

## **Below steps are followed to build the heart attack prediction model**

### **Tools used**

#### **Sagemaker Notebook instance**

An *Amazon SageMaker notebook instance* is a machine learning (ML) compute instance running the Jupyter Notebook App. SageMaker manages creating the instance and related resources. Use Jupyter notebooks in your notebook instance to prepare and process data, write code to train models, deploy models to SageMaker hosting, and test or validate your models

#### **Amazon S3 bucket**

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Amazon S3 provides easy-to-use management features so you can organize your data and configure finely-tuned access controls to meet your specific business, organizational, and compliance requirements. Amazon S3 is designed for 99.999999999% (11 9's) of durability, and stores data for millions of applications for companies all around the world.

Store your data in Amazon S3 and secure it from unauthorized access with encryption features and access management tools. S3 is the only object storage service that allows you to block public access to all of your objects at the bucket or the account level with S3 Block Public Access.

#### **AWS Lambda function**

AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers, creating workload-aware cluster scaling logic, maintaining event integrations, or managing runtimes. With Lambda, you can run code for virtually any type of application or backend service - all with zero administration. Just upload your code as a ZIP file or container image, and Lambda automatically and precisely allocates compute execution power and runs your code based on the incoming request or event, for any scale of traffic.

**Application programming interface (API)** is an interface that defines interactions between multiple software applications or mixed hardware-software intermediaries.<sup>[1]</sup> It defines the kinds of calls or requests that can be made, how to make them, the data formats that should be used, the conventions to follow, etc. It can also provide extension mechanisms so that users can extend existing functionality in various ways and to varying degrees.

A **REST API** (also known as RESTful API) is an application programming interface (API or web API) that conforms to the constraints of REST architectural style and allows for interaction with RESTful web services.

### **Model Used**

#### **Naïve Bayes Classifier (Most Efficient)**

**Naive Bayes classifiers** are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naïve) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models,<sup>[1]</sup> but coupled with kernel density estimation, they can achieve higher accuracy levels.<sup>[2][3]</sup>

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. There is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable. For some types of probability models, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood. Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations.

An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.

**Other models are described later in the report.**

Neural Net

LogisticRegression

KNN

SVC

DecisionTree

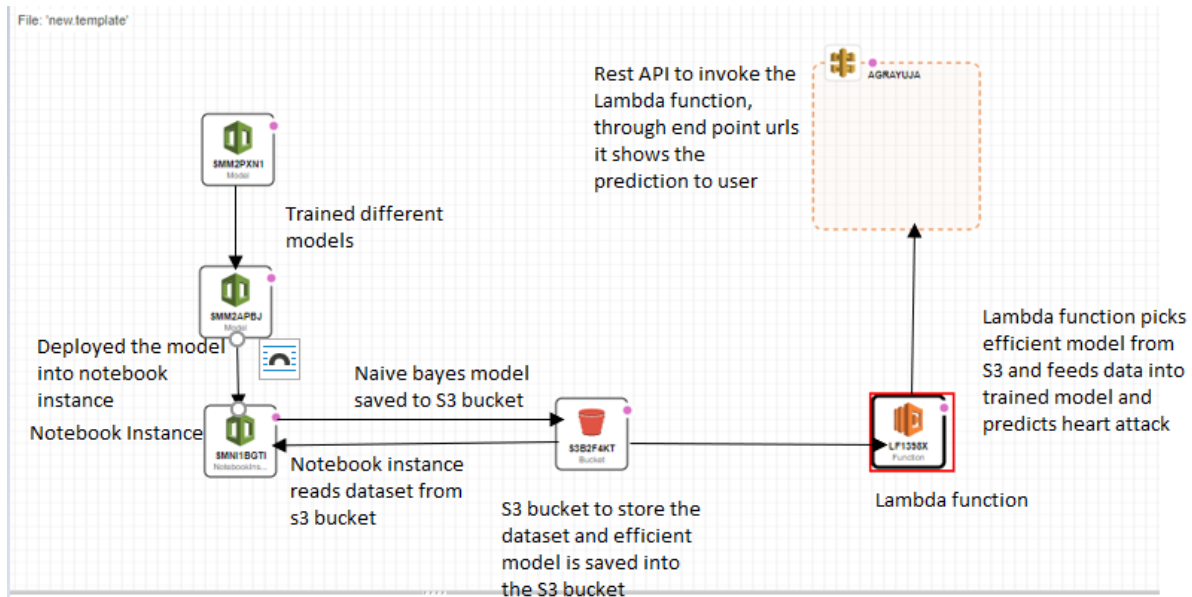
RandomForest

GradientBoost

AdaBoost

XGB

## Architecture



To keep the dataset safe and secure as it contains personal data, we have used S3 bucket.

## 1. Creation of S3 bucket

The screenshot shows the 'Create bucket' page in the AWS S3 Management Console. The browser address bar shows 's3.console.aws.amazon.com/s3/bucket/create?region=us-east-1'. The page has a dark header with the AWS logo, 'Services' dropdown, a search bar, and user information. The main content area is divided into sections. The first section contains a 'Bucket name' input field with the text 'team1-s3bucket-heartattack', a note about naming rules, an 'AWS Region' dropdown set to 'US East (N. Virginia) us-east-1', and a 'Copy settings from existing bucket - optional' section with a 'Choose bucket' button. The second section is titled 'Block Public Access settings for this bucket' and contains a checkbox 'Block all public access' which is checked. Below it, there is a sub-section 'Block public access to buckets and objects granted through new access control lists (ACLs)' with a checkbox that is also checked. The footer of the console shows 'Feedback', 'English (US)', and copyright information.

Bucket name  
team1-s3bucket-heartattack  
Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region  
US East (N. Virginia) us-east-1

Copy settings from existing bucket - optional  
Only the bucket settings in the following configuration are copied.  
[Choose bucket](#)

**Block Public Access settings for this bucket**  
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☒ **Block all public access**  
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☒ **Block public access to buckets and objects granted through new access control lists (ACLs)**  
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources.

The screenshot shows the 'Home' page of the AWS S3 Management Console. A green notification banner at the top states 'Successfully created bucket "team1-s3bucket-heartattack"' with a 'View details' button. The left sidebar shows the 'Amazon S3' navigation menu with options like 'Buckets', 'Access Points', 'Object Lambda Access Points', 'Batch Operations', 'Access analyzer for S3', 'Block Public Access settings for this account', 'Storage Lens', 'Dashboards', 'AWS Organizations settings', 'Feature spotlight', and 'AWS Marketplace for S3'. The main content area displays an 'Account snapshot' with metrics: Total storage (731.7 KB), Object count (118), and Avg. object size (6.2 KB). Below this is a 'Buckets (2)' section with a search bar and a table listing the buckets. The table has columns for Name, AWS Region, Access, and Creation date. The first bucket is 'cf-templates-13edpgj2uquad-us-east-1' and the second is 'team1-s3bucket-heartattack'. The footer of the console shows 'Feedback', 'English (US)', and copyright information.

**Successfully created bucket "team1-s3bucket-heartattack"**  
To upload files and folders, or to configure additional bucket settings choose [View details](#).

**Account snapshot**  
Last updated: May 3, 2021 by Storage Lens. Metrics are generated every 24 hours. [Learn more](#)

[View Storage Lens dashboard](#)

Total storage	Object count	Avg. object size	You can enable advanced metrics in the "default-account-dashboard" configuration.
731.7 KB	118	6.2 KB	

**Buckets (2)**  
Buckets are containers for data stored in S3. [Learn more](#)

	Name	AWS Region	Access	Creation date
<input type="radio"/>	cf-templates-13edpgj2uquad-us-east-1	US East (N. Virginia) us-east-1	Objects can be public	April 10, 2021, 19:14:11 (UTC-04:00)
<input type="radio"/>	team1-s3bucket-heartattack	US East (N. Virginia) us-east-1	Bucket and objects not public	May 4, 2021, 23:01:50 (UTC-04:00)

2. After the creation of S3 bucket, dataset which will be used for heart attack prediction is uploaded to the S3 bucket.

The screenshot shows the Amazon S3 Management Console interface. The left sidebar contains navigation options like Buckets, Access Points, and Storage Lens. The main content area displays the 'team1-s3bucket-heartattack' bucket. Under the 'Objects' tab, there are two objects listed in a table:

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	heart.csv	csv	May 4, 2021, 23:14:09 (UTC-04:00)	11.1 KB	Standard
<input type="checkbox"/>	heart_function-1.0.0.jar	jar	May 4, 2021, 23:14:11 (UTC-04:00)	8.1 MB	Standard

3. Once the creation is done, we have created Notebook instance provided by Amazon SageMaker studio which will allow us to train and deploy the models.

The screenshot shows the Amazon SageMaker console interface. The left sidebar contains navigation options like Amazon SageMaker Studio, Dashboard, Search, and Notebook. The main content area displays the 'Notebook instances' page. At the top, there is a 'Create notebook instance' button. Below it, there is a search bar and a table with columns: Name, Instance, Creation time, Status, and Actions. The table is currently empty, and a message states: 'There are currently no resources.'

#### 4. We have used ml.t2.medium instance type which is used for the creation of notebook instance type

Amazon SageMaker

console.aws.amazon.com/sagemaker/home?region=us-east-1#/notebook-instances/create

Services Search for services, features, marketplace products, and do [Alt+S]

Amazon SageMaker > Notebook instances > Create notebook instance

### Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

#### Notebook instance settings

Notebook instance name  
team1-notebookinstance-heartattack  
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type  
ml.t2.medium

Elastic Inference [Learn more](#)  
none

► Additional configuration

#### Permissions and encryption

Feedback English (US)

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

#### 5. We have created a new IAM role and provided the S3 bucket name which we have created earlier so that the notebook instance can access the data from S3 bucket.

Amazon SageMaker

console.aws.amazon.com/sagemaker/home?region=us-east-1#/notebook-instances/create

Services Search for services, features, marketplace products, and do [Alt+S]

Amazon SageMaker > Notebook instances > Create notebook instance

### Create an IAM role

Passing an IAM role gives Amazon SageMaker permission to perform actions in other AWS services on your behalf. Creating a role here will grant permissions described by the [AmazonSageMakerFullAccess](#) IAM policy to the role you create.

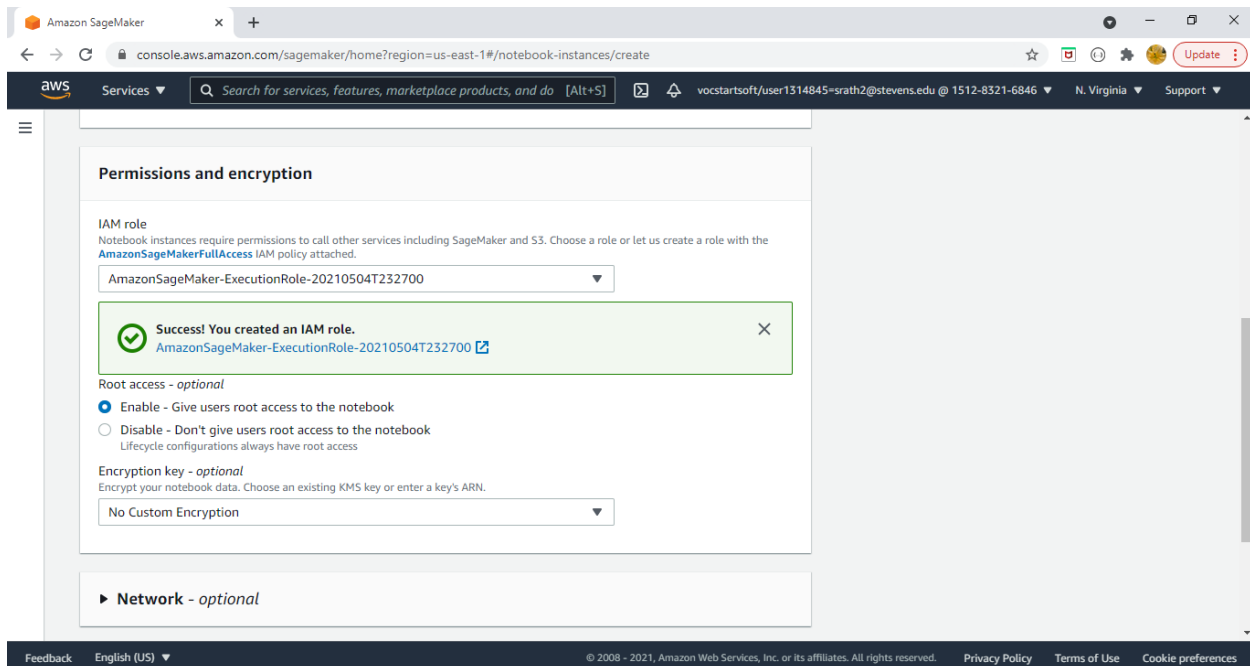
The IAM role you create will provide access to:

- ✓ S3 buckets you specify - optional
  - ☒ Any S3 bucket  
Allow users that have access to your notebook instance access to any bucket and its contents in your account.
  - ☐ Specific S3 buckets  
team1-s3bucket-heartattack  
Comma delimited. ARNs, "\*" and "/" are not supported.
  - ☐ None
- ✓ Any S3 bucket with "sagemaker" in the name
- ✓ Any S3 object with "sagemaker" in the name
- ✓ Any S3 object with the tag "sagemaker" and value "true" [See Object tagging](#)
- ✓ S3 bucket with a Bucket Policy allowing access to SageMaker [See S3 bucket policies](#)

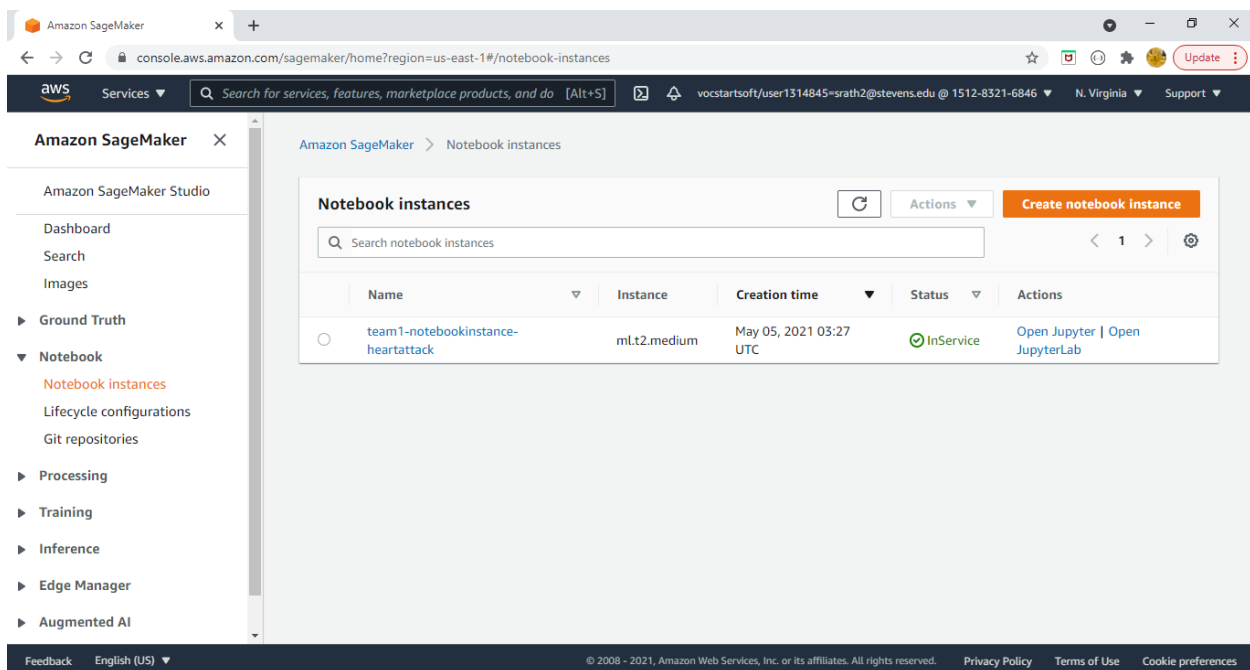
Cancel Create role

Feedback English (US)

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences



## 6. After the successful creation of IAM role, notebook instance is created



7. After the successful creation of notebook instance, for deploying the models we will go to Jupyter notebook. We can use Jupyter notebook in our notebook instance to prepare and process data, write code to train models, deploy models to SageMaker hosting, and test or validate your models.
8. We have uploaded our Heart Prediction.ipynb file to the Jupyter instance which includes the programming part of our all the different classifiers which we have used in our project.

9. Once we have clicked on Heart Prediction.ipynb, it showed the programming part which we have implemented with different classifiers to predict the heart attack.



10. We have imported all the required libraries which were required for the project.

For Linear algebra numpy is imported

A **numpy array** is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the **array**; the shape of an **array** is a tuple of integers giving the size of the **array** along each dimension

For data processing pandas is imported

**pandas** is a fast, powerful, flexible and easy to use open source **data analysis** and manipulation tool, built on top of the Python programming language.

For building the model

```
from sklearn.metrics import accuracy_score
```

Accuracy classification score.

In multilabel classification, this function computes subset accuracy: the set of labels predicted for a sample must *exactly* match the corresponding set of labels in y\_true.



```
from sklearn.metrics import confusion_matrix
```

By definition a confusion matrix  $C$  is such that  $C_{i,j}$  is equal to the number of observations known to be in group  $i$  but predicted to be in group  $j$ .

```
from sklearn.preprocessing import StandardScaler
```

Standardize features by removing the mean and scaling to unit variance

The standard score of a sample  $x$  is calculated as:

$$z = (x - u) / s$$

where  $u$  is the mean of the training samples or zero if `with_mean=False`, and  $s$  is the standard deviation of the training samples or one if `with_std=False`.

```
from sklearn.linear_model import LogisticRegression
```

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi\_class' option is set to 'ovr', and uses the cross-entropy loss if the 'multi\_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag', 'saga' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag', 'saga' and 'lbfgs' solvers. **Note that regularization is applied by default.** It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation, or no regularization. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty. The Elastic-Net regularization is only supported by the 'saga' solver.

```
from sklearn.neighbors import KNeighborsClassifier
```

[`KNeighborsClassifier`](#) implements learning based on the  $k$  nearest neighbors of each query point, where  $k$  is an integer value specified by the user

The k-neighbors classification in [KNeighborsClassifier](#) is the most commonly used technique. The optimal choice of the value k is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distinct.

```
from sklearn.naive_bayes import GaussianNB
```

[GaussianNB](#) implements the Gaussian Naive Bayes algorithm for classification. The likelihood of the features is assumed to be Gaussian:

$$P(x_i | y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

$$P(x_i | y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

The parameters  $\sigma_y$  and  $\mu_y$  are estimated using maximum likelihood.

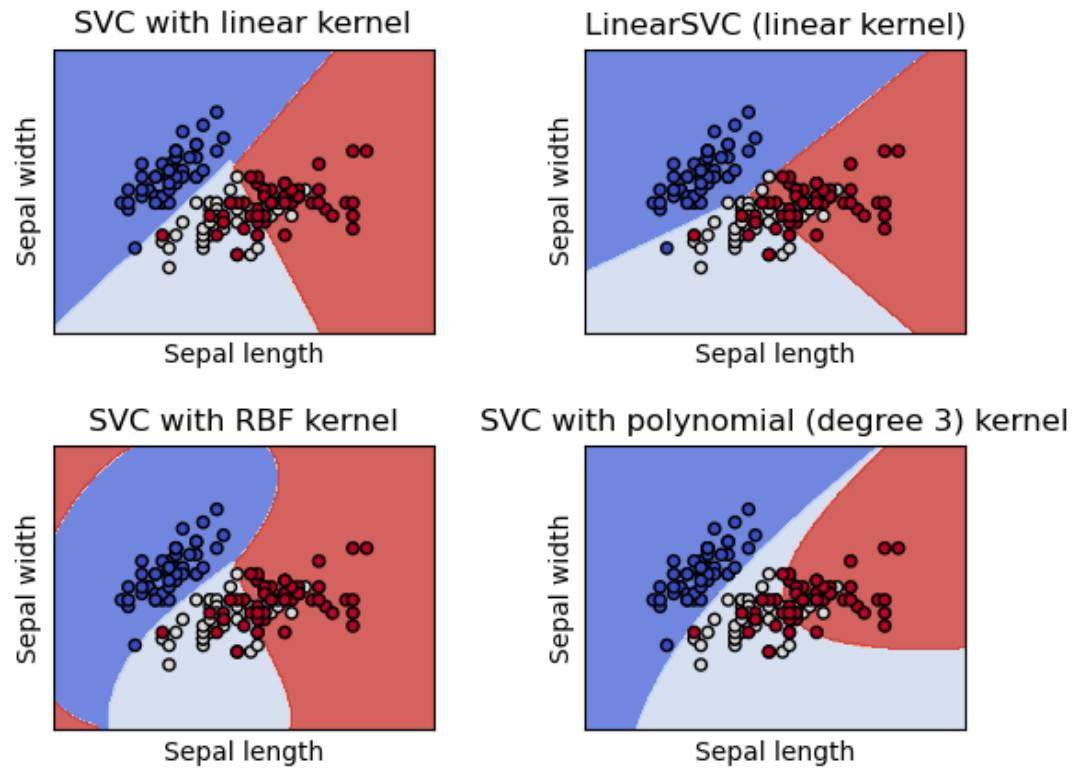
```
from sklearn.neural_network import MLPClassifier
```

This model optimizes the log-loss function using LBFGS or stochastic gradient descent.

```
from sklearn.svm import SVC
```

C-Support Vector Classification.

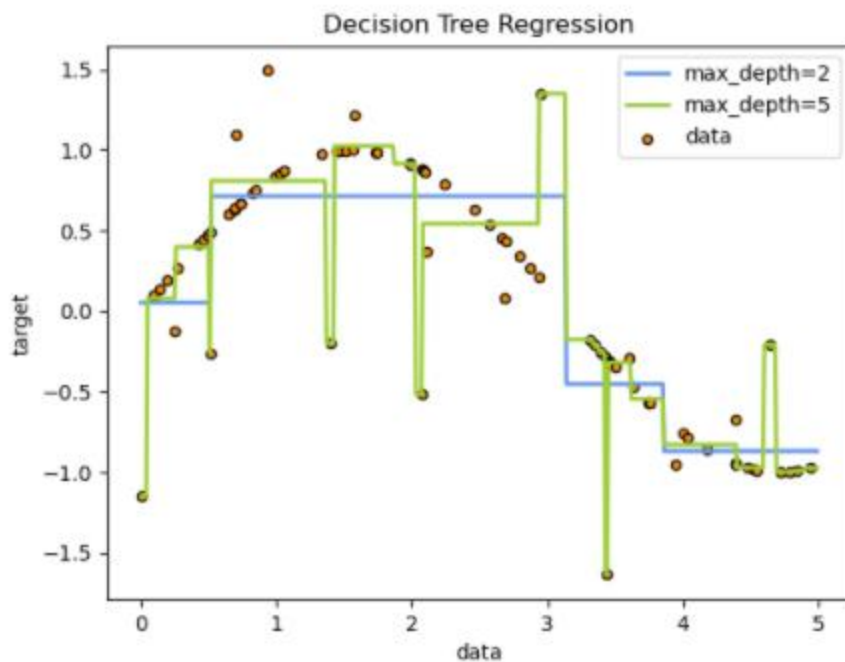
The implementation is based on libsvm. The fit time scales at least quadratically with the number of samples and may be impractical beyond tens of thousands of samples. For large datasets consider using [LinearSVC](#) or [SGDClassifier](#) instead, possibly after a [Nystroem](#) transformer.



```
from sklearn.tree import DecisionTreeClassifier
```

**Decision Trees (DTs)** are a non-parametric supervised learning method used for [classification](#) and [regression](#). The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.



```
from sklearn.ensemble import RandomForestClassifier
```

A random forest classifier.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the `max_samples` parameter if `bootstrap=True` (default), otherwise the whole dataset is used to build each tree.

```
from sklearn.ensemble import GradientBoostingClassifier
```

Gradient Boosting for classification.

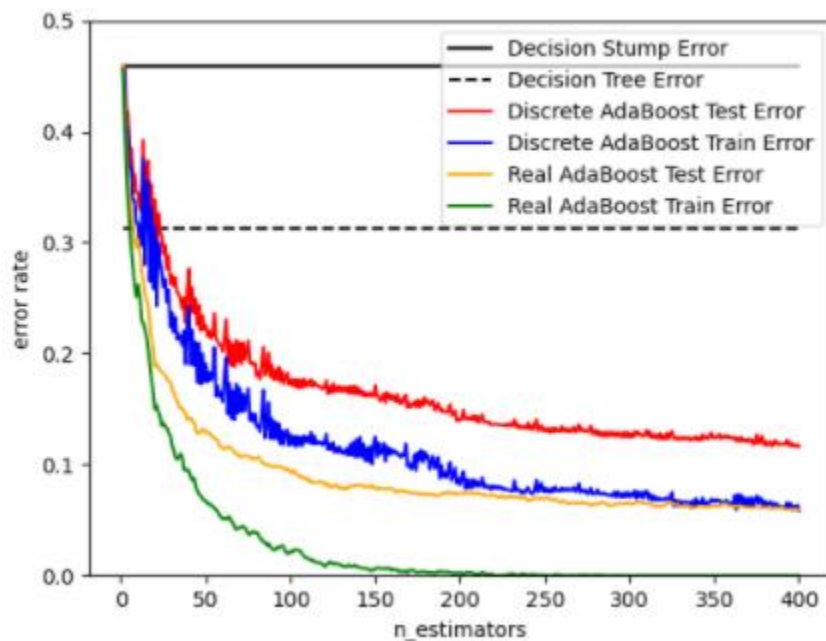
GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage `n_classes_` regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

[Gradient Tree Boosting](#) or Gradient Boosted Decision Trees (GBDT) is a generalization of boosting to arbitrary differentiable loss functions. GBDT is an accurate and effective off-the-shelf procedure that can be used for both regression and classification problems in a variety of areas including Web search ranking and ecology.

```
from sklearn.ensemble import AdaBoostClassifier
```

An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

The core principle of AdaBoost is to fit a sequence of weak learners (i.e., models that are only slightly better than random guessing, such as small decision trees) on repeatedly modified versions of the data. The predictions from all of them are then combined through a weighted majority vote (or sum) to produce the final prediction



```
import xgboost as xgb
```

**XGBoost** is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. **XGBoost** is an implementation of gradient boosted decision trees designed for speed and performance.

```
In [504]: import numpy as np #linear algebra
import pandas as pd #data processing

import matplotlib.pyplot as plt #data visualization
import seaborn as sns #data visualization

import warnings
warnings.filterwarnings("ignore") #to ignore the warnings

#for model building
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
import xgboost as xgb

In [505]: #Reading the csv file heart.csv in variable
```

11. We have specified the S3 bucket name which we have created to store the dataset , to read from it.

12. After reading the data from the S3 bucket we are looking at the first five rows and last five rows.

```
In [505]: #Reading the csv file heart.csv in variable
heartdata='s3://team1-s3bucket-heartattack/heart.csv'
df=pd.read_csv(heartdata)

In [506]: # Looking at the first 5 rows of our data
df.head()

Out[506]:
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
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

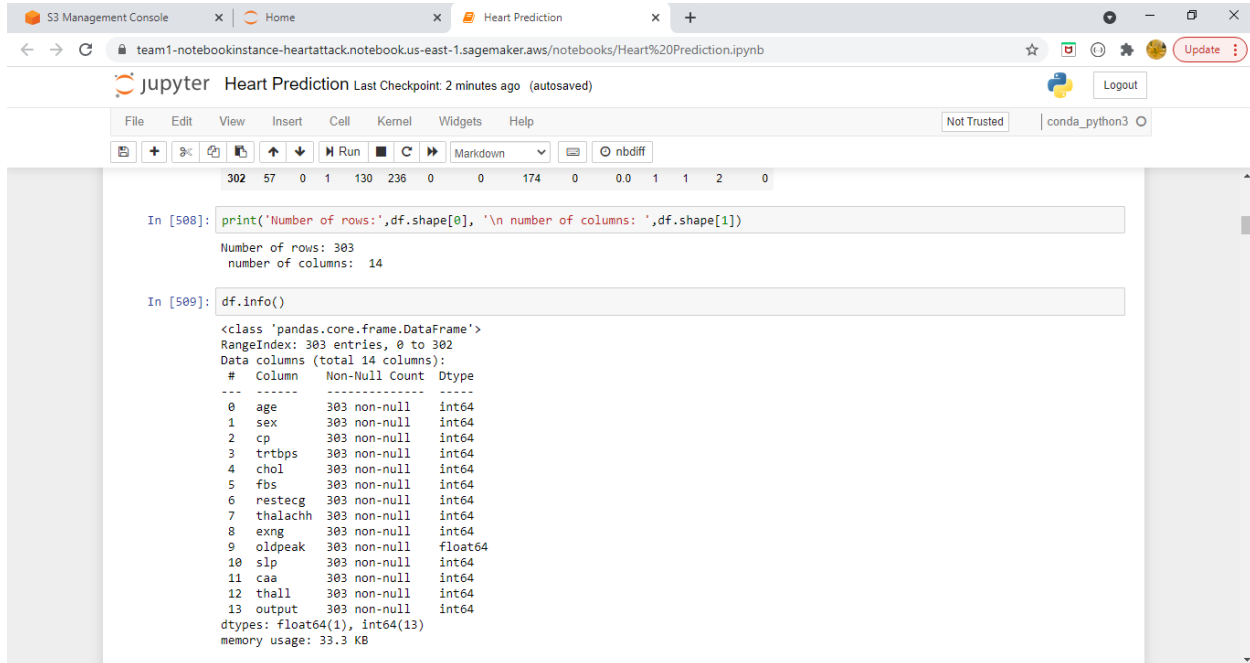
```
In [507]: df.tail()

Out[507]:
```

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

13. Then we are checking the number of rows in our dataset and the number of columns.

14. We are also checking with the help of pandas library the datatype and non-null column for all the fields.



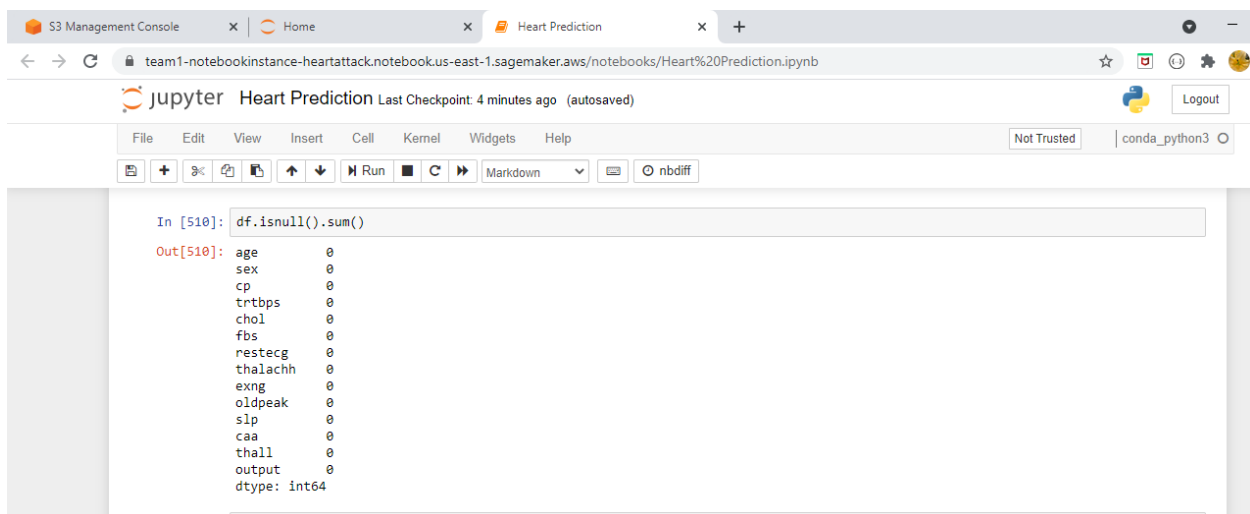
```
In [508]: print('Number of rows:',df.shape[0], '\n number of columns: ',df.shape[1])

Number of rows: 303
number of columns: 14

In [509]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
#   Column      Non-Null Count  Dtype
---  ---
0    age        303 non-null    int64
1    sex        303 non-null    int64
2    cp         303 non-null    int64
3    trtbps     303 non-null    int64
4    chol       303 non-null    int64
5    fbs        303 non-null    int64
6    restecg    303 non-null    int64
7    thalachh   303 non-null    int64
8    exng       303 non-null    int64
9    oldpeak    303 non-null    float64
10   slp        303 non-null    int64
11   caa        303 non-null    int64
12   thall      303 non-null    int64
13   output     303 non-null    int64
dtypes: float64(1), int64(13)
memory usage: 33.3 KB
```

15. Checking the columns if there are any fields which are null.



```
In [510]: df.isnull().sum()

Out[510]: age      0
sex      0
cp       0
trtbps   0
chol     0
fbs      0
restecg  0
thalachh 0
exng     0
oldpeak  0
slp      0
caa      0
thall    0
output   0
dtype: int64
```

16. Checking the data for duplicate values, if there are any then removing it from the dataset.

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

Jupyter Heart Prediction Last Checkpoint: 4 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda\_python3

dtype: int64

```
In [511]: df.duplicated().sum()
Out[511]: 1

In [512]: df.drop_duplicates(inplace=True)
           print('Number of rows are',df.shape[0], 'and number of columns are ',df.shape[1])
Number of rows are 302 and number of columns are 14
```

**17. We are calculating the total count, mean, standard deviation, minimum, and maximum for all the columns.**

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

Jupyter Heart Prediction Last Checkpoint: 5 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted

Number of rows are 302 and number of columns are 14

```
In [513]: df.describe().T
Out[513]:
```

	count	mean	std	min	25%	50%	75%	max
age	302.0	54.420530	9.047970	29.0	48.00	55.5	61.00	77.0
sex	302.0	0.682119	0.466426	0.0	0.00	1.0	1.00	1.0
cp	302.0	0.963576	1.032044	0.0	0.00	1.0	2.00	3.0
trtbps	302.0	131.602649	17.563394	94.0	120.00	130.0	140.00	200.0
chol	302.0	246.500000	51.753489	126.0	211.00	240.5	274.75	564.0
fbs	302.0	0.149007	0.356686	0.0	0.00	0.0	0.00	1.0
restecg	302.0	0.526490	0.526027	0.0	0.00	1.0	1.00	2.0
thalachh	302.0	149.569536	22.903527	71.0	133.25	152.5	166.00	202.0
exng	302.0	0.327815	0.470196	0.0	0.00	0.0	1.00	1.0
oldpeak	302.0	1.043046	1.161452	0.0	0.00	0.8	1.60	6.2
slp	302.0	1.397351	0.616274	0.0	1.00	1.0	2.00	2.0
caa	302.0	0.718543	1.006748	0.0	0.00	0.0	1.00	4.0
thall	302.0	2.314570	0.613026	0.0	2.00	2.0	3.00	3.0
output	302.0	0.543046	0.498970	0.0	0.00	1.0	1.00	1.0

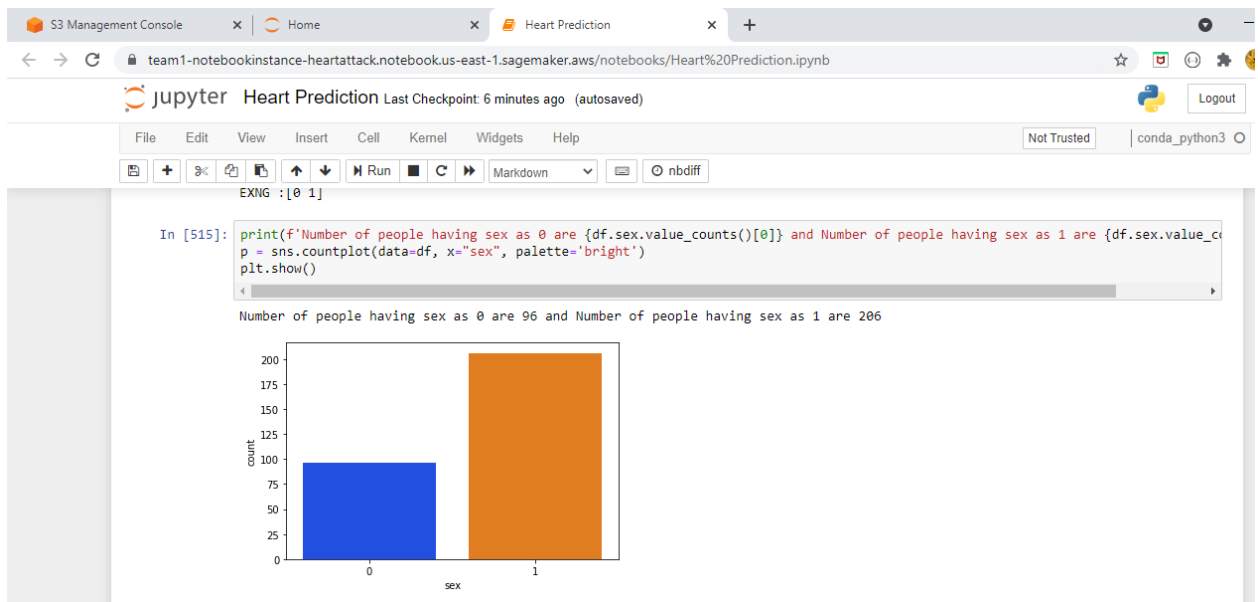


```
S3 Management Console x Home x Heart Prediction x +
team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb
Jupyter Heart Prediction Last Checkpoint: 6 minutes ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help Not Trusted | conda_python3 C
In [514]: #This is to look at what all unique values have . Just trying to use python
list_col=['sex','chol','trtbps','cp','thall','exng']

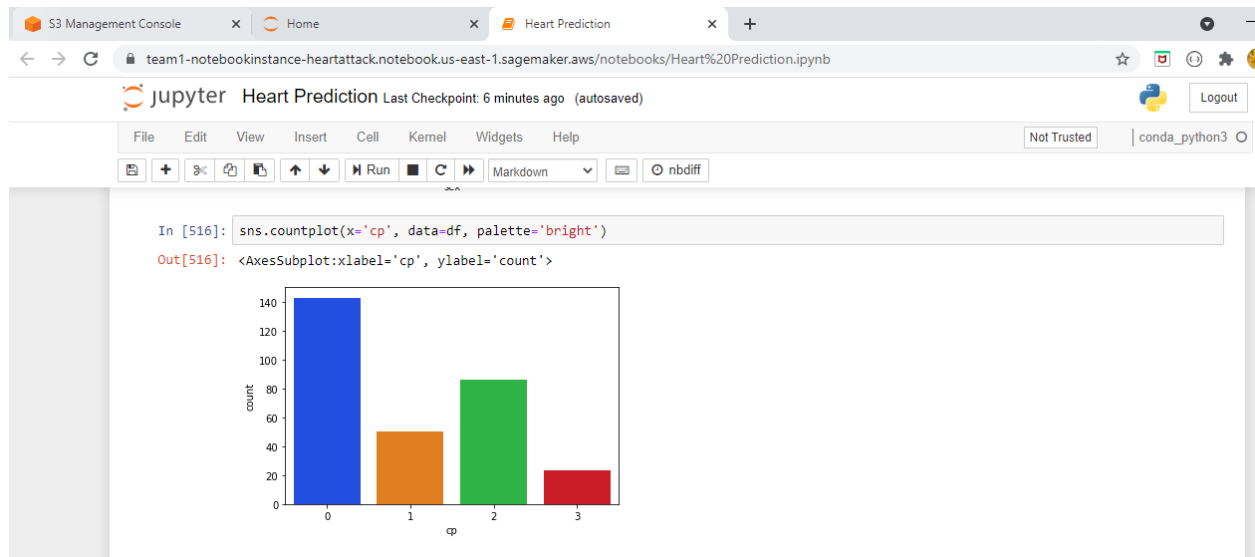
for col in list_col:
    print('{} :{}'.format(col.upper(),df[col].unique()))

SEX :[1 0]
CHOL :[233 250 204 236 354 192 294 263 199 168 239 275 266 211 283 219 340 226
247 234 243 302 212 175 417 197 198 177 273 213 304 232 269 360 308 245
208 264 321 325 235 257 216 256 231 141 252 201 222 260 182 303 265 309
186 203 183 220 209 258 227 261 221 205 240 318 298 564 277 214 248 255
207 223 288 160 394 315 246 244 270 195 196 254 126 313 262 215 193 271
268 267 210 295 306 178 242 180 228 149 278 253 342 157 286 229 284 224
206 167 230 335 276 353 225 330 290 172 305 188 282 185 326 274 164 307
249 341 407 217 174 281 289 322 299 300 293 184 409 259 200 327 237 218
319 166 311 169 187 176 241 131]
TRTBPS :[145 130 120 140 172 150 110 135 160 105 125 142 155 104 138 128 108 134
122 115 118 100 124 94 112 102 152 101 132 148 178 129 180 136 126 106
156 170 146 117 200 165 174 192 144 123 154 114 164]
CP :[3 2 1 0]
THALL :[1 2 3 0]
EXNG :[0 1]
```

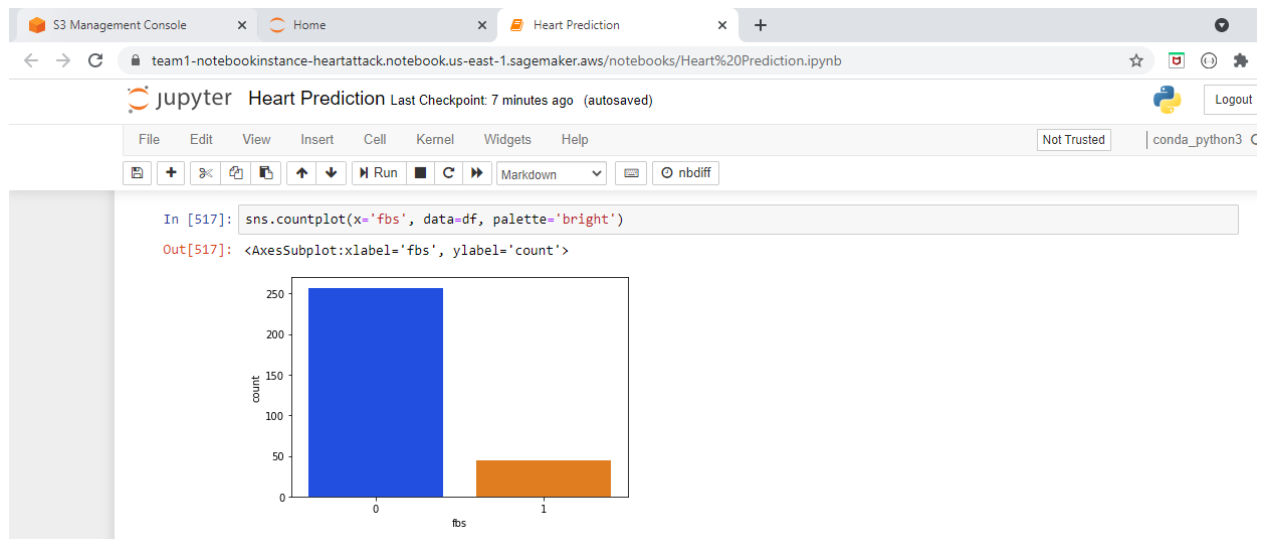
18. Plotting the graph for the sex field which will show the ratio of male and female.



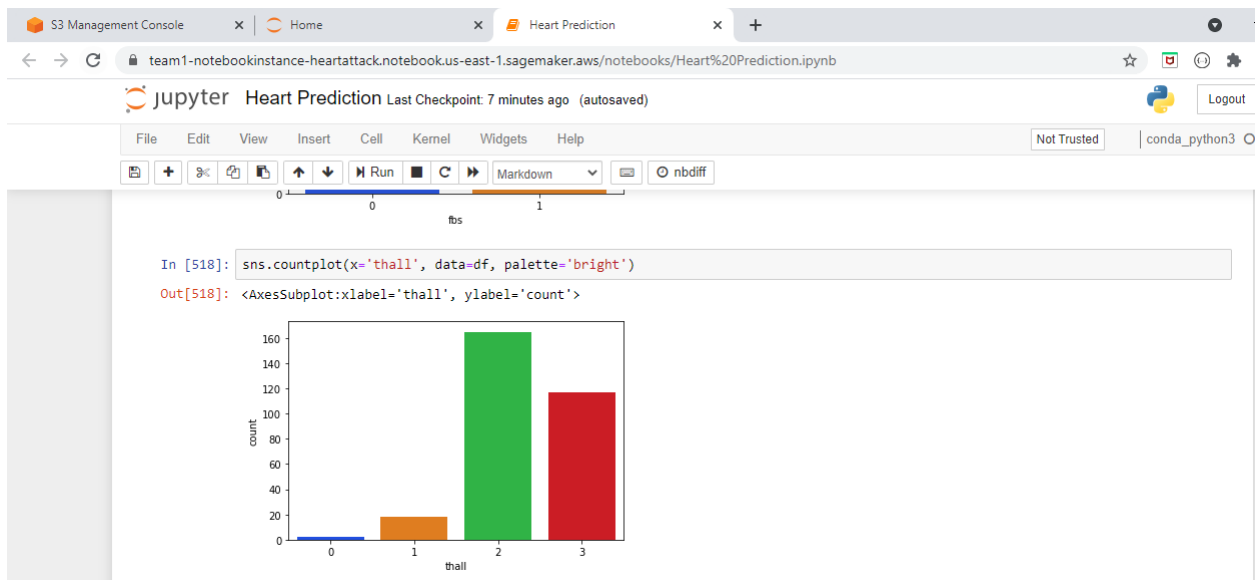
19. Plotting the graph for chest pain which shows the number of individuals are having the chest pain according to the severity level.



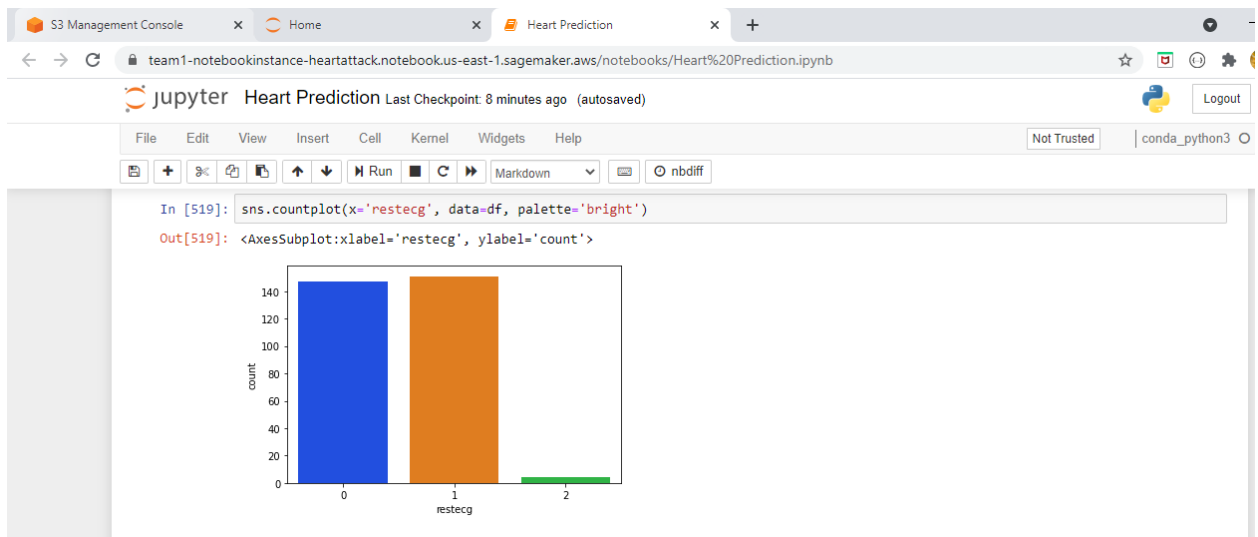
20. Plotting the graph for the fasting blood sugar which shows how many individuals have the sugar level greater than 120.



21. Plotting the graph for the maximum heart rate achieved field which will show how many individuals have achieved the maximum heart rate.

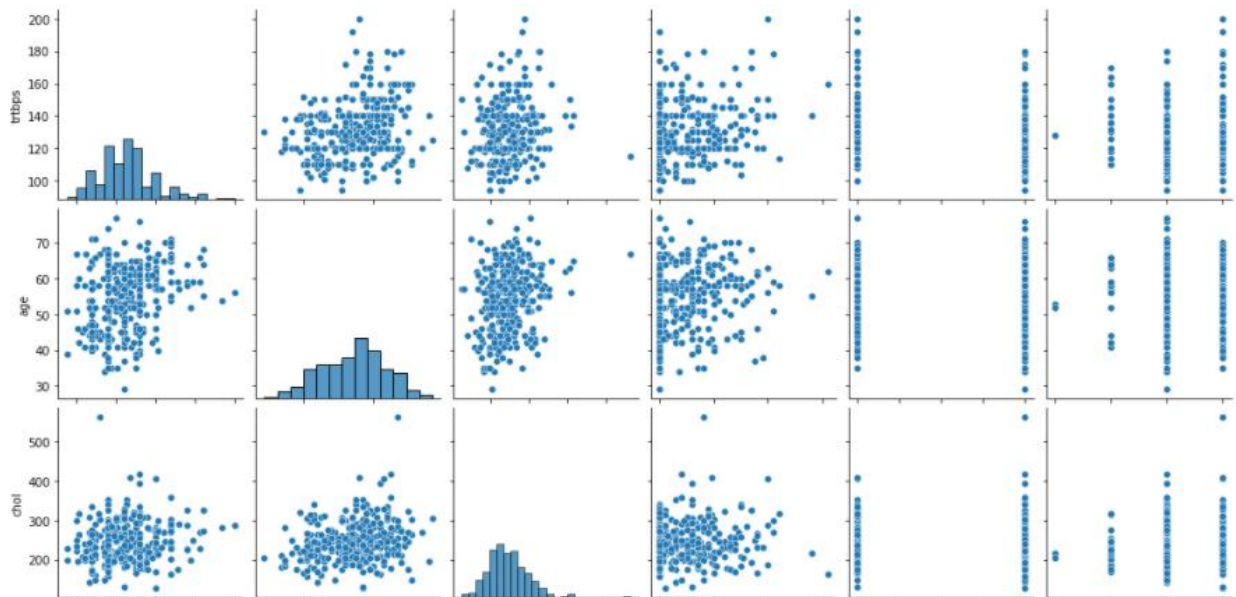
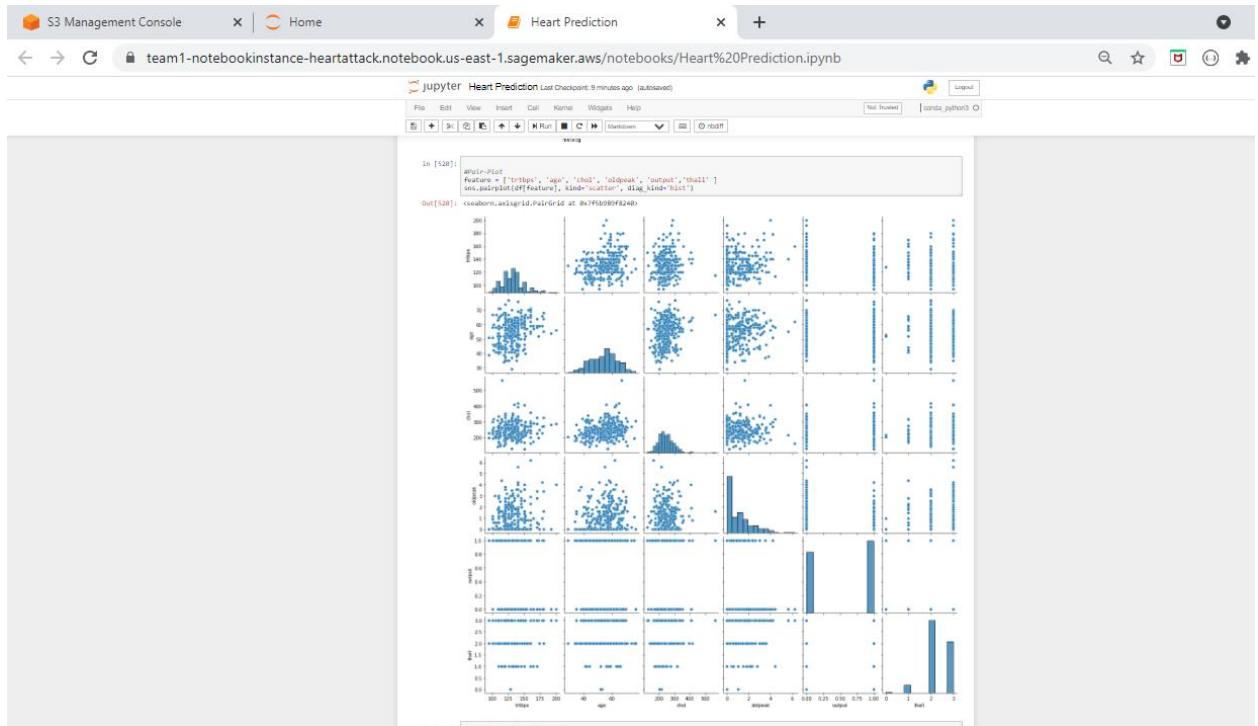


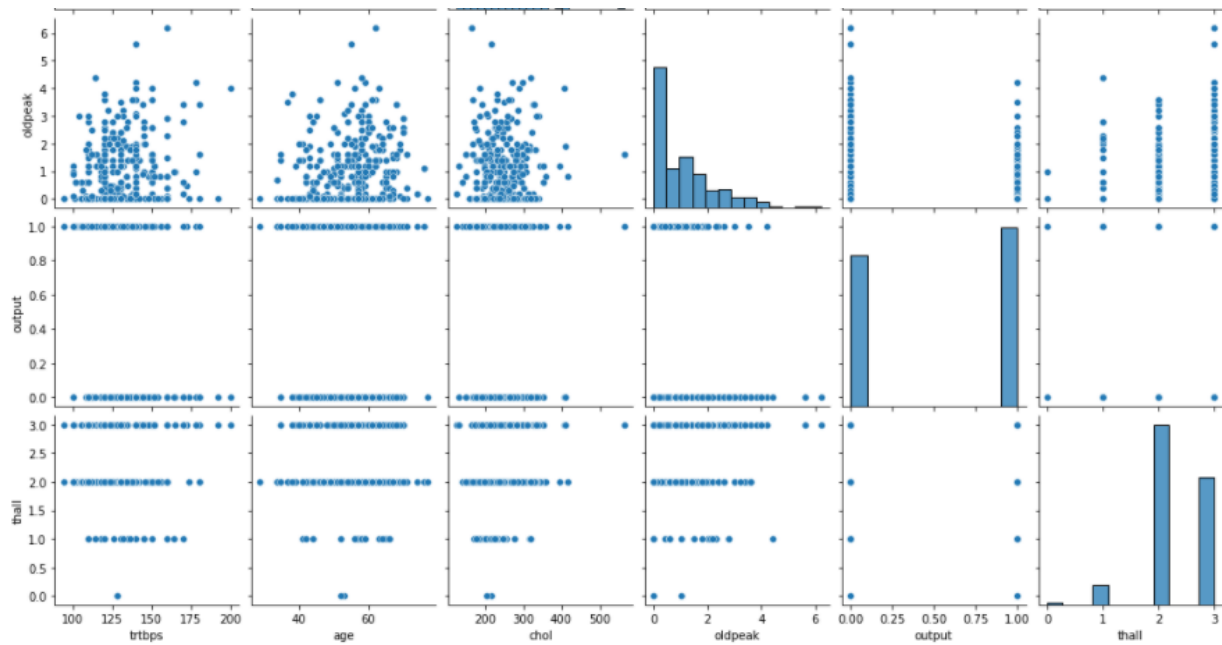
## 22. Plotting the graph for the resting electrocardiographic results.



## 23. Plotting the pair plots which will help in visualizing the data more clearly.

**Pair Plots** are a really simple (one-line-of-code simple!) way to visualize relationships between each variable. It produces a matrix of relationships between each variable in your data for an instant examination of our data. It can also be a great jumping off point for determining types of regression analysis to use

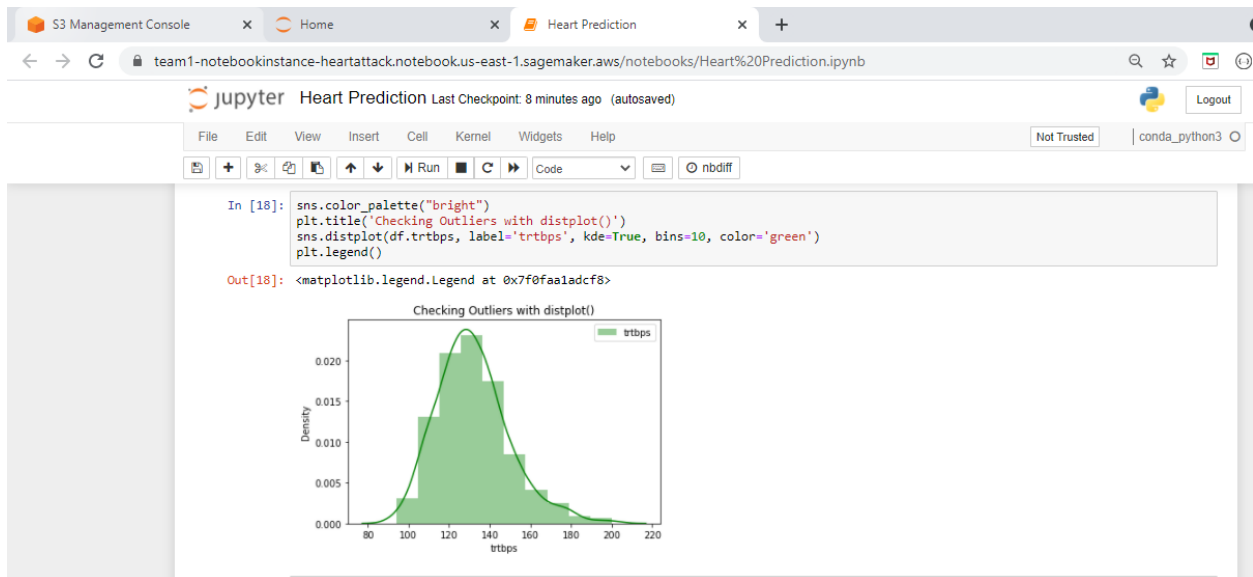




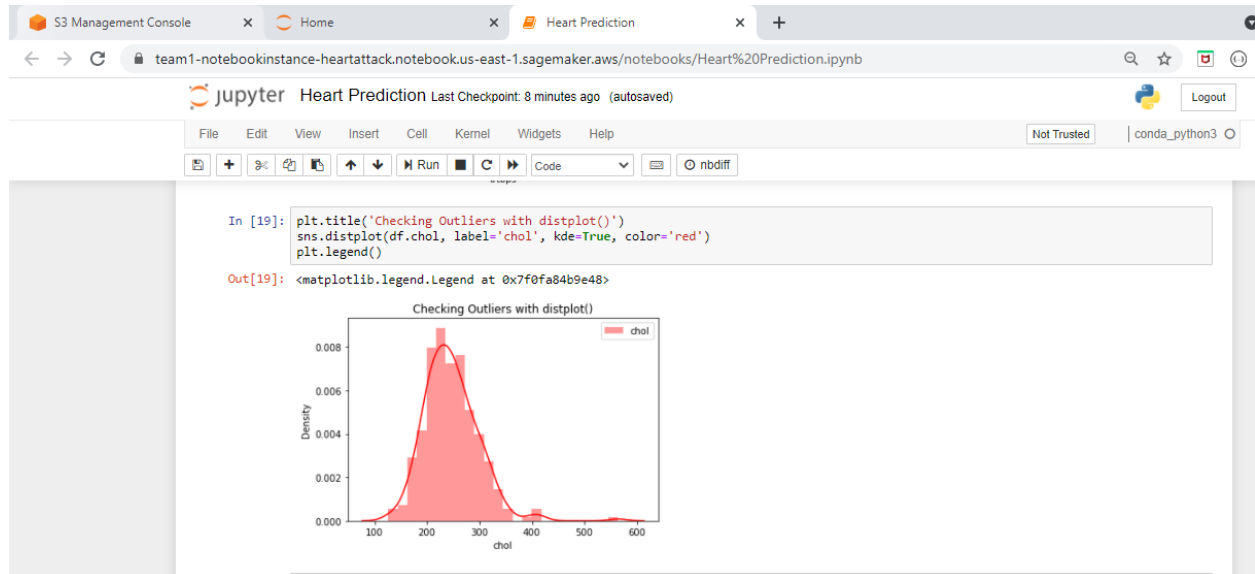
## 24. Checking the outliers for resting blood pressure using distplot.

Seaborn distplot lets you show a histogram with a line on it. This can be shown in all kinds of variations. We use seaborn in combination with matplotlib, the Python plotting module.

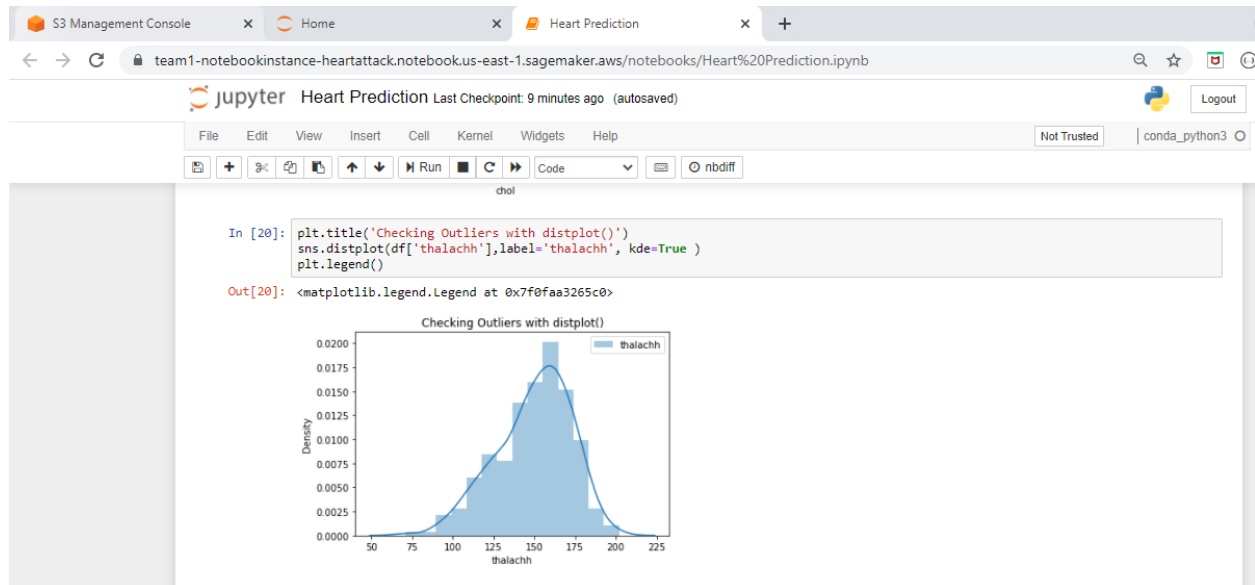
A distplot plots a univariate distribution of observations. The distplot() function combines the matplotlib hist function with the seaborn kdeplot() and rugplot() functions.



## 25. Checking the outliers for cholesterol using distplot.



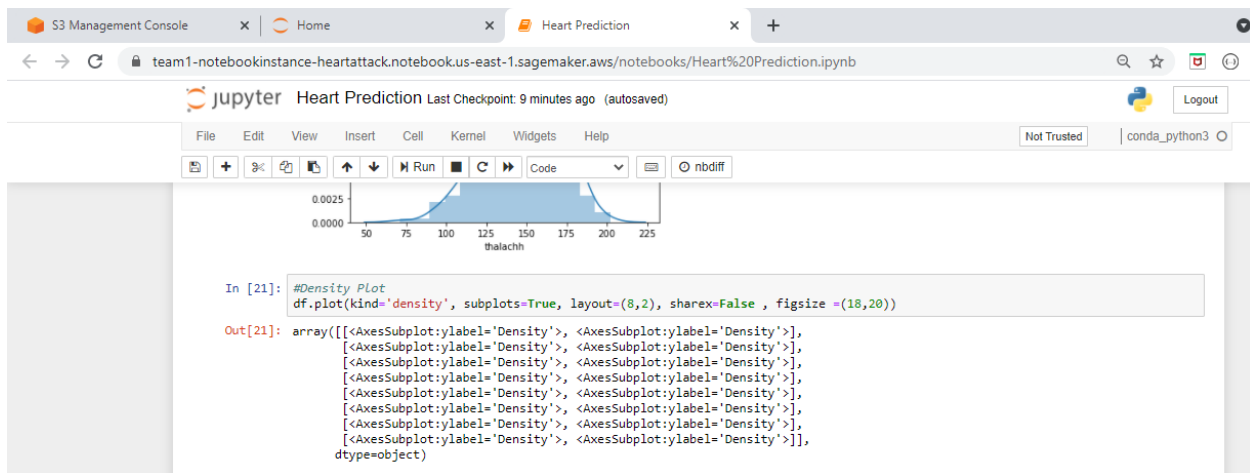
## 26. Checking the outliers for maximum heart rate using distplot.



## 27. Plotting the density plot for all the columns, cholesterol, heart rate, fasting blood sugar, age, sex etc.

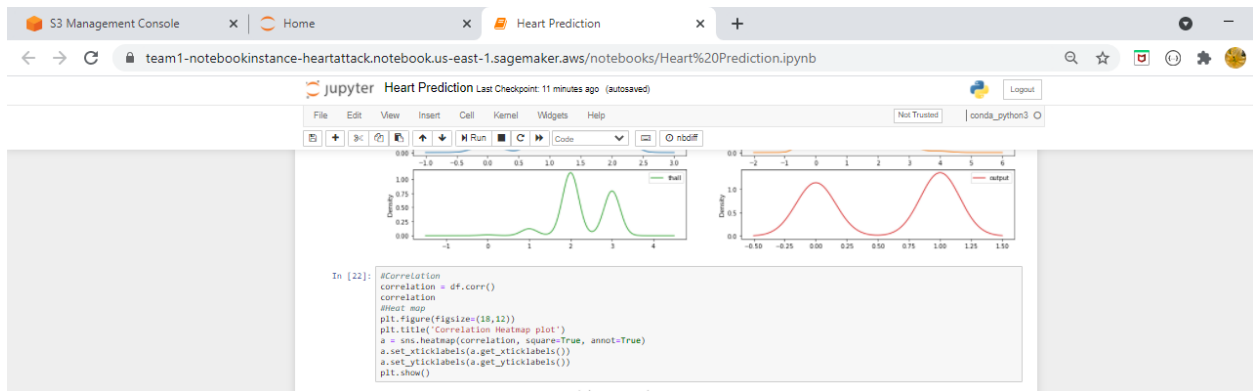
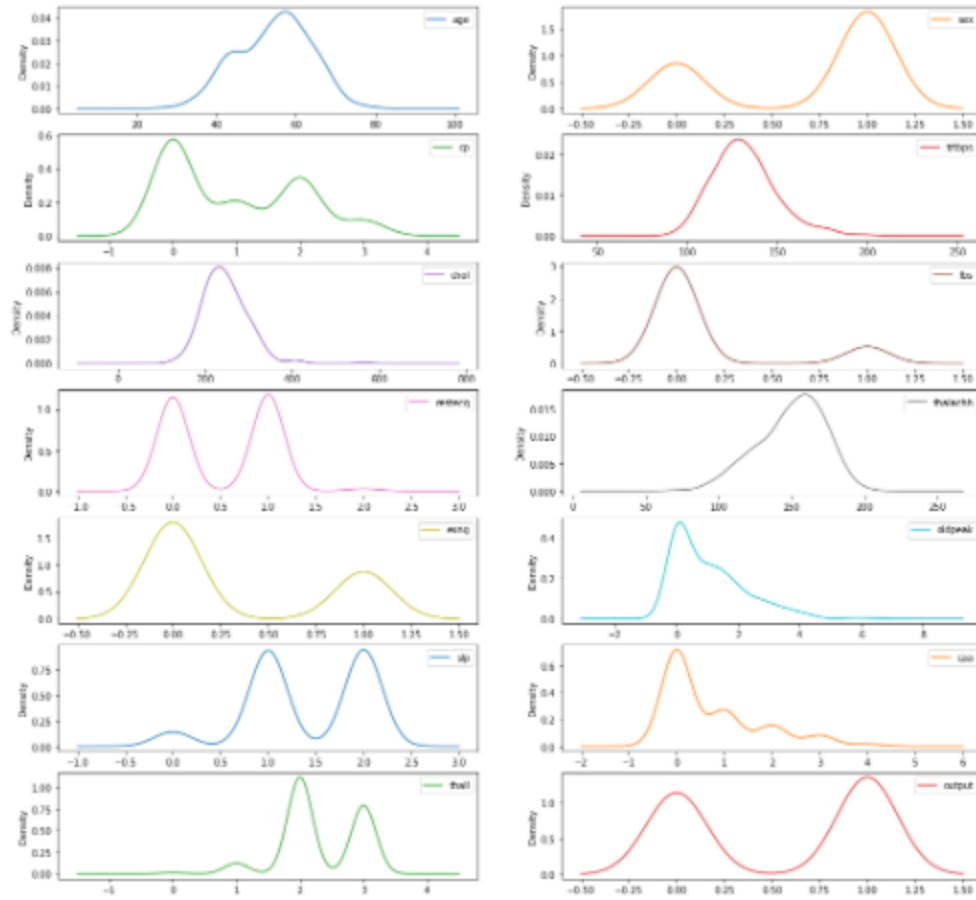
A Density Plot visualises the distribution of data over a continuous interval or time period. This chart is a variation of a Histogram that uses kernel smoothing to plot values, allowing for smoother distributions by smoothing out the noise. The peaks of a Density Plot help display where values are concentrated over the interval.

An advantage Density Plots have over Histograms is that they're better at determining the distribution shape because they're not affected by the number of bins used (each bar used in a typical histogram). A Histogram comprising of only 4 bins wouldn't produce a distinguishable enough shape of distribution as a 20-bin Histogram would. However, with Density Plots, this isn't an issue.



```
In [21]: #Density Plot
df.plot(kind='density', subplots=True, layout=(8,2), sharex=False, figsize=(18,28))
```

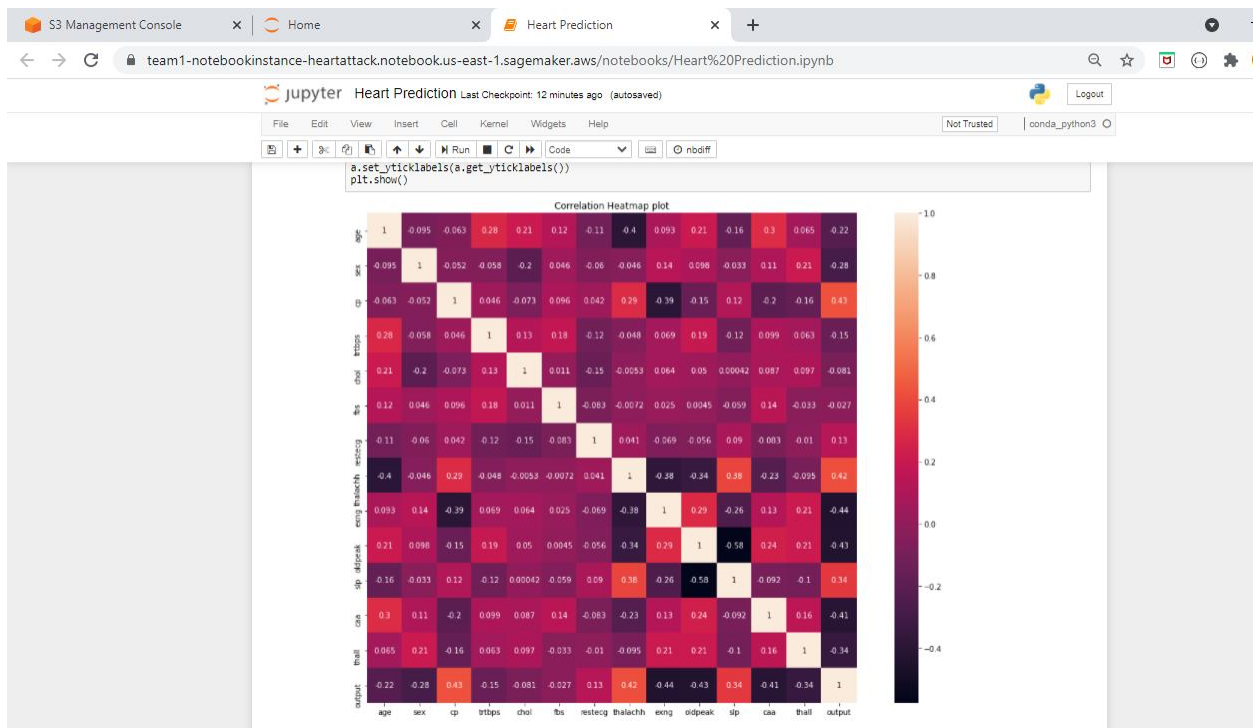
```
Out[21]: array([[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>],
[<AxesSubplot:ylabel='Density'>, <AxesSubplot:ylabel='Density'>]],
dtype=object)
```





28. Plotting the correlation heat map plot for all the columns, cholesterol, heart rate, fasting blood sugar, age, sex etc.

A heatmap (aka heat map) depicts values for a main variable of interest across two axis variables as a grid of colored squares. The axis variables are divided into ranges like a [bar chart](#) or [histogram](#), and each cell's color indicates the value of the main variable in the corresponding cell range.



29. Now we have scaled the data so that all the values will fall within the same range which will help in analyzing the data easily and in calculating the efficiency.

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

jupyter Heart Prediction Last Checkpoint: 12 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda\_python3

In [23]: `X = df.drop('output', axis = 1)`  
`y = df['output']`

In [24]: `df.reset_index(drop=True, inplace=True)`

In [25]: `columns_to_scale = df.iloc[:, [0,3,4,7,9,]]`  
`columns_to_scale`

Out[25]:

	age	trtbps	chol	thalachh	oldpeak
0	63	145	233	150	2.3
1	37	130	250	187	3.5
2	41	130	204	172	1.4
3	56	120	236	178	0.8
4	57	120	354	163	0.6
...	...	...	...	...	...
297	57	140	241	123	0.2
298	45	110	264	132	1.2
299	68	144	193	141	3.4
300	57	130	131	115	1.2
301	57	130	236	174	0.0

302 rows x 5 columns

In [26]: `# Splitting the data`  
`from sklearn.model_selection import train_test_split`  
`X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)`

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

jupyter Heart Prediction Last Checkpoint: 13 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda\_python3

302 rows x 5 columns

In [26]: `# Splitting the data`  
`from sklearn.model_selection import train_test_split`  
`X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)`  
`ss = StandardScaler()`  
`scaled_values = ss.fit_transform(columns_to_scale)`  
`scaled_values = pd.DataFrame(scaled_values, columns=columns_to_scale.columns)`  
`scaled_values`

Out[26]:

	age	trtbps	chol	thalachh	oldpeak
0	0.949794	0.764066	-0.261285	0.018826	1.084022
1	-1.928548	-0.091401	0.067741	1.636979	2.118926
2	-1.485726	-0.091401	-0.822564	0.980971	0.307844
3	0.174856	-0.661712	-0.203222	1.243374	-0.209608
4	0.285561	-0.661712	2.080602	0.587366	-0.382092
...	...	...	...	...	...
297	0.285561	0.478910	-0.106449	-1.161988	-0.727060
298	-1.042904	-1.232023	0.338703	-0.768384	0.135360
299	1.503322	0.707035	-1.035462	-0.374779	2.032684
300	0.285561	-0.091401	-2.235438	-1.511859	0.135360
301	0.285561	-0.091401	-0.203222	1.068439	-0.899544

302 rows x 5 columns

In [27]: `scaled_df = pd.concat([scaled_values,df.iloc[:, [1,2,5,6,8,10,11,12,13]]],axis=1)`  
`scaled_df`

```

In [27]: scaled_df = pd.concat([scaled_values, df.iloc[:, [1, 2, 5, 6, 8, 10, 11, 12, 13]]], axis=1)
scaled_df

Out[27]:
   age  tmbps  chol  thalach  oldpeak  sex  cp  fbs  restecg  exng  slp  caa  thall  output
0  0.949794  0.764066 -0.261285  0.018826  1.084022  1  3  1  0  0  0  0  1  1
1 -1.928548 -0.091401  0.067741  1.636979  2.118926  1  2  0  1  0  0  0  2  1
2 -1.485726 -0.091401 -0.822564  0.980971  0.307844  0  1  0  0  0  2  0  2  1
3  0.174856 -0.661712 -0.203222  1.243374 -0.209608  1  1  0  1  0  2  0  2  1
4  0.285561 -0.661712  2.080602  0.587366 -0.382092  0  0  0  1  1  2  0  2  1
...
297 0.285561  0.478910 -0.106449 -1.161988 -0.727060  0  0  0  1  1  1  0  3  0
298 -1.042904 -1.232023  0.338703 -0.768384  0.135360  1  3  0  1  0  1  0  3  0
299 1.503322  0.707035 -1.035462 -0.374779  2.032684  1  0  1  1  0  1  2  3  0
300 0.285561 -0.091401 -2.235438 -1.511859  0.135360  1  0  0  1  1  1  1  3  0
301 0.285561 -0.091401 -0.203222  1.068439 -0.899544  0  1  0  0  0  1  1  2  0
302 rows x 14 columns

```

### 30. Plotting the confusion matrix for all the 10 classifiers.

A **confusion matrix** is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

An example of confusion matrix for a binary classifier

n=165	Predicted: NO	Predicted: YES
Actual: NO	50	10
Actual: YES	5	100

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

Jupyter Heart Prediction Last Checkpoint: 13 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted conda\_python3

0/2 rows x 14 columns

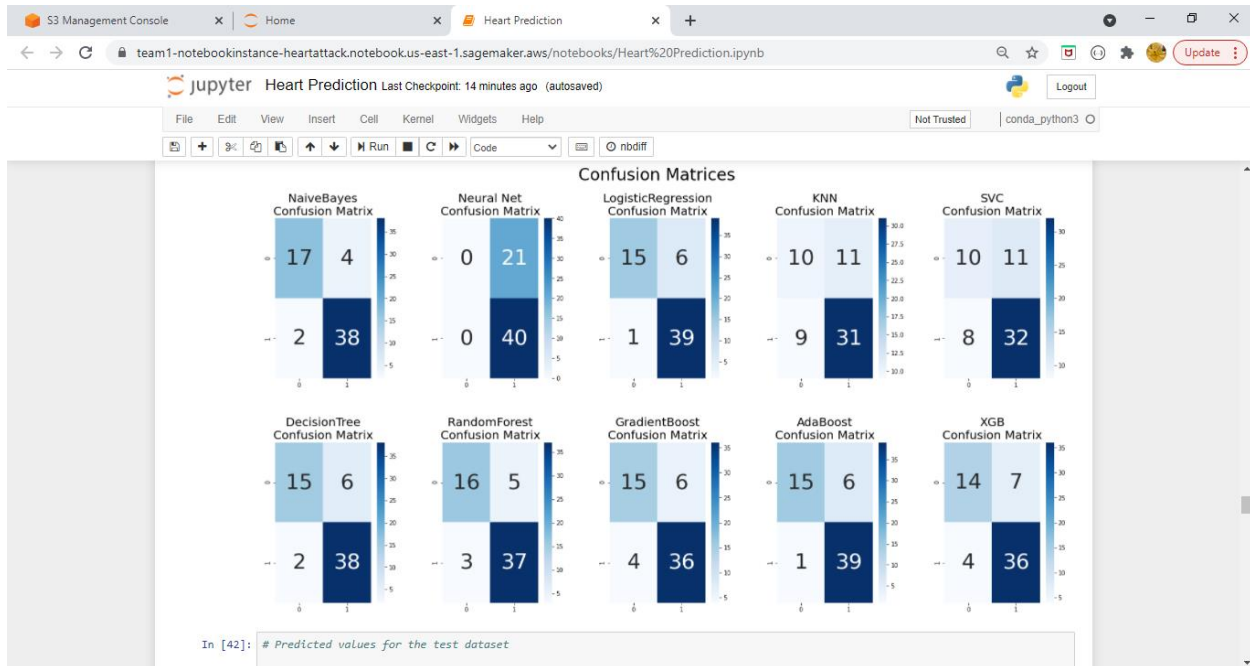
```
In [39]: key = ['NaiveBayes','Neural Net','LogisticRegression','KNN','SVC','DecisionTree','RandomForest','GradientBoost','AdaBoost','XGB']
value = [GaussianNB(),MLPClassifier(solver='lbfgs', alpha=1e-5,hidden_layer_sizes=(5, 2), random_state=1),LogisticRegression(ran
models = dict(zip(key,value))

In [40]: predicted =[]

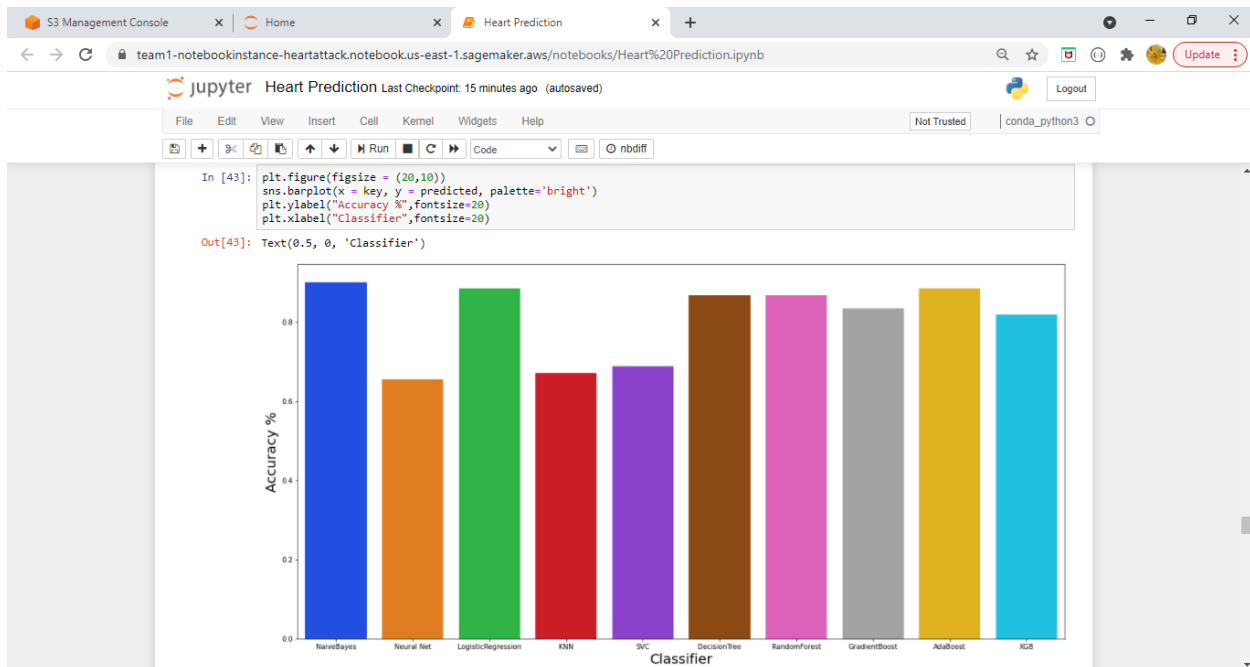
In [41]: i=1
plt.figure(figsize=(24,12))

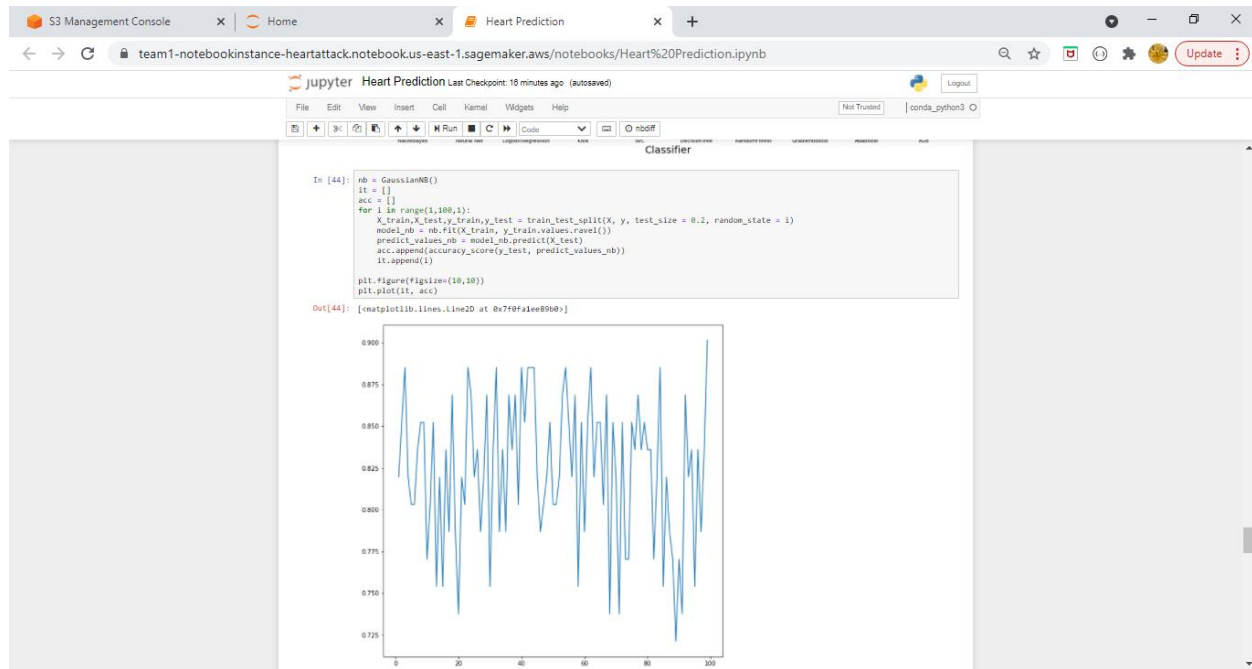
plt.suptitle("Confusion Matrices",fontsize=28)
plt.subplots_adjust(wspace = 0.4, hspace= 0.4)
for name,algo in models.items():
    model=algo
    model.fit(X_train,y_train)
    predict = model.predict(X_test)
    cm = confusion_matrix(y_test,model.predict(X_test))
    plt.subplot(2,5,i)
    i +=1
    plt.title(name+"\nConfusion Matrix",fontsize=20)
    sns.heatmap(cm,annot=True,cmap="Blues",fmt="d",cbar=True, annot_kws={"size": 36})
    acc = accuracy_score(y_test, predict)
    predicted.append(acc)
    print(name,acc)
```

```
NaiveBayes 0.9016393442622951
Neural Net 0.6557377049180327
LogisticRegression 0.8852459016393442
KNN 0.6721311475409836
SVC 0.6885245901639344
DecisionTree 0.8688524590163934
RandomForest 0.8688524590163934
GradientBoost 0.8360655737704918
AdaBoost 0.8852459016393442
[04:12:20] WARNING: ../src/learner.cc:1095: Starting in XGBoost 1.3.0, the default evaluation metric used with the objective 'b
inary:logistic' was changed from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore the old behavior.
XGB 0.819672131147541
```



31. Plotting the bar plot for all the 10 classifiers which will show us the efficiency and out of all the models we can see Naïve Bayes is the efficient one with efficiency more than 85 percentage.





**32. Once we have found our efficient model which is Naïve Bayes, we are exporting it to the S3 bucket.**

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

jupyter Heart Prediction Last Checkpoint: 16 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Not Trusted | conda\_python3

```
In [45]: #Exporting model using joblib library
import joblib
joblib.dump(model_nb,"team1-s3bucket-heartattack_model.pkl")

Out[45]: ['team1-s3bucket-heartattack_model.pkl']

In [47]: import tempfile
import boto3
s3 = boto3.resource('s3')

# dumping it in .pkl format
location = 'heart/'
model_filename = 'nbmodel.pkl'
OutputFile = location + model_filename

# WRITE
with tempfile.TemporaryFile() as fp:
    joblib.dump(model_nb, fp)
    fp.seek(0)

# bucket_name and OutputFile - s3 location path in string format.
s3.Bucket('team1-s3bucket-heartattack').put_object(Key= OutputFile, Body=fp.read())
```

S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

jupyter Heart Prediction Last Checkpoint: 17 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted | conda\_python3

```
In [48]: from ipynbwidgets import widgets
it1=widgets.BoundedFloatText(
    value=50,
    min=0,
    max=1000,
    step=1,
    description='Age:',
    disabled=False,
    layout={'width': 'max-content'})
display(it1)
d1=widgets.Dropdown(
    options=['1', '0'],
    value='1',
    description='Sex:',
    disabled=False,
    layout={'width': 'max-content'})
display(d1)
it2=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=1000,
    step=1,
    description='Cp:',
    disabled=False,
    layout={'width': 'max-content'})
display(it2)
it3=widgets.BoundedFloatText(
    value=140,
    min=0,
    max=1000,
    step=1,
    description='Trestbps:',
    disabled=False,
    layout={'width': 'max-content'})
```

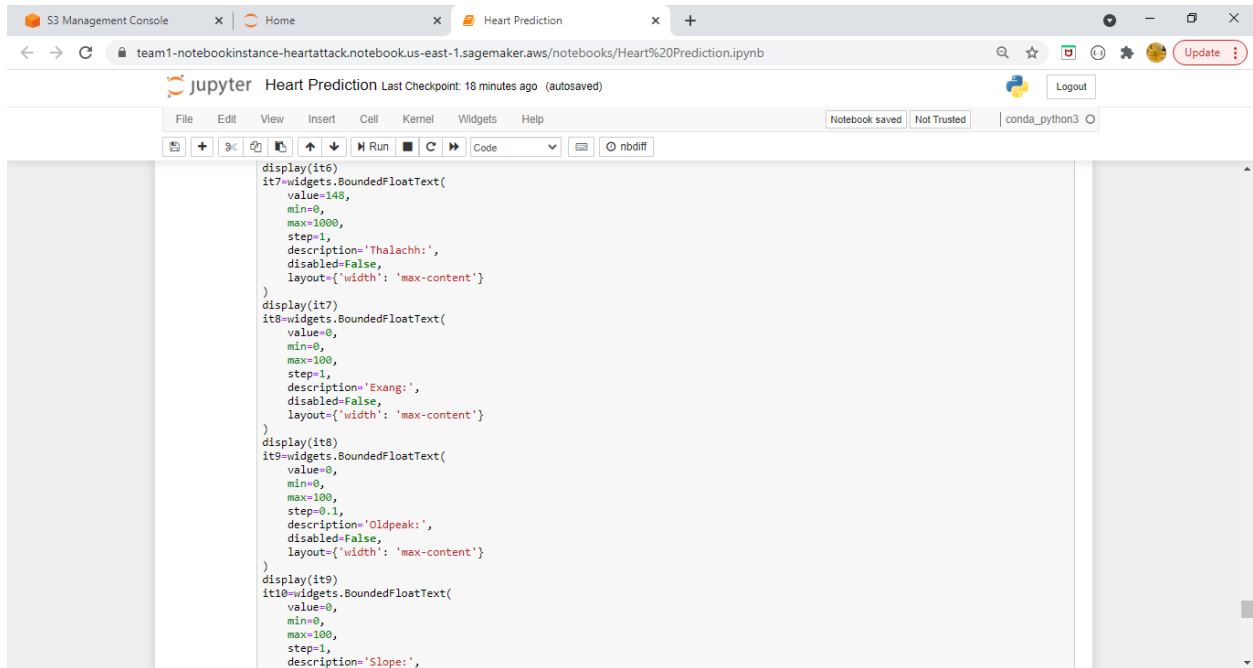
S3 Management Console x Home x Heart Prediction x +

team1-notebookinstance-heartattack.notebook.us-east-1.sagemaker.aws/notebooks/Heart%20Prediction.ipynb

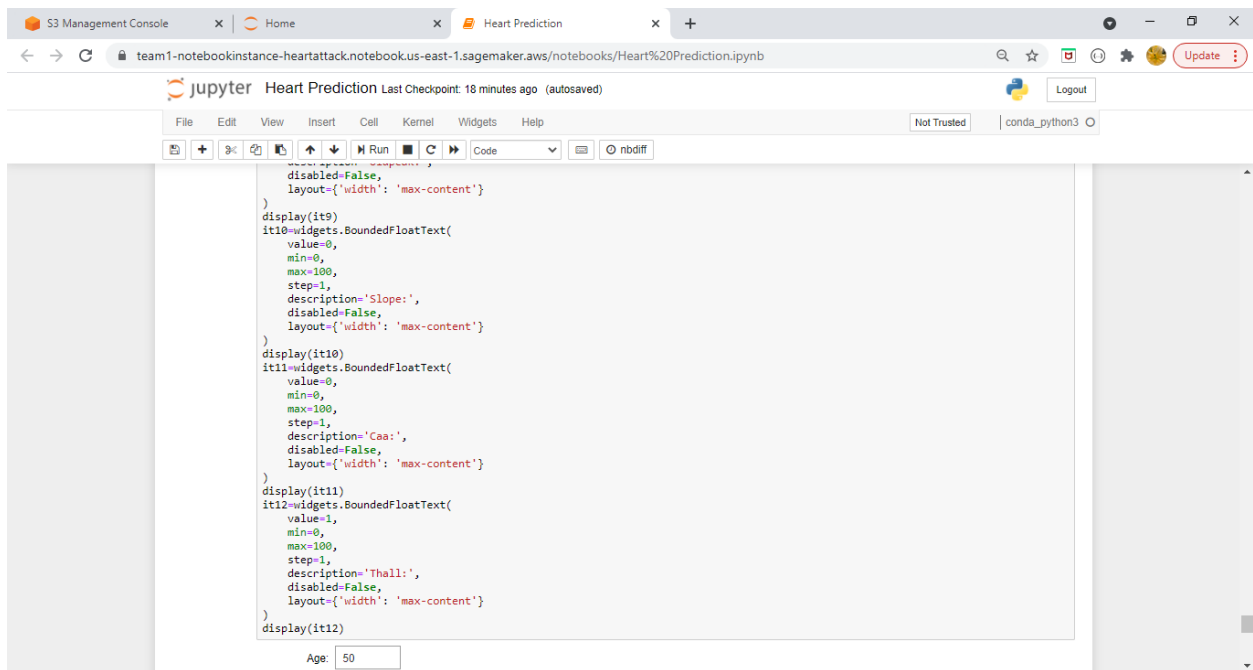
jupyter Heart Prediction Last Checkpoint: 18 minutes ago (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted | conda\_python3

```
display(it3)
it4=widgets.BoundedFloatText(
    value=190,
    min=0,
    max=1000,
    step=1,
    description='Cholestrol:',
    disabled=False,
    layout={'width': 'max-content'})
display(it4)
it5=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=1000,
    step=1,
    description='Fbs:',
    disabled=False,
    layout={'width': 'max-content'})
display(it5)
it6=widgets.BoundedFloatText(
    value=1,
    min=0,
    max=100,
    step=1,
    description='Restecg:',
    disabled=False,
    layout={'width': 'max-content'})
display(it6)
it7=widgets.BoundedFloatText(
    value=148,
    min=0,
    max=1000,
    step=1,
    description='Thal:',
```



```
display(it6)
it7=widgets.BoundedFloatText(
    value=145,
    min=0,
    max=1000,
    step=1,
    description='Thalachh:',
    disabled=False,
    layout={'width': 'max-content'})
display(it7)
it8=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=100,
    step=1,
    description='Exang:',
    disabled=False,
    layout={'width': 'max-content'})
display(it8)
it9=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=100,
    step=0.1,
    description='Oldpeak:',
    disabled=False,
    layout={'width': 'max-content'})
display(it9)
it10=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=100,
    step=1,
    description='Slope:',
```

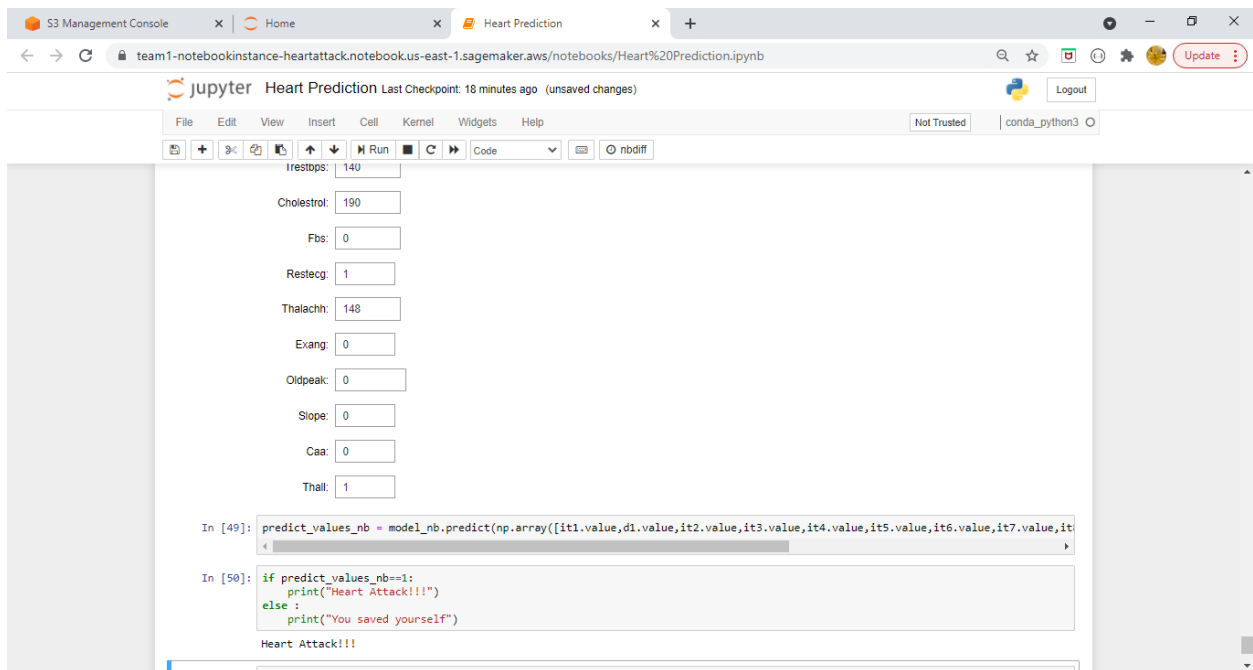
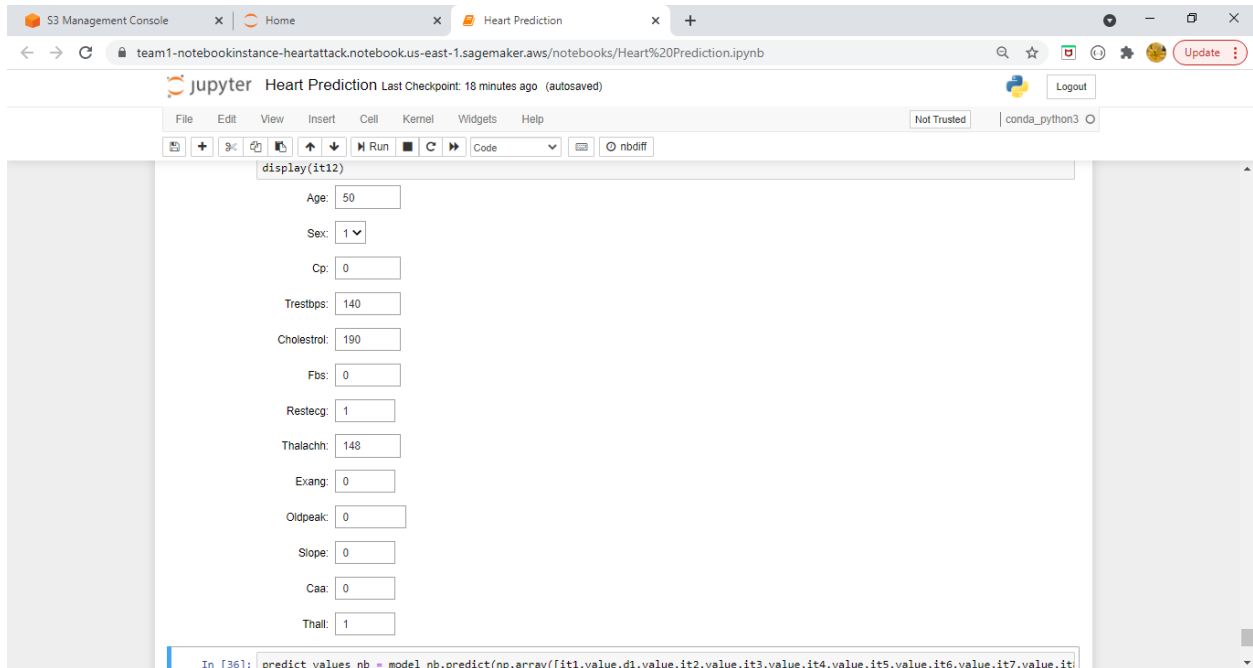


```
display(it9)
it10=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=100,
    step=1,
    description='Slope:',
    disabled=False,
    layout={'width': 'max-content'})
display(it10)
it11=widgets.BoundedFloatText(
    value=0,
    min=0,
    max=100,
    step=1,
    description='Caa:',
    disabled=False,
    layout={'width': 'max-content'})
display(it11)
it12=widgets.BoundedFloatText(
    value=1,
    min=0,
    max=100,
    step=1,
    description='Thall:',
    disabled=False,
    layout={'width': 'max-content'})
display(it12)
```

Age: 50

**33. Also provided the interface to enter values for an individual and on basis of the data, it will predict whether the individual is at risk of heart attack or not.**





**34. Naïve Bayes Model is exported to S3 bucket under folder heart which we have defined in Heart Prediction.ipynb file.**

Amazon S3

Buckets

Access Points

Object Lambda Access Points

Batch Operations

Access analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

AWS Organizations settings

Feature spotlight

AWS Marketplace for S3

Amazon S3 > team1-s3bucket-heartattack

team1-s3bucket-heartattack

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (3)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy URL

Open

Download

Delete

Actions

Create folder

Upload

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	heart_function-1.0.0.jar	jar	May 4, 2021, 23:14:11 (UTC-04:00)	8.1 MB	Standard
<input type="checkbox"/>	heart.csv	csv	May 4, 2021, 23:14:09 (UTC-04:00)	11.1 KB	Standard
<input type="checkbox"/>	heart/	Folder	-	-	-

Feedback

English (US)

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Privacy Policy

Terms of Use

Cookie preferences

Amazon S3

Buckets

Access Points

Object Lambda Access Points

Batch Operations

Access analyzer for S3

Block Public Access settings for this account

Storage Lens

Dashboards

AWS Organizations settings

Feature spotlight

AWS Marketplace for S3

Amazon S3 > team1-s3bucket-heartattack > heart/

heart/

Objects

Properties

Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Copy URL

Open

Download

Delete

Actions

Create folder

Upload

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	nbmodel.pkl	pkl	May 5, 2021, 00:12:41 (UTC-04:00)	1.1 KB	Standard

Feedback

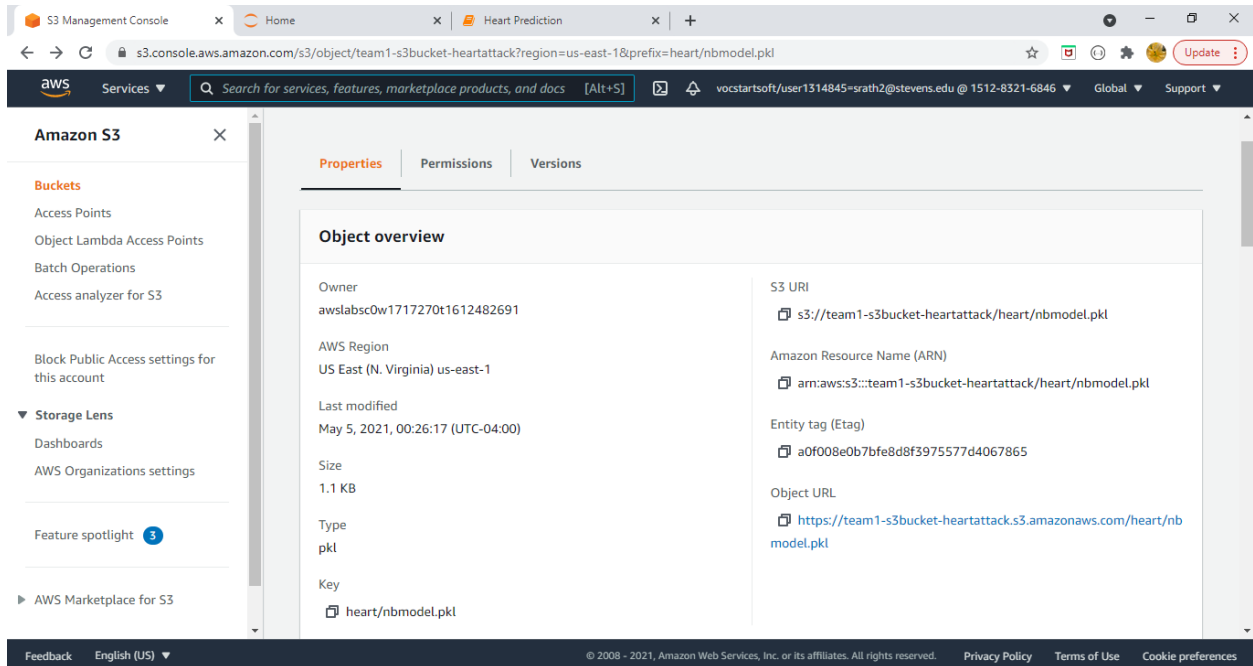
English (US)

© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

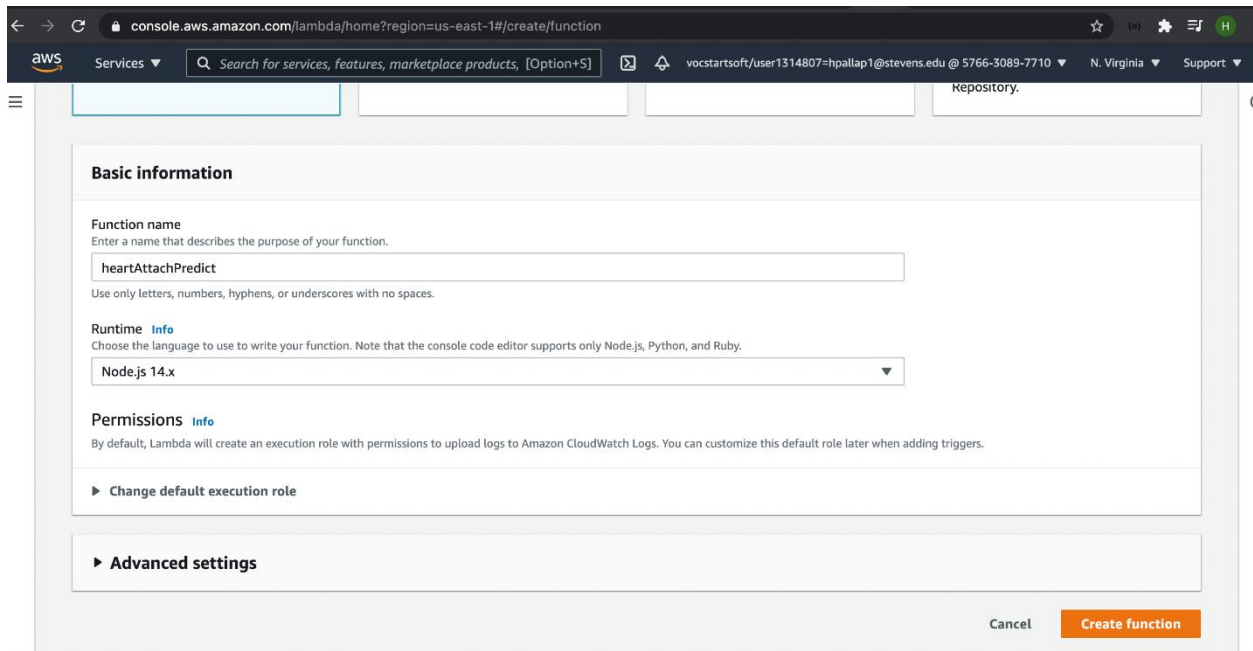
Privacy Policy

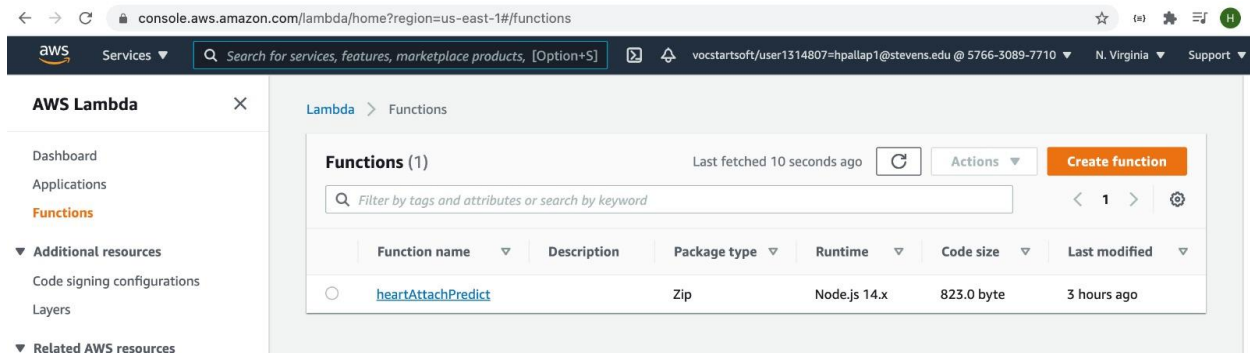
Terms of Use

Cookie preferences

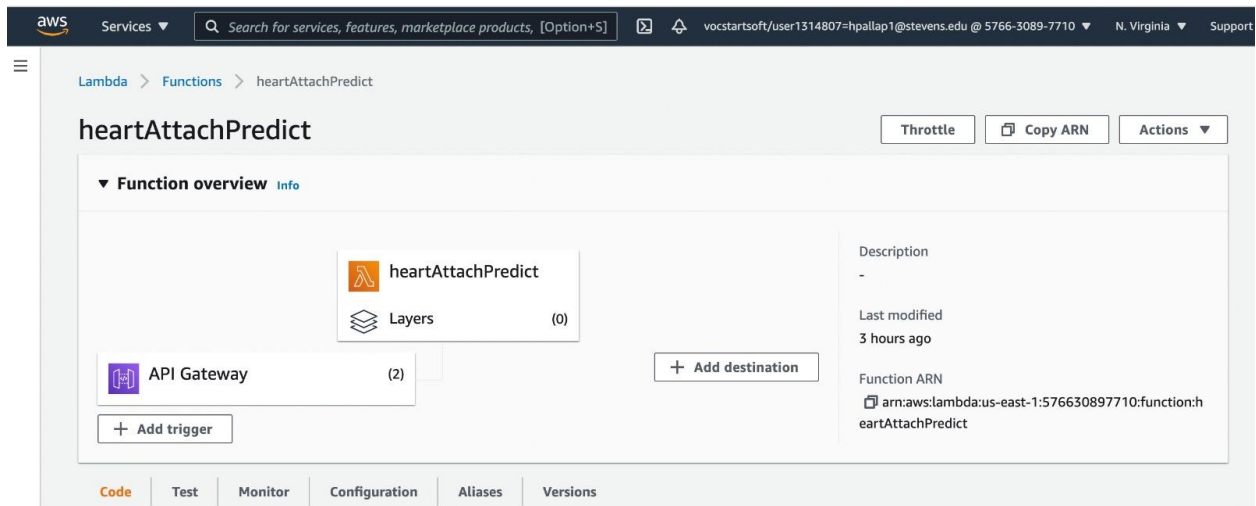


35. Lambda function will access the most efficient model from S3 bucket, so that if user submits his/her data via endpoint url then user will get the result whether he is at risk or heart attack or not.





36. Rest API trigger is added to invoke the lambda function whenever user hits the endpoint url.



37. The code source part to predict the heart attack by using the model weight values which is calculated in notebook instance while training the model and also S3 bucket name is provided so that it will feed the user's data into the trained model.

38. Endpoint URL to invoke the lambda function.

aws Services Search for services, features, marketplace products, [Option+S] vocstartsoft/user1314807=hpallap1@stevens.edu @ 5766-3089-7710 N. Virginia Support

+ Add trigger

Code Test Monitor **Configuration** Aliases Versions

General configuration

**Triggers**

Permissions

Destinations

Environment variables

Tags

VPC

Monitoring and operations tools

Concurrency

**Triggers (1)** Enable Disable Fix errors Delete Add trigger

API Gateway

1 match

Trigger

**API Gateway: heartAttachPredict-API**  
arn:aws:execute-api:us-east-1:576630897710:o4m2twwi8h/\*/\*/heartAttachPredict

Details

API endpoint: <https://o4m2twwi8h.execute-api.us-east-1.amazonaws.com/default/heartAttachPredict>

API type: REST

Authorization: NONE

Method: ANY

Resource path: /heartAttachPredict

Stage: default

**39. Invoking the lambda function through postman as the method is post by providing the JSON object with the individual data and it will show the response whether the individual is at risk of heart attack or not.**

GET https://o4m2twwi8h.execute-api.us-east-1.amazonaws.com/default/heartAttachPredict Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL **JSON** Beautify

```

1 {
2   "age": 57,
3   "sex": 0,
4   "cp": 0,
5   "trtbps": 140,
6   "chol": 241,
7   "fbs": 0,
8   "restecg": 1,
9   "thalachh": 123,
10  "exng": 1,
11  "oldpeak": 0.2,
12  "slp": 1,
13  "caa": 0,
14  "thall": 3
15
16 }

```

Body Cookies Headers (7) Test Results 200 OK 571 ms 311 B Save Response

Pretty Raw Preview Visualize JSON

1 No HeartAttack Predicted!

https://o4m2twwi8h.execute-api.us-east-1.amazonaws.com/default/heartAttachPredict

GET https://o4m2twwi8h.execute-api.us-east-1.amazonaws.com/default/heartAttachPredict Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "age": 63,
3   "sex": 1,
4   "cp": 3,
5   "trtbps": 145,
6   "chol": 233,
7   "fbs": 1,
8   "restecg": 0,
9   "thalachh": 150,
10  "exng": 0,
11  "oldpeak": 2.3,
12  "slp": 0,
13  "caa": 0,
14  "thall": 1
15
16 }
```

Body Cookies Headers (7) Test Results 200 OK 193 ms 308 B Save Response

Pretty Raw Preview Visualize JSON

```
1 HeartAttack Predicted!
```

## References

<http://scikit-learn.org>

<https://docs.aws.amazon.com/sagemaker/latest/dg/nbi.html>

<https://aws.amazon.com/lambda/>

<https://aws.amazon.com/s3/>

<https://www.healthline.com/health/heart-disease/statistics#Who-is-at-risk?>

<https://www.kaggle.com/rashikrahmanpritom/heart-attack-analysis-prediction-dataset>