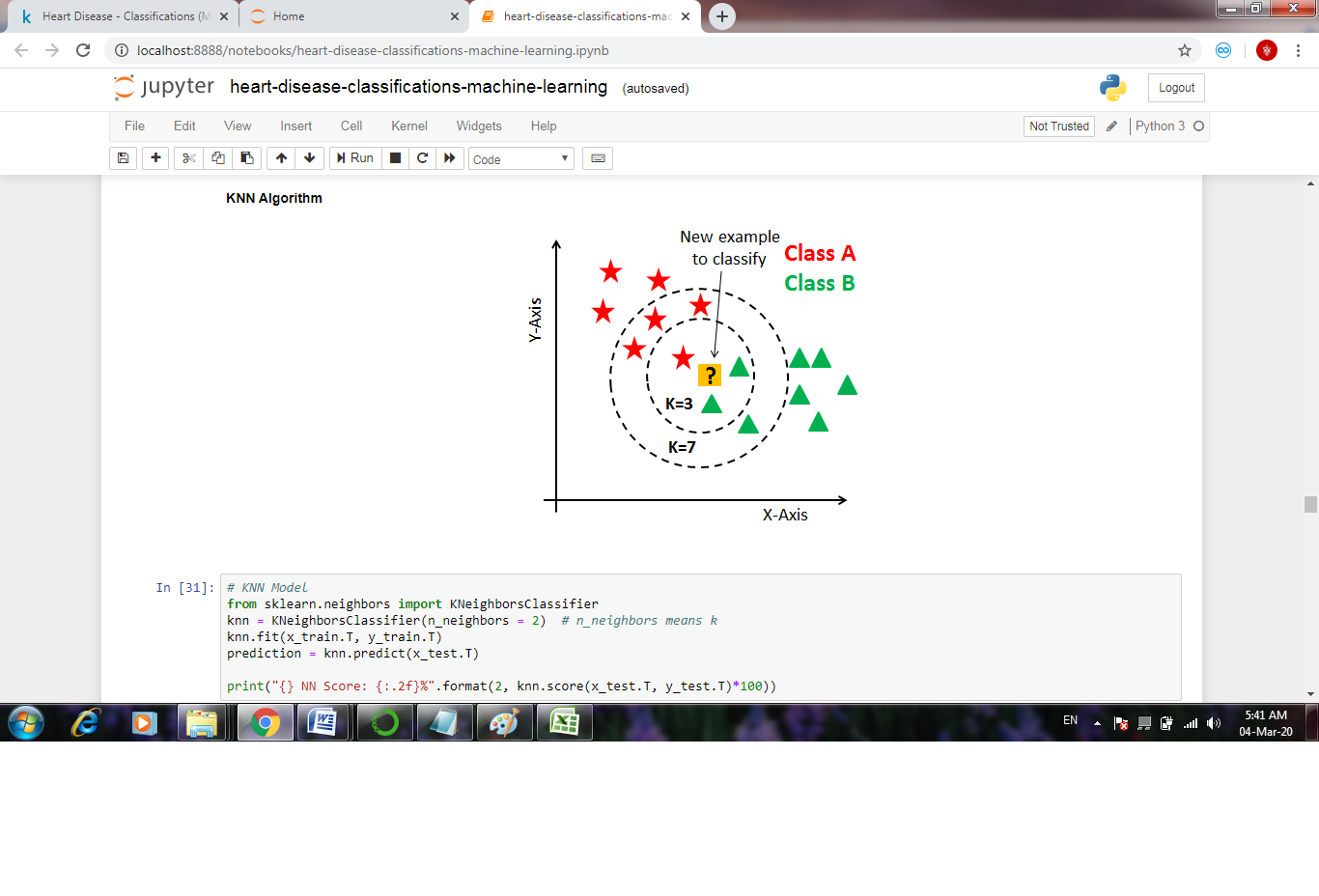
****

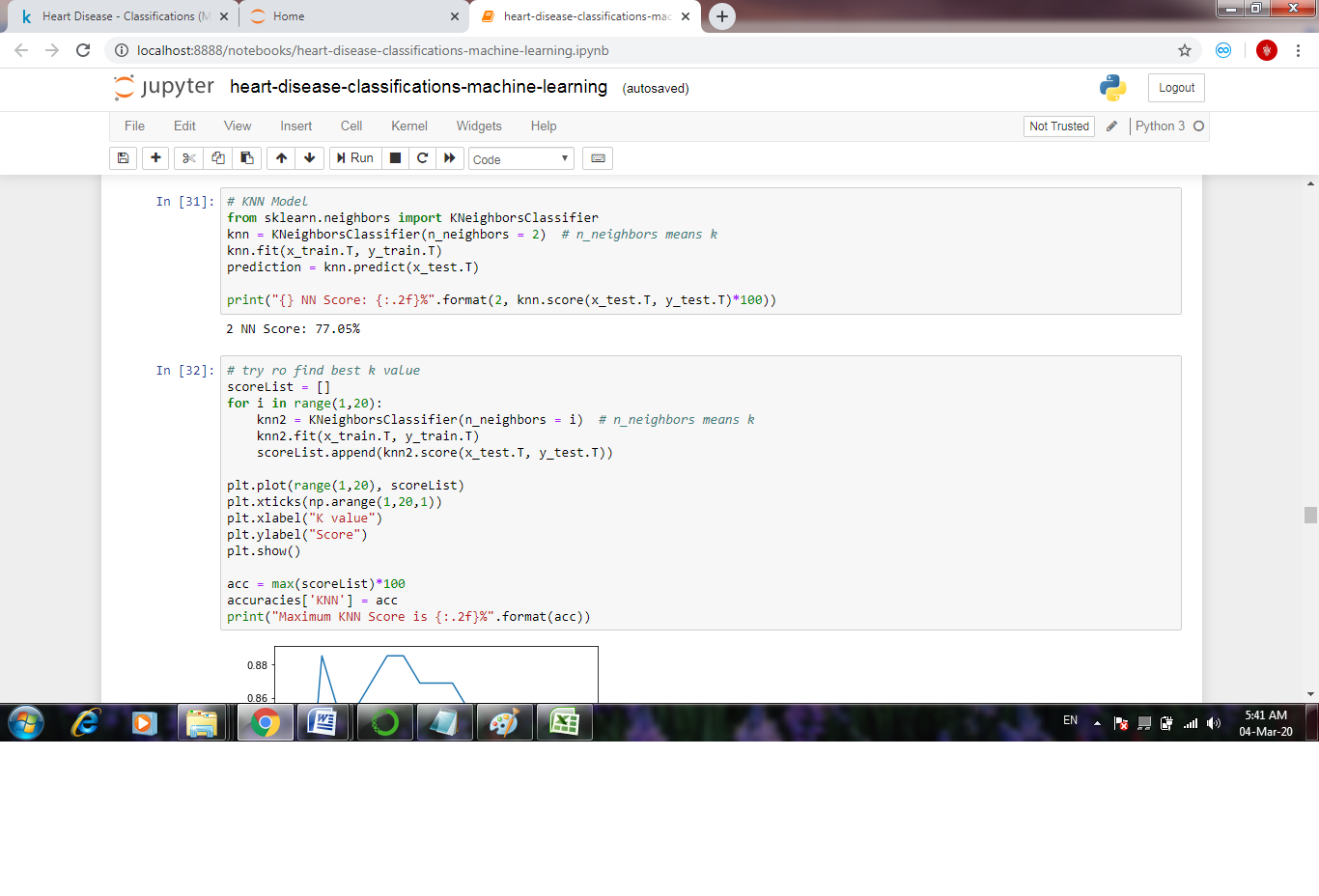
****

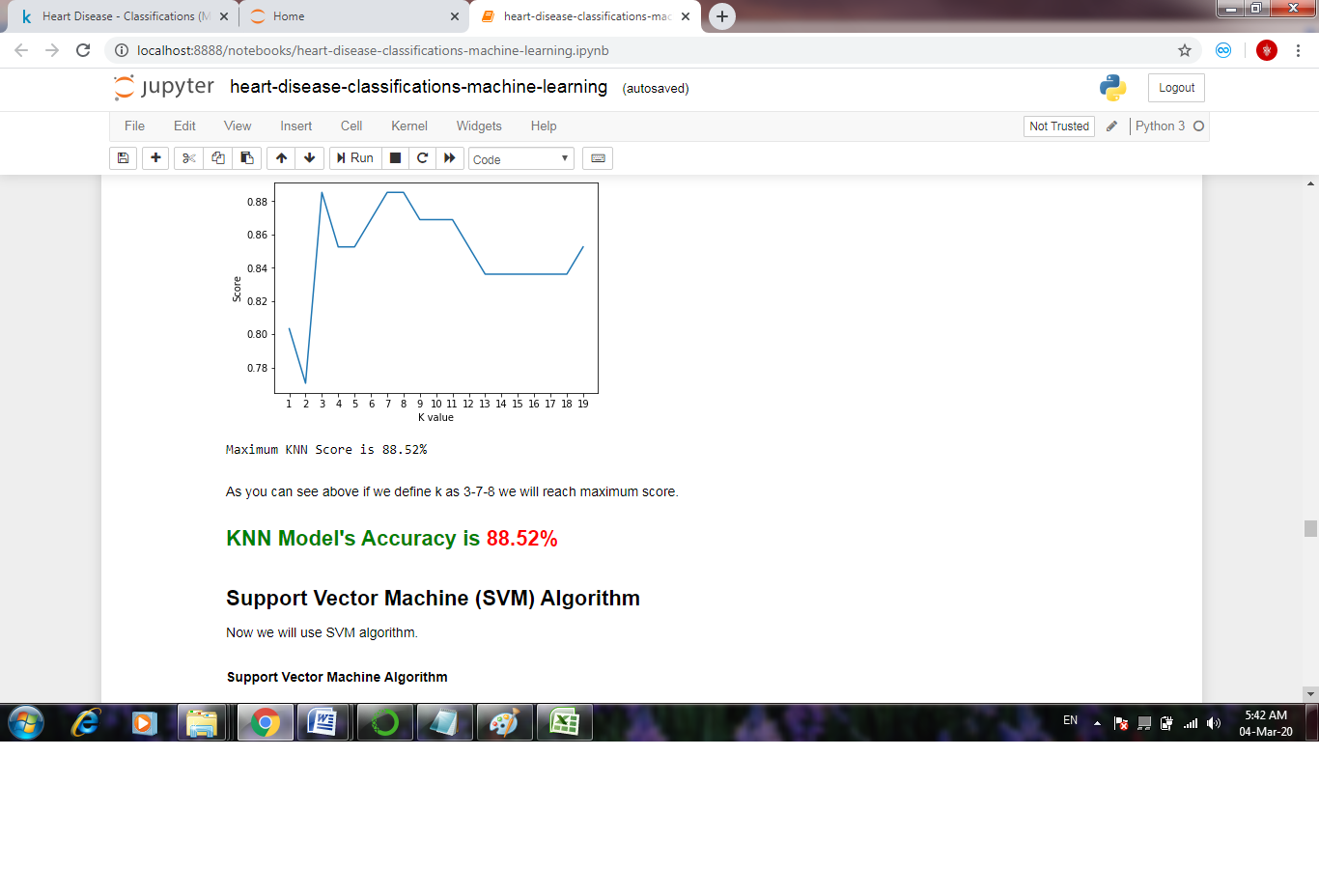
****

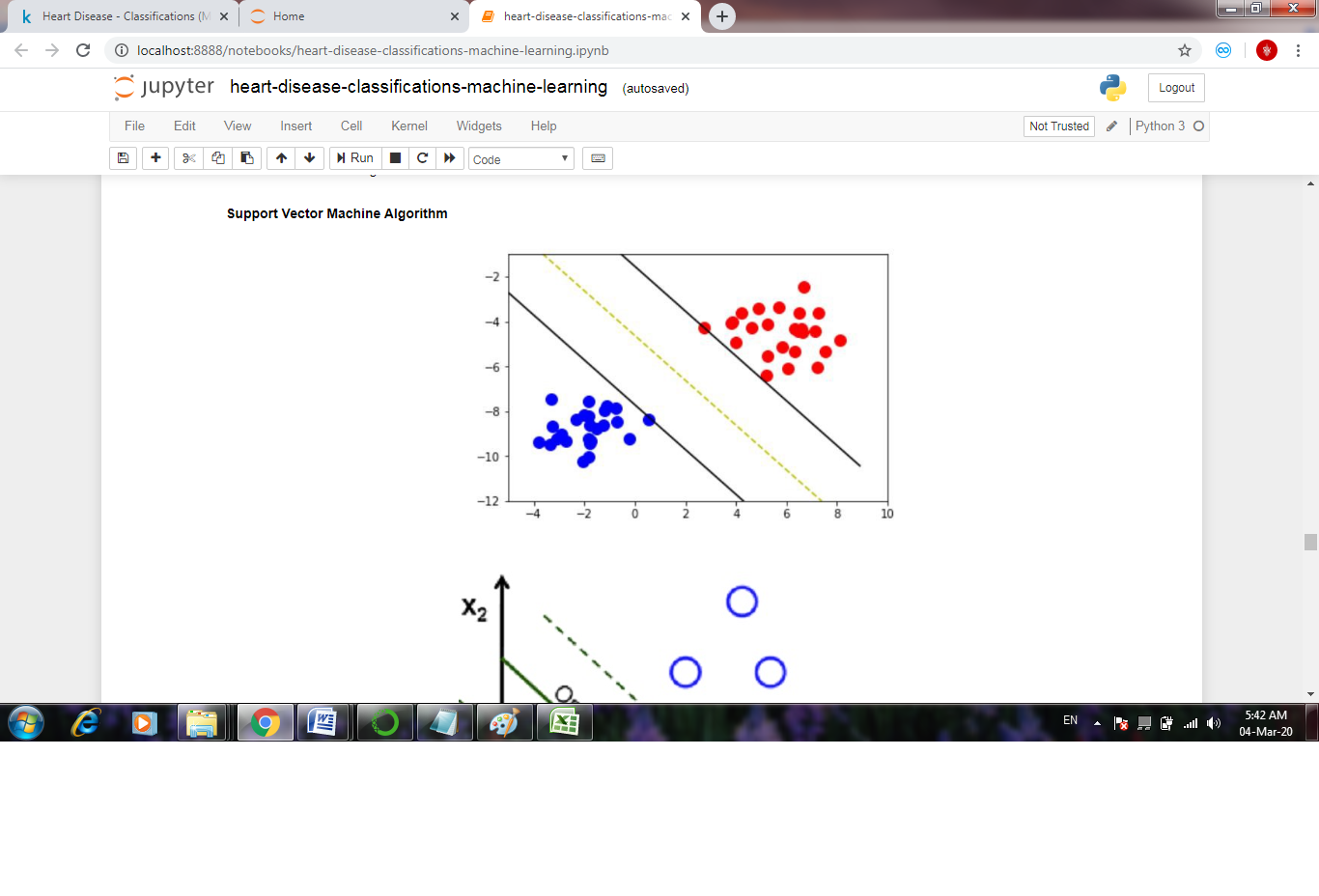
**** ****

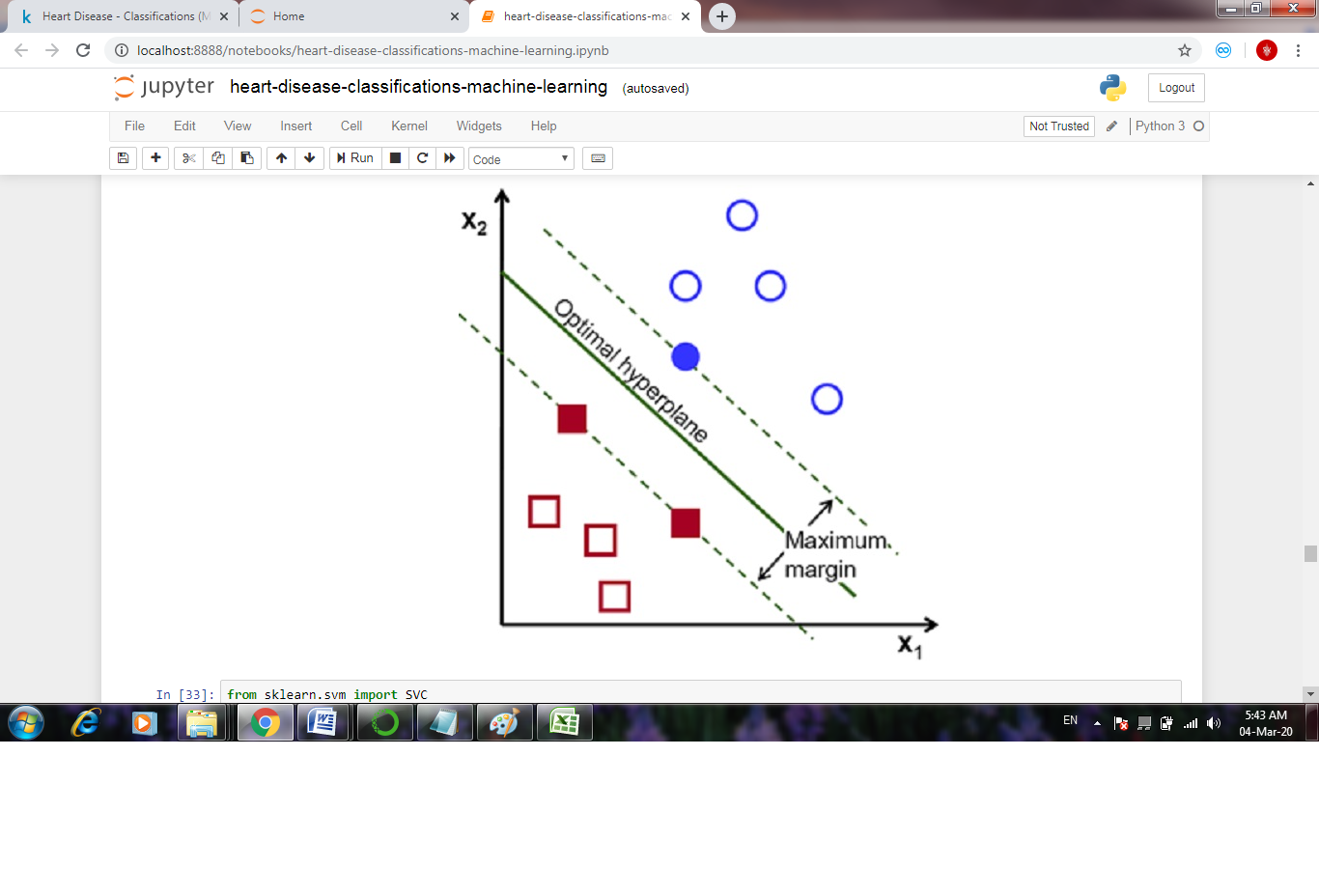
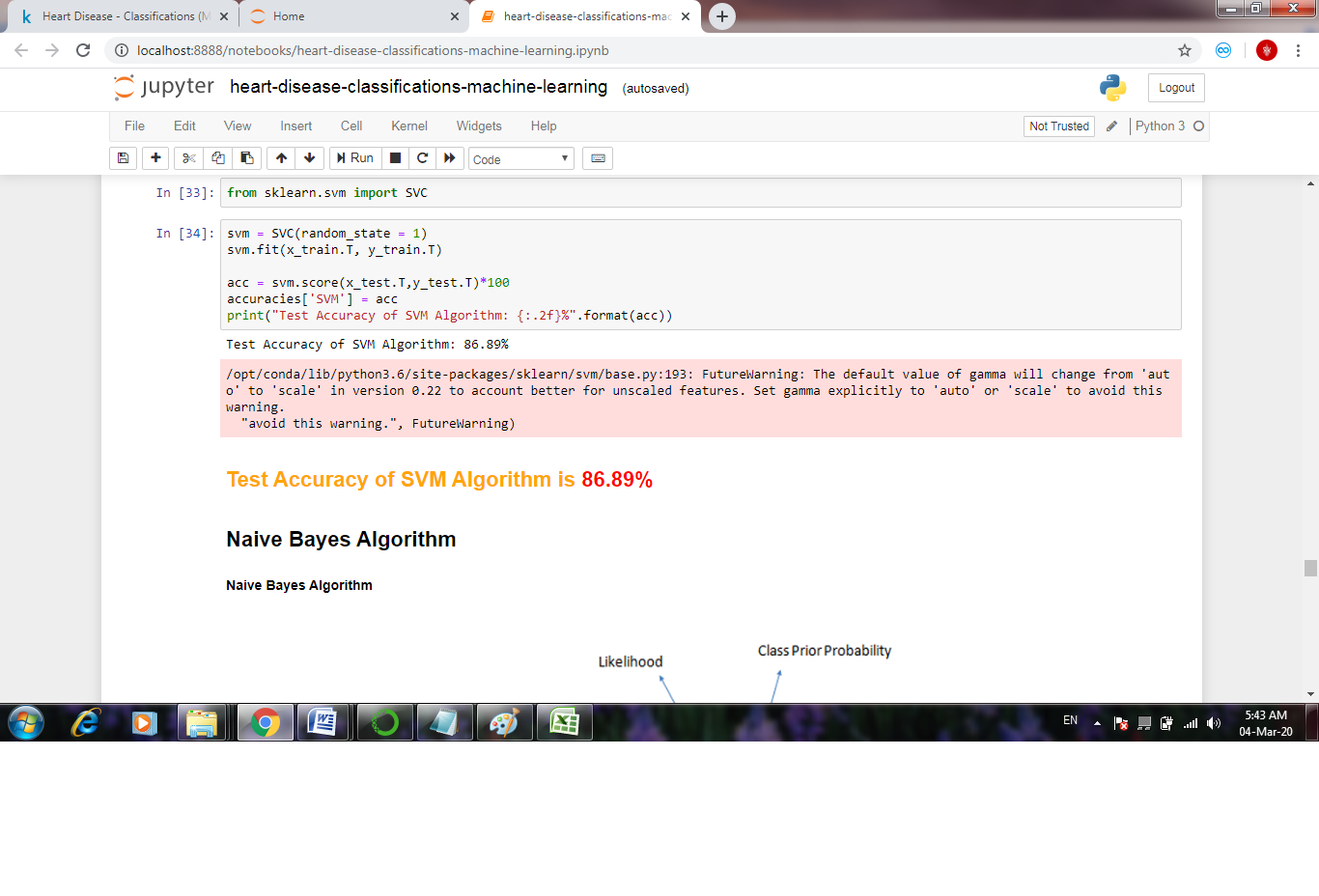
**** ****

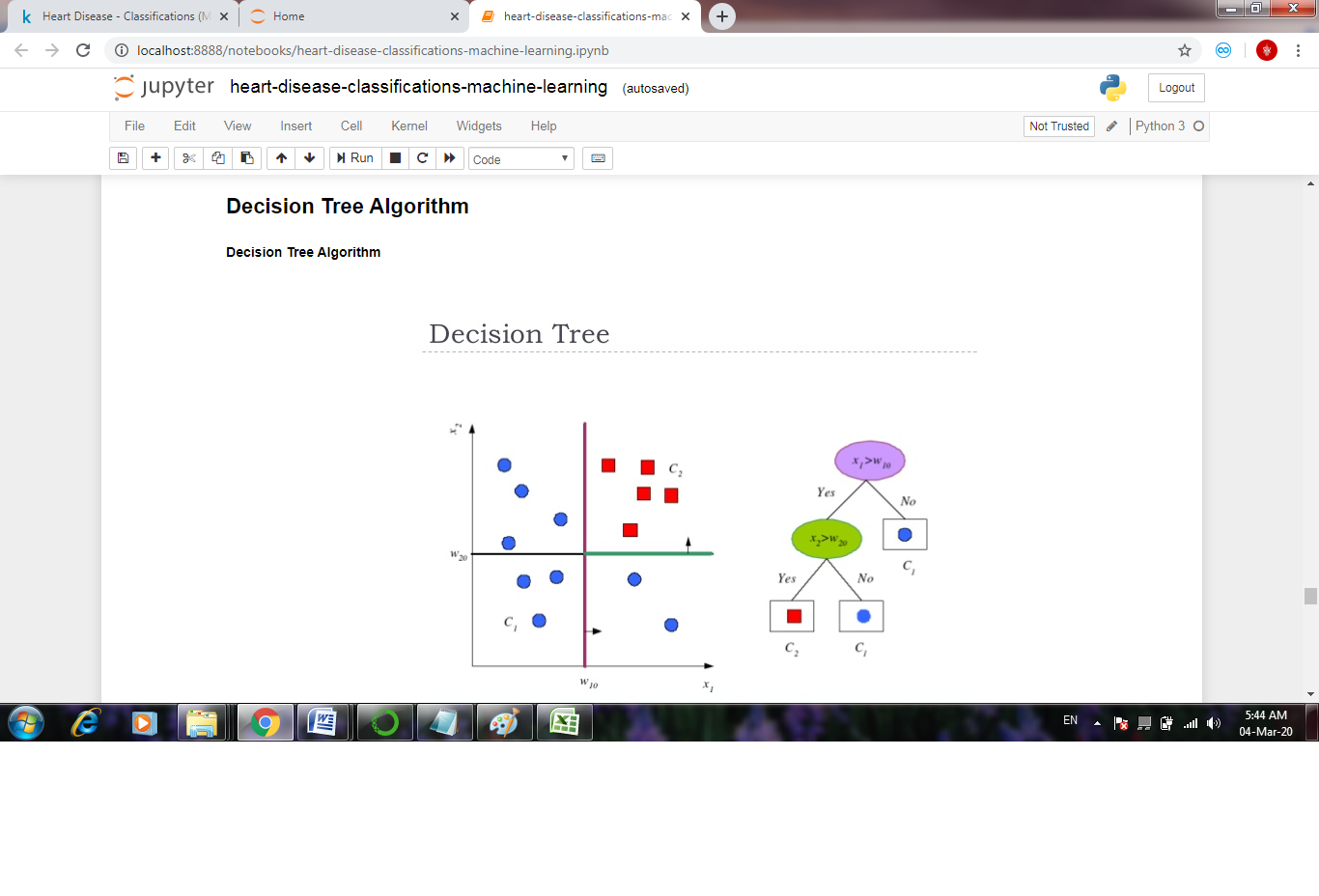
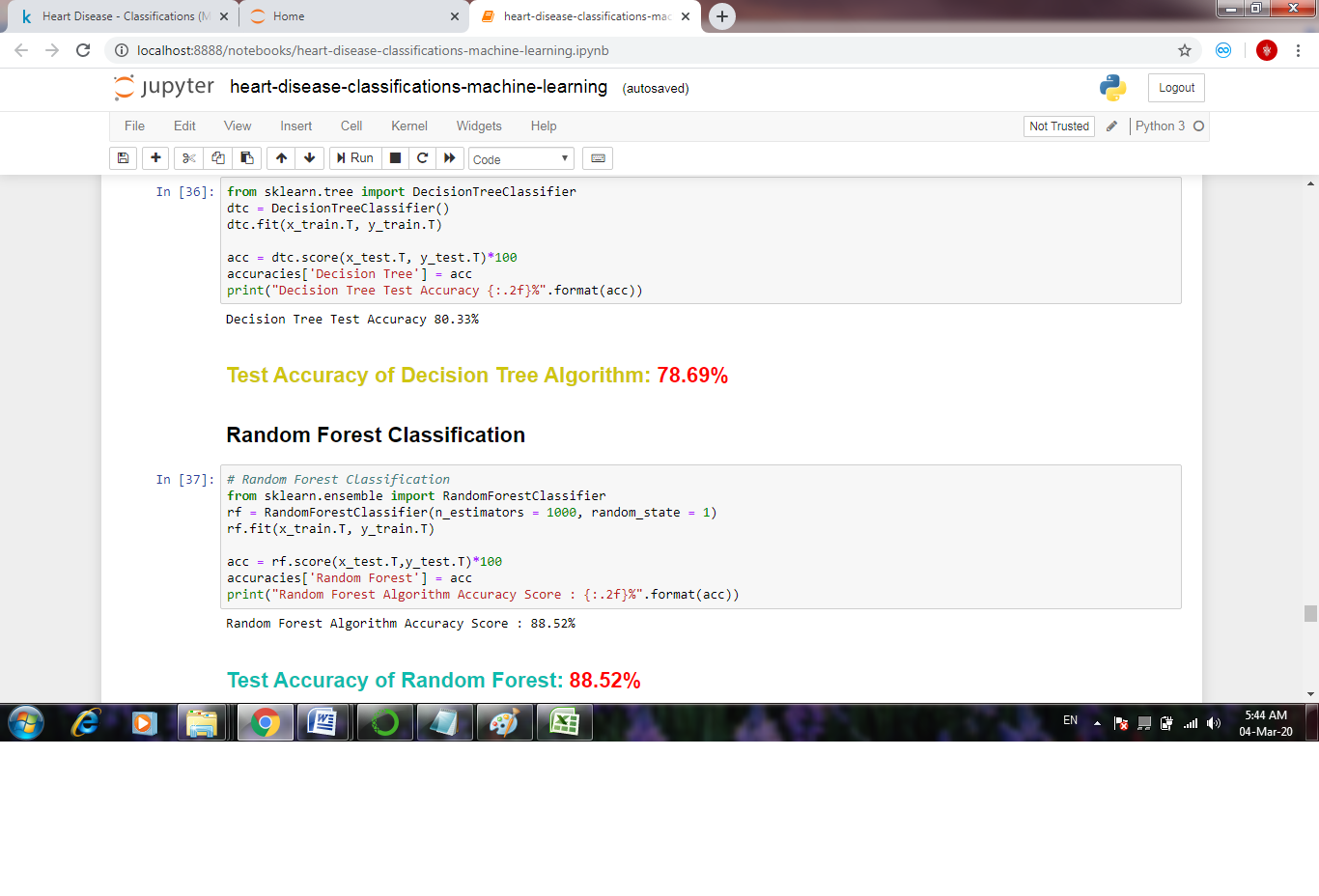


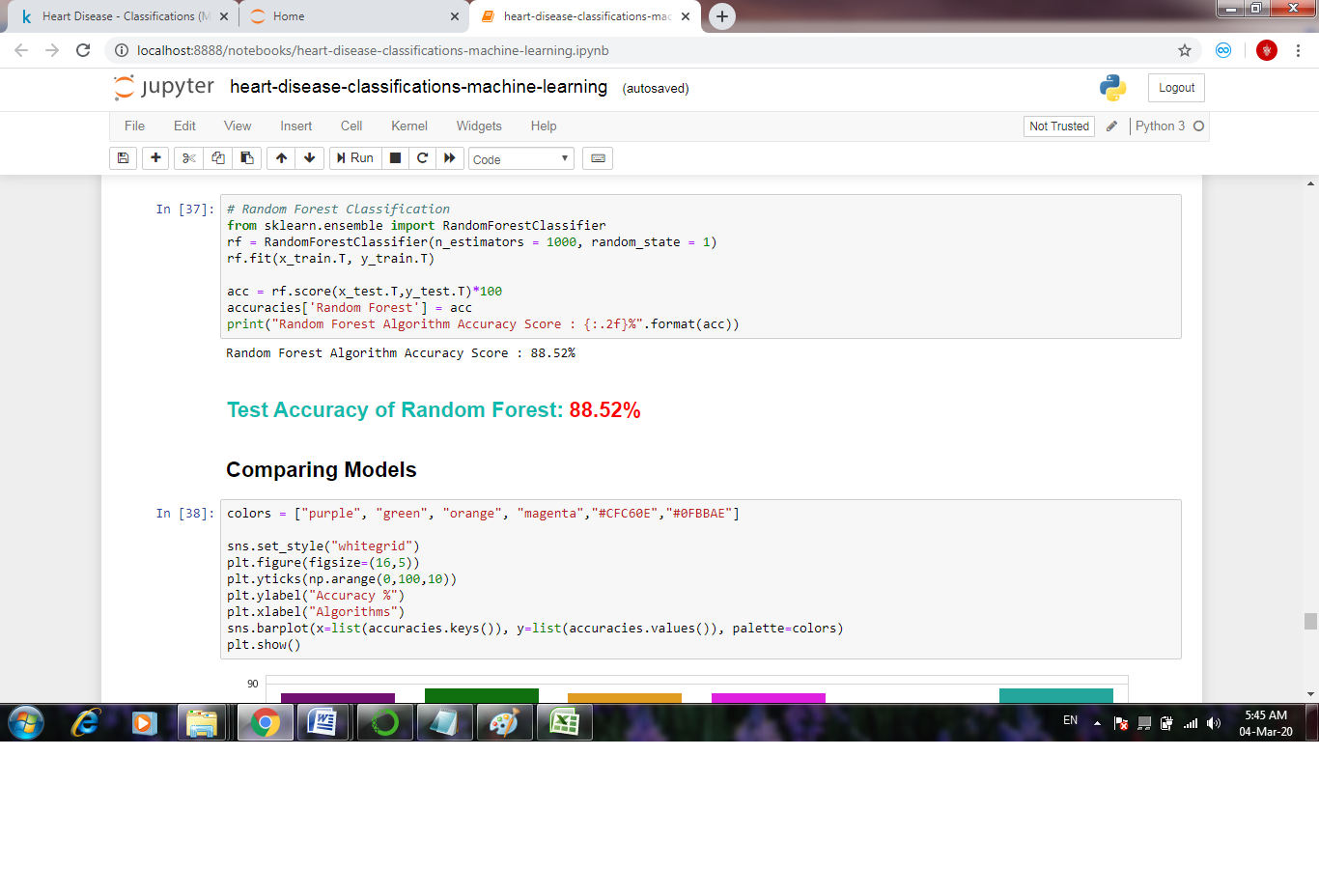
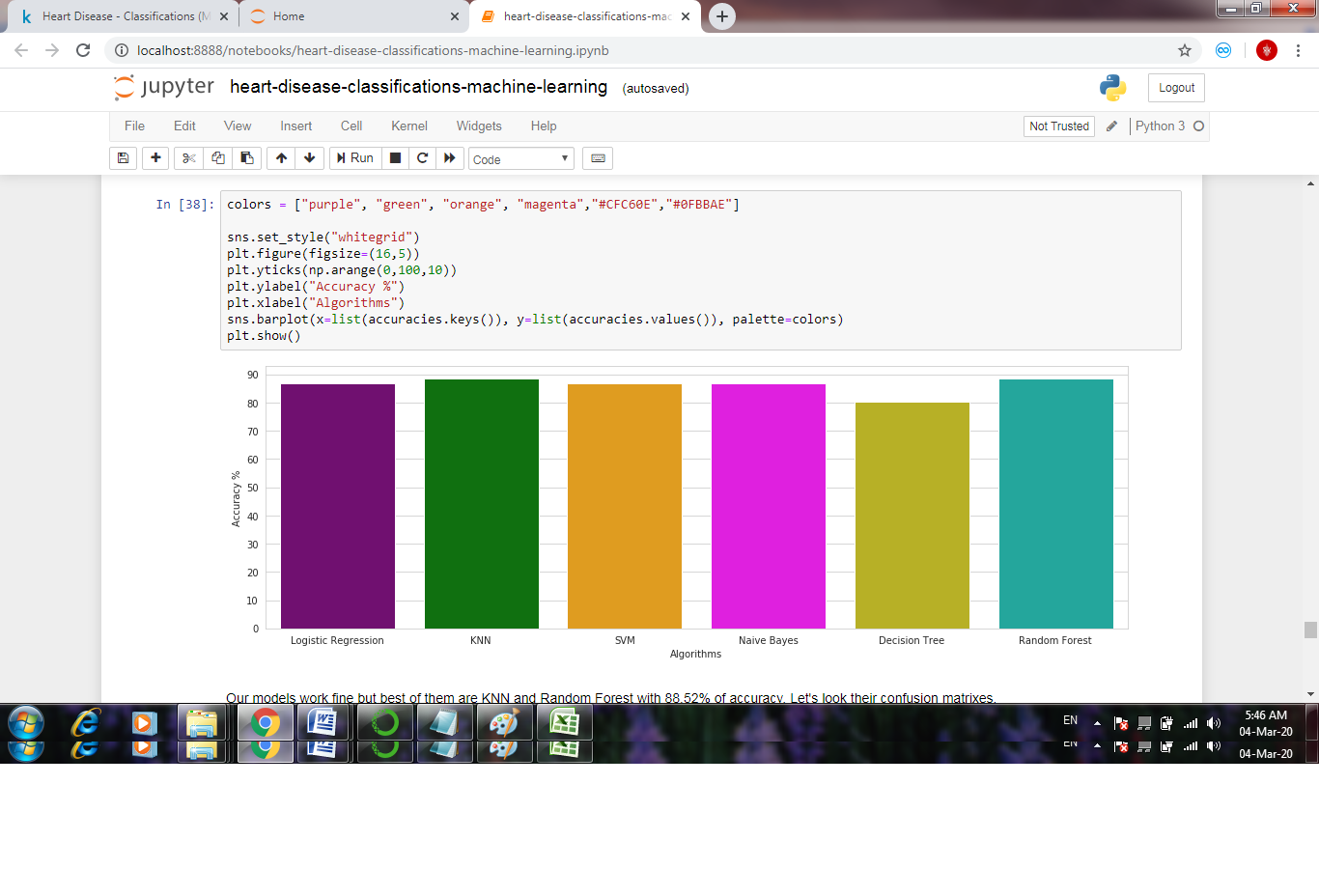


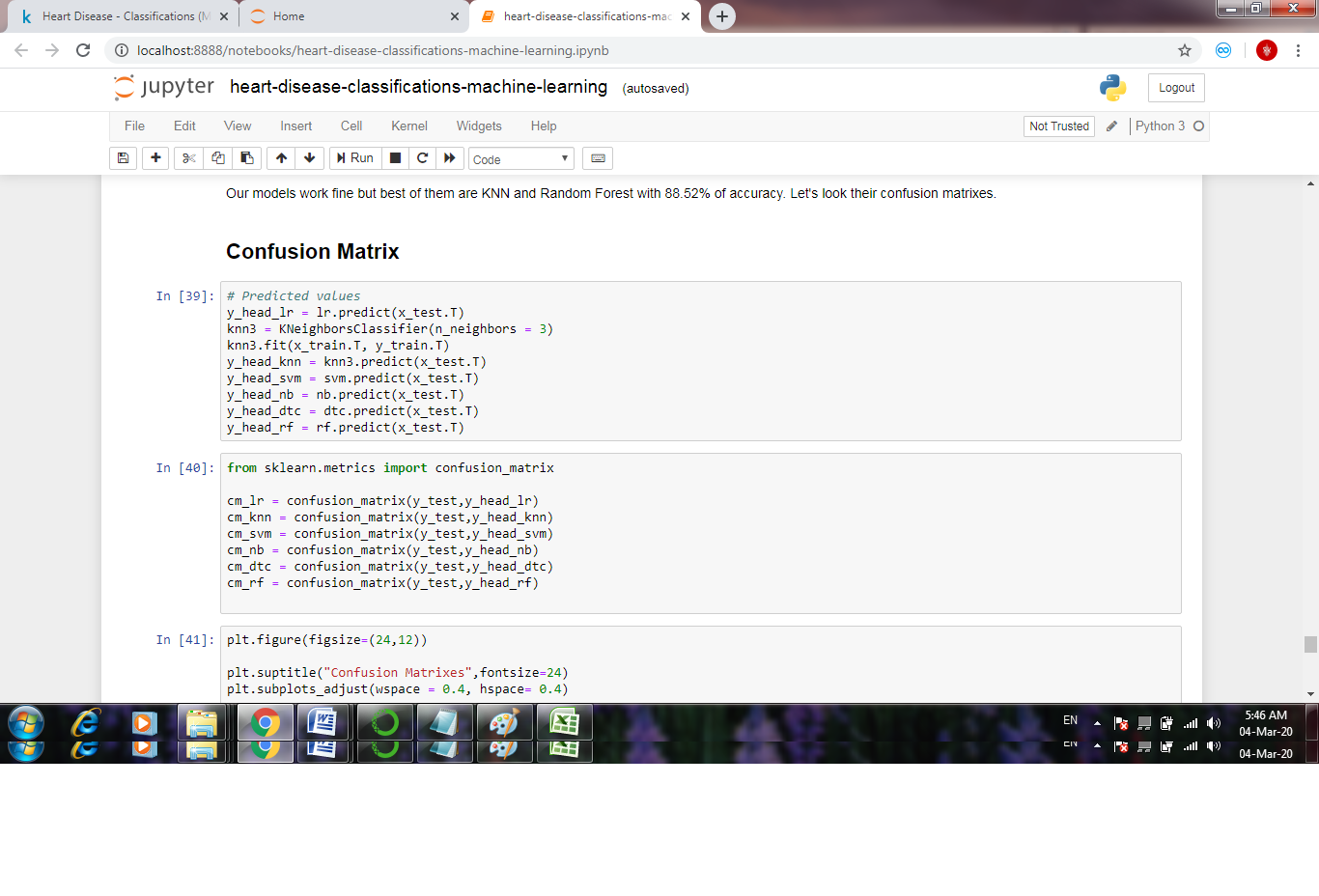
****

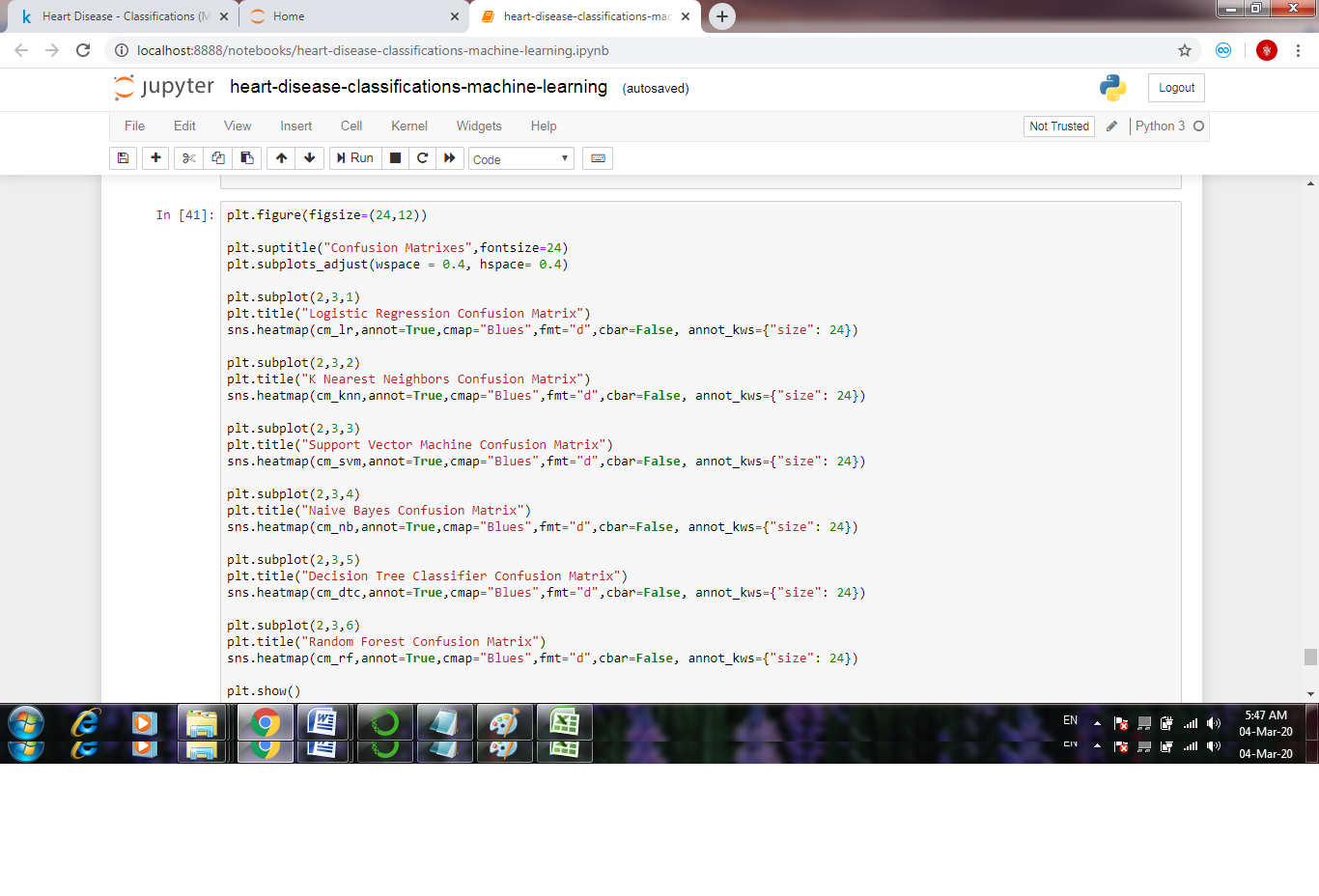
****

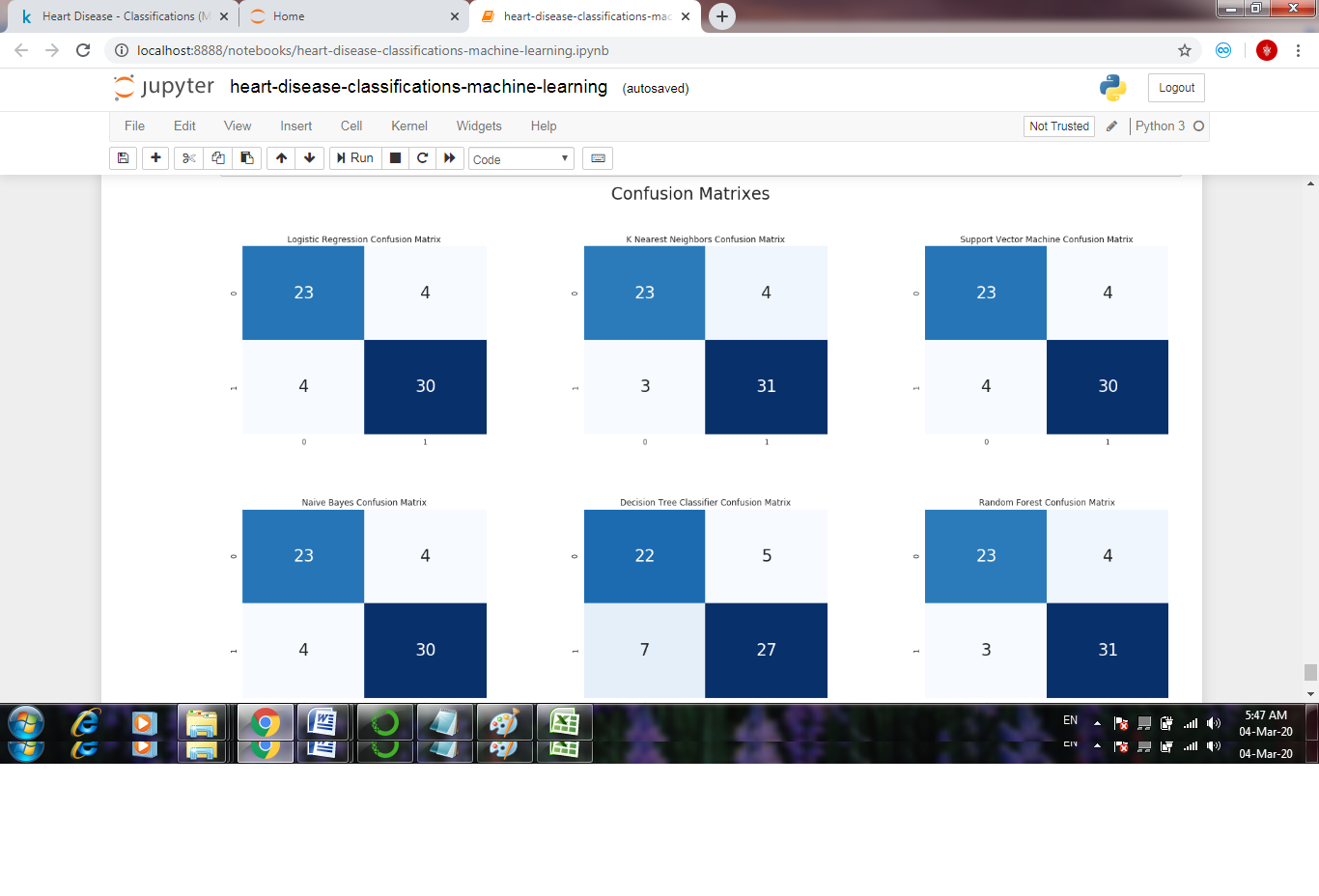
**** ****

**** ****

**** ****

****

****

****

CODING:

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

import os

print(os.listdir("../input"))

df = pd.read\_csv("../input/heart.csv")

df.head()

df.target.value\_counts()

sns.countplot(x="target", data=df, palette="bwr")

plt.show()

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)))

sns.countplot(x='sex', data=df, palette="mako\_r")

plt.xlabel("Sex (0 = female, 1= male)")

plt.show()

df.groupby('target').mean()

pd.crosstab(df.age,df.target).plot(kind="bar",figsize=(20,6))

plt.title('Heart Disease Frequency for Ages')

plt.xlabel('Age')

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()

plt.ylabel('Frequency')

plt.savefig('heartDiseaseAndAges.png')

plt.show()

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

plt.scatter(x=df.age[df.target==0], y=df.thalach[(df.target==0)])

plt.legend(["Disease", "Not Disease"])

plt.xlabel("Age")

plt.ylabel("Maximum Heart Rate")

plt.show()

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()

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.ylabel('Frequency of Disease or Not')

plt.show()

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()

plt.figure(figsize=(24,12))

plt.suptitle("Confusion Matrixes",fontsize=24)

plt.subplots\_adjust(wspace = 0.4, hspace= 0.4)

plt.subplot(2,3,1)

plt.title("Logistic Regression Confusion Matrix")

sns.heatmap(cm\_lr,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.subplot(2,3,2)

plt.title("K Nearest Neighbors Confusion Matrix")

sns.heatmap(cm\_knn,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.subplot(2,3,3)

plt.title("Support Vector Machine Confusion Matrix")

sns.heatmap(cm\_svm,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.subplot(2,3,4)

plt.title("Naive Bayes Confusion Matrix"sns.heatmap(cm\_nb,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.subplot(2,3,5)

plt.title("Decision Tree Classifier Confusion Matrix")

sns.heatmap(cm\_dtc,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.subplot(2,3,6)

plt.title("Random Forest Confusion Matrix")

sns.heatmap(cm\_rf,annot=True,cmap="Blues",fmt="d",cbar=False, annot\_kws={"size": 24})

plt.show()

CONCLUSION

Based on the above review, it can be concluded that there is a huge scope for machine learning algorithms in predicting cardiovascular diseases or heart related diseases. Each of the above-mentioned algorithms have performed extremely well in some cases but poorly in some other cases. Alternating decision trees when used with PCA, have performed extremely well but decision trees have performed very poorly in some other cases which could be due to overfitting. Random Forest and Ensemble models have performed very well because they solve the problem of overfitting by employing multiple algorithms (multiple Decision Trees in case of Random Forest). Models based on Naïve Bayes classifier were computationally very fast and have also performed well.SVM performed extremely well for most of the cases. Systems based on machine learning algorithms and techniques have been very accurate in predicting the heart related diseases but still there is a lot scope of research to be done on how to handle high dimensional data and overfitting. A lot of research can also be done on the correct ensemble of algorithms to use for a particular type of data.

**URL:**