Supervised Machine Learning – Cardiovascular Risk Prediction

Sethupathy M

Sri harish A

Capstone project 3 – Alma Better

Abstract:

The purpose of this project is to try a machine learning approach for Cardiovascular Risk Prediction by given the id, sex, and information about the health condition of the person. This project contains: Exploratory data analysis, feature engineering, choosing appropriate features, cross algorithms, cross validation, tuning the algorithms, analysis of feature importance, analysis of model performance. The predictions of future could help for saving a person's life and could help them to get rid of unhealthy habits. Another point of view is to test the machine learning algorithms how good are at solving this problem. With RandomForestRegressor best Precision Recall AUC score of 97.8% is achieved.

In this experiment by using EDA, we are analyzing and visualizing the different perspectives. Which gives us different insights like

- Distribution of Independent and dependent variables.
- How does the Ten-year risk of CHD vary with Age?
- How does the Ten-year risk of CHD vary with Sex?
- How does the Ten-year risk of CHD vary with education?
- How does the Ten-year risk of CHD vary with Diabetes?
- On which medical factors contribute more to develop a risk of CHD.
- And also, we have used various Machine Learning algorithms to predict the Ten-year risk of CHD.

Problem Statement

This project contains the data record of Patients who could possibly may or not develop a risk of Cardiovascular disease in upcoming 10 years. The Dataset contains the details like id, sex, age, medical records etc. There are a total of 17 variables describing 3390 observations. Each observation represents whether the patient will or will not develop a risk of Cardiovascular disease in upcoming 10 years.

Features:

Independent variables:

- **Id** – Unique identification number for each patient.
- **Age** – Age of the patient (Continuous).
- **Education** – Educational qualification of the patients(ordinal).
- **Sex** – Male or Female (‘M’ or ’F’).
- **Is_smoking** – Whether or not the patient is a current smoker (Binary).
- **CigsPerDay** – The number of cigarettes that the person smoked on average in one day (Continuous).
- **BPMeds** – Whether or not the patient was on blood pressure medication (Nominal).
- **prevalentStroke** – Whether or not the patient previously had a stroke (Nominal).
- **prevalentHyp** – Whether or not the patient was hypertensive (Nominal).
- **Diabetes** – Whether or not the patient had diabetes (Nominal).
- **Tot_Chol** – Total Cholesterol level (Continuous).
- **Sys_**BP – Systolic blood pressure (Continuous).
- **Dia_**BP – Diastolic blood pressure (Continuous).
- **BMI** – Body Mass Index (Continuous).
- **Heart rate** – The speed at which the heart beats (Continuous).
- **Glucose** – Glucose level in blood (Continuous).

Dependent variable:

- **TenYearCHD –** Whether the patient will develop a risk of Cardiovascular Disease in 10 years (Binary)

Introduction

Cardiovascular disease is the leading cause of death worldwide and a major public health concern. Therefore, its risk assessment is crucial to many existing treatment guidelines.

Risk estimates are also being used to predict the magnitude of future cardiovascular disease mortality and morbidity at the population level and in specific subgroups to inform policymakers and health authorities about these risks. Additionally, risk prediction inspires individuals to change their lifestyle and behavior and to adhere to medications.

The disease is caused when the heart's blood vessels, the coronary arteries, become narrowed or blocked and can't supply enough blood to the heart. This can cause a heart attack or angina. The good news is that if we tackle these risk factors, we can reduce our numbers of people developing heart disease.

By CHD risk prediction, we can analyze the risk factors which are contributing more to develop a heart disease.

The ability to predict risk for and from cardiovascular disease is increasingly important for several reasons. Perhaps foremost among these is the need to determine individual risk to better plan investigation and management with the greatest accuracy and safety for patients and lowest costs for the health care system. This will be even more important as personalized medicine becomes more widespread.

The purpose of cardiovascular disease risk prediction

CHD risk calculators are used to determine an individual’s risk of developing CHD in the short term, generally five or 10 years, to determine if the patient is at high risk. Prescription of preventive medication, generally both lipid-lowering and blood pressure–lowering medication, is recommended for individuals at high risk, for whom it is most likely that benefits will outweigh the harms. CHD risk calculators should not be used to determine whether counselling and advice for lifestyle factors should be given to individuals, as this is recommended regardless of the CHD risk estimate. Patients who smoke or are obese or not physically active should be encouraged to adopt healthier lifestyles. Patients with very high blood pressure or lipid levels should also be prescribed medication to lower these, and patients with other conditions that are considered high risk for CHD should be treated with both medications, irrespective of estimated CHD risk

Steps involved:

Outliers Treatment

Our dataset has many outliers which could affect the effectiveness of the model. So, by using bar plot outliers had been detected and by using Z- score technique all the outliers had been imputed with median.

Treating Null Values

Our dataset has missing values with many features which could lead us to loss of information and can mislead to wrong results so addressing Null values are important Null values are imputed with closest mean value as possible.

Exploratory Data Analysis

After loading the dataset, we performed this method by comparing our target variable that is ‘TenYearCHD with other independent variables. This process helped us figure out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

Building structured multi-plot grids and graphs

When exploring multidimensional data, a useful approach is to draw multiple instances of the same plot on different subsets of your dataset. It allows a viewer to quickly extract a large amount of information about a complex dataset. Matplotlib offers good support for making figures with multiple axes; seaborn builds on top of this to directly link the structure of the plot to the structure of your dataset.

We have used count plot, box plot, distplot and pie chart in multi-plots for various features and the graphs provide a high level of convenience for comparison

Multicollinearity for all the variables has been plotted and some highly correlated variables have been detected.

Feature Engineering

Our dataset has high class imbalance which cannot can be directly fed into the models without feature engineering. RandomOverSampling is a sampling technique which Randomly duplicate examples in the minority class. For our dataset RandomOverSampling is found to be more effective.

Feature Selection

Feature Selection is the method of reducing the input variable to your model by using only relevant data and getting rid of noise in data. For our dataset all features except 'id' plays significant role in enabling the model to correctly classify the Target variable.

The Metric Trap

One of the major issues when dealing with imbalanced datasets relates to the metrics used to evaluate their model. Using simpler metrics like accuracy score can be misleading. In a dataset with highly imbalanced classes, the classifier will always "predict" the most common class without performing any analysis of the features and it will have a high accuracy rate, obviously not the correct one.

Choosing Appropriate metric

A classifier is only as good as the metric used to evaluate it. Choosing an appropriate metric is challenging generally in applied machine learning, but is particularly difficult for imbalanced classification problems. Firstly, because most of the standard metrics that are widely used assume a balanced class distribution, and because typically not all classes, and therefore, not all prediction errors, are equal for imbalanced classification. Appropriate metrics for our dataset would be Precision Recall AUC score and Confusion matrix

True Negative (TN) →Persons who dont have a risk of 10-
year CHD, and our model predicted that they dont have a risk of 10-year CHD.
True Positive (TP) → Persons who have a risk of 10-
year CHD, and our model predicted that they have a risk of 10-year CHD.
False Negative (FN) → Persons who have a risk of 10-
year CHD, and our model predicted that they dont have a risk of 10-year CHD.
False Positive (FP) → Persons who dont have a risk of 10-
year CHD, and our model predicted that they have a risk of 10-year CHD**.**

Precision - It summarizes the fraction of examples assigned the positive class

that belong to the positive class.
Recall - It summarizes how well the positive class was predicted and is the same calculation as sensitivity.

For our dataset,
Precision→Shows how correct the model is, when it predicts that a person has a risk of 10-year CHD.
Recall → For all patients who have a risk of 10-year CHD, how many predictions the model have correctly predicted as having a risk of 10- year CHD.
True Negative → Should be low as possible because we don't our model to classify a patient as No risk of CHD but he actually has a risk of CHD.

Data Preparation for model
Train Test Split
Train test split is a model validation procedure that allows you to simulate how a model would perform on new/unseen data.
Standardization
Standardizing a dataset involves rescaling the distribution of values so that the mean of observed values is 0 and the standard deviation is 1.

Building Machine Learning Models:
Logistic Regression

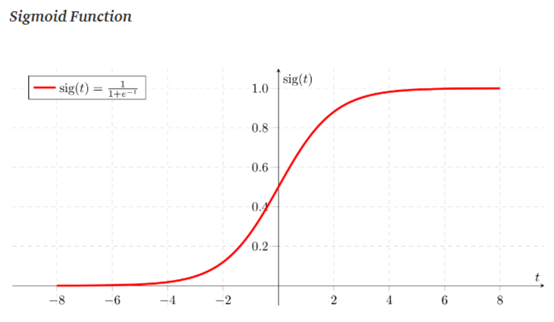
Logistic Regression is used when the dependent variable (target) is categorical.

For example,

- To predict whether a Patient has a risk of 10-Year CHD (1) or not (0)
- To predict whether an email is spam (1) or (0)

Consider a scenario where we need to classify whether the Patient has a risk of 10-Year CHD or not. If we use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done. Say if the actual class is risk, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not risk which can lead to serious consequence in real time.

From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. **Their value strictly ranges from 0 to 1.**



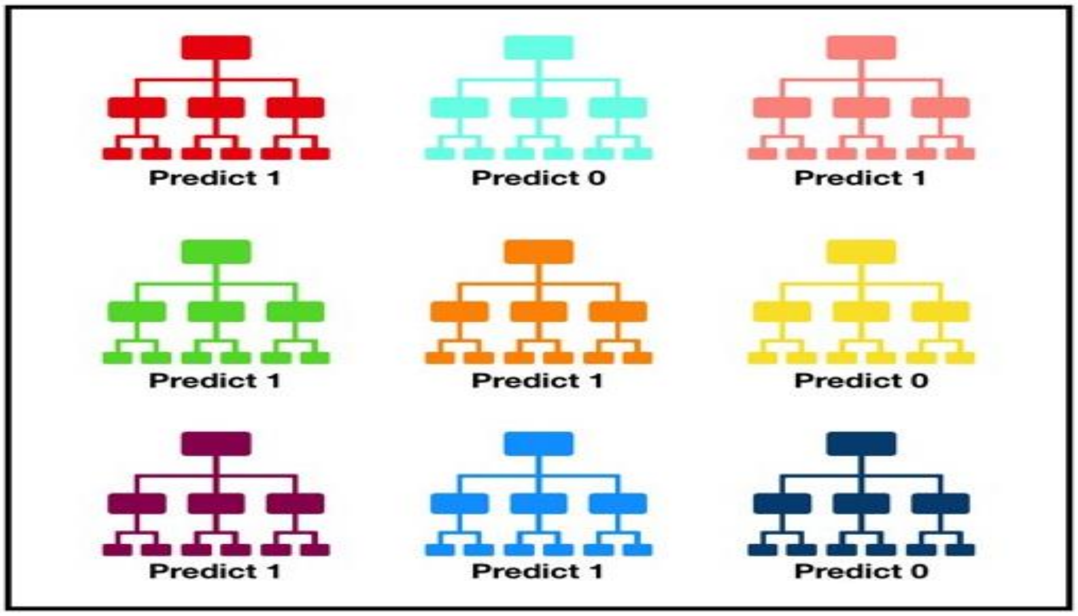
RandomForestClassifier

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. **The reason for this wonderful effect is that the trees protect each other from their individual errors** (as long as they don’t constantly all err in the same direction). While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So, the prerequisites for random forest to perform well are:

- There needs to be some actual signal in our features so that models built using those features do better than random guessing.
- The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other.



GradientBoostClassifier

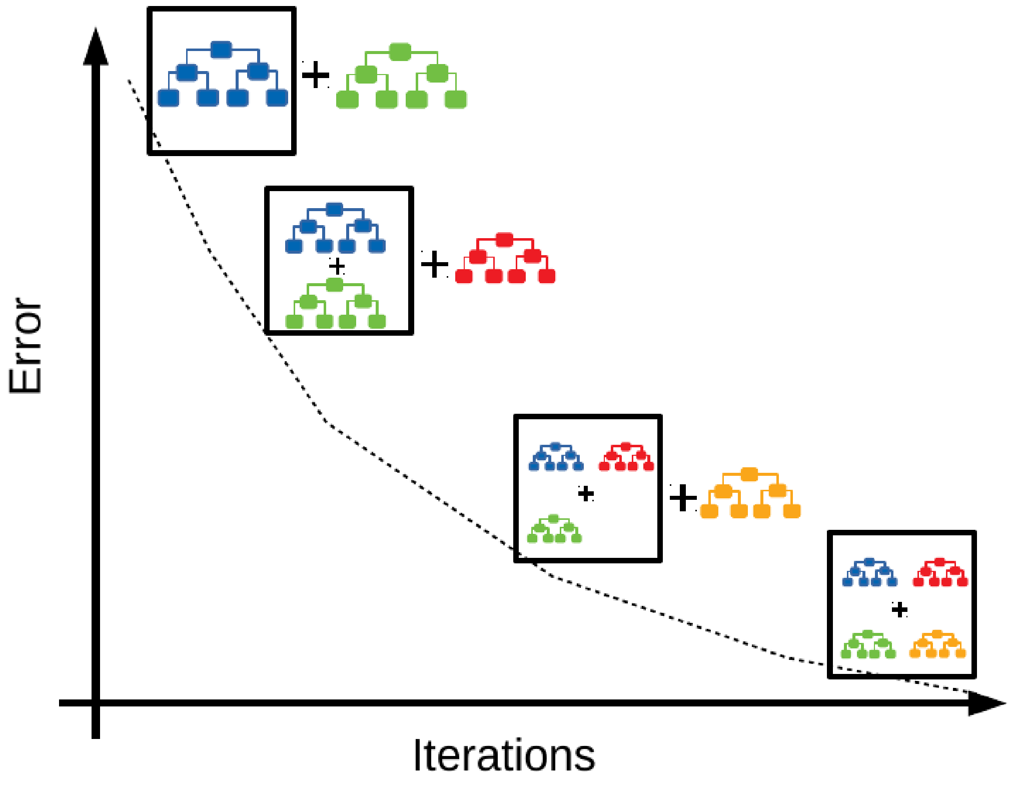
In Gradient Boosting, each predictor tries to improve on its predecessor by reducing the errors. But the fascinating idea behind Gradient Boosting is that instead of fitting a predictor on the data at each iteration, it actually fits a new predictor to the residual errors made by the previous predictor.

Now, to make new predictions, we do two things:

- get the log(odds) prediction for each instance in the training set
- convert that prediction into a probability

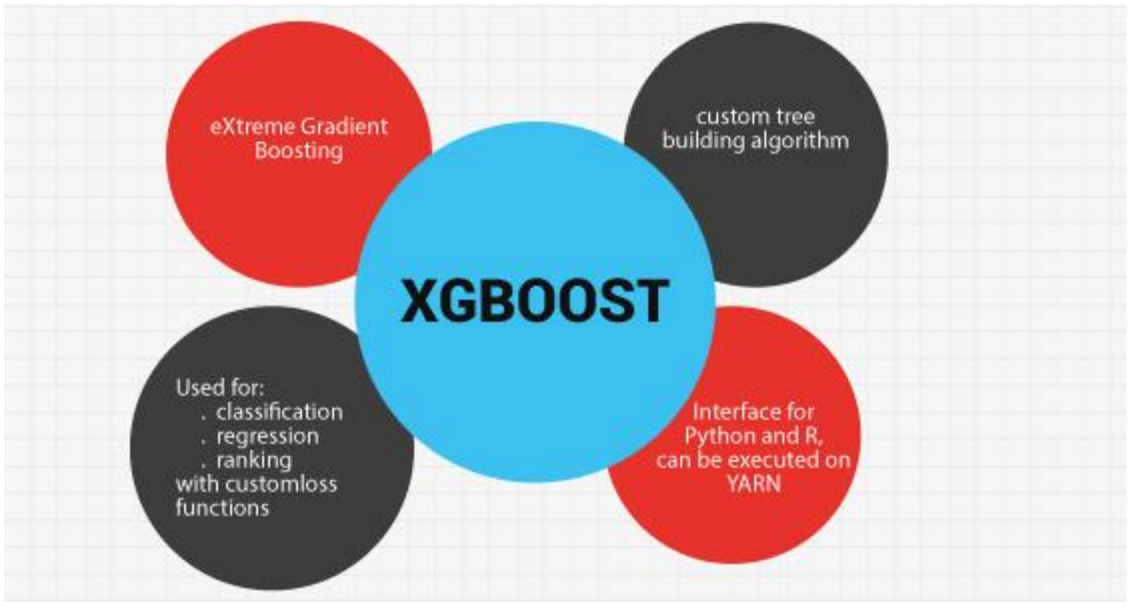
The learning rate is a hyperparameter that is used to scale each trees contribution, sacrificing bias for better variance. In other words, we multiply this number by the predicted value so that we do not overfit the data.

Once we have calculated the log(odds) prediction, we now must convert it into a probability using the previous formula for converting log(odds) values into probabilities.



XGB Classifier

XGB Classifier is an implementation of gradient boosted decision trees designed for speed and performance that is dominative competitive machine learning.

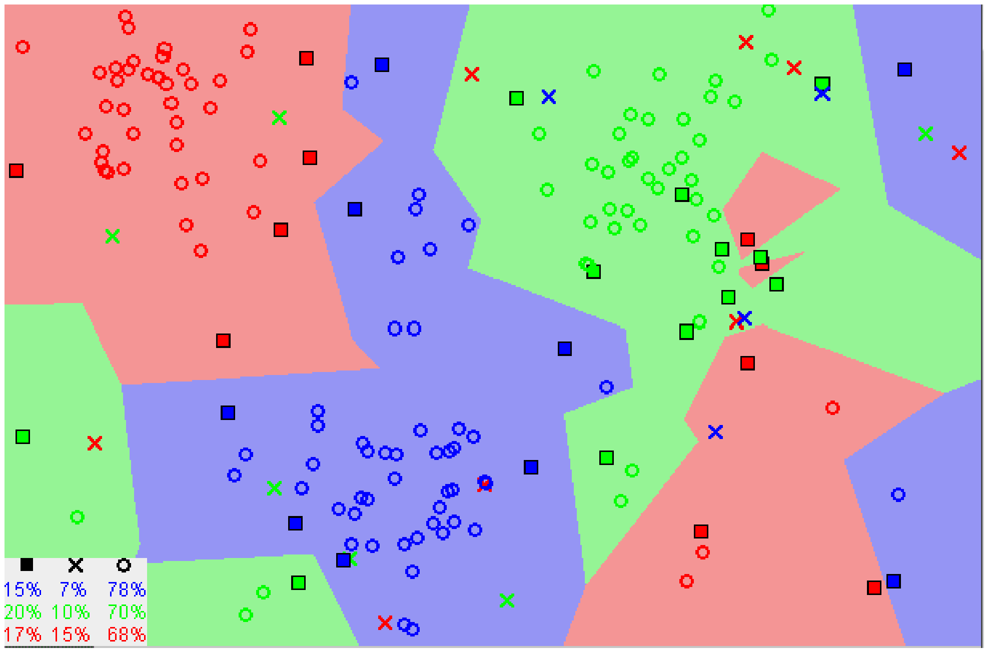

eXtreme Gradient Boosting
custom tree building algorithm
XGBOOST
Used for:
. classification
. regression
. ranking
with customloss functions
Interface for Python and R, can be executed on YARN

KNNeighbors Classifier

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems.

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other.

Notice in the image below that most of the time, similar data points are close to each other. The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.



Libraries Used

Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Pandas is mainly used for data analysis and associated manipulation of tabular data in Data Frames. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features. The development of pandas introduced into Python many comparable features of working with Data frames that were established in the R programming language. The pandas library is built upon another library NumPy, which is oriented to efficiently working with arrays instead of the features of working on Data frames.

NumPy

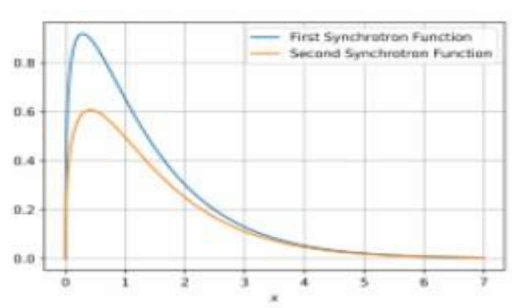
NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted, and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.

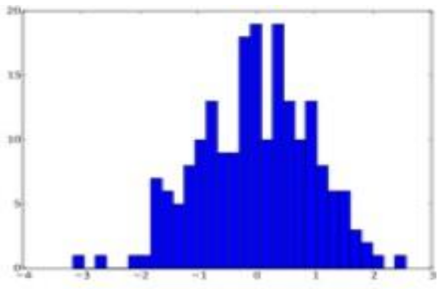
Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as a universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

Matplotlib

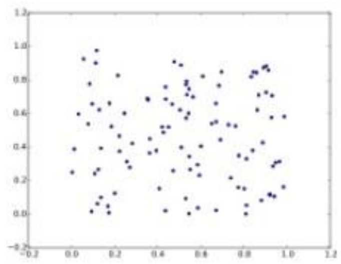
Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.



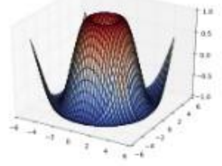
Line plot



Histogram



Scatter plot



3D plot

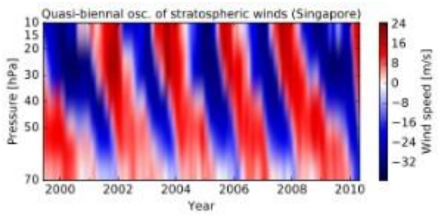
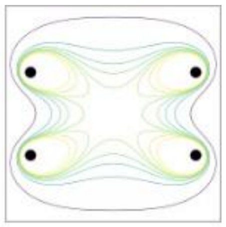
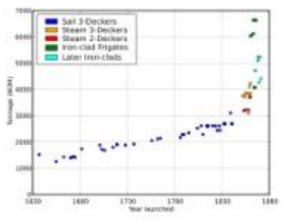


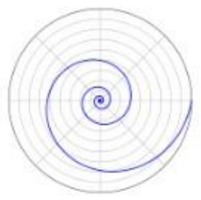
Image plot



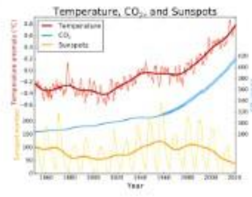
Contour plot



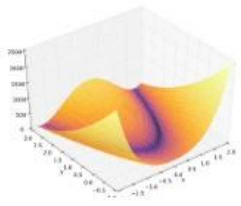
Scatter plot



Polar plot



Line plot



3-D plot

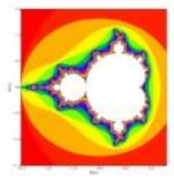
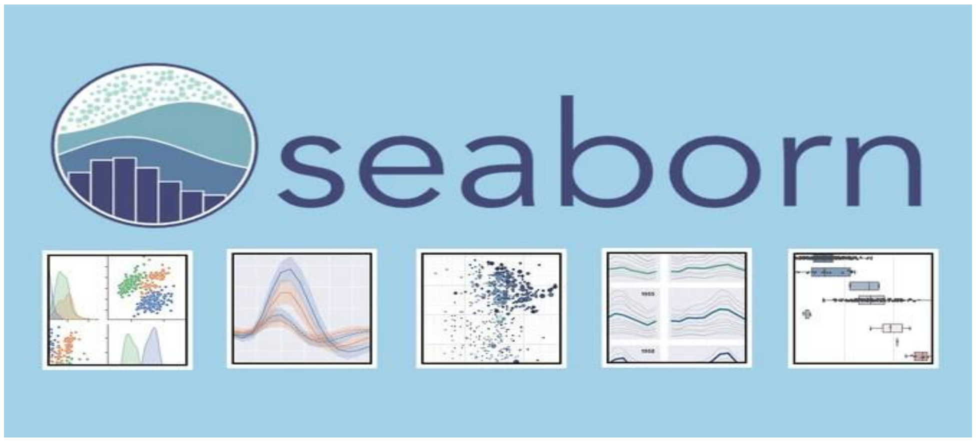


Image plot

Seaborn

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures.

Seaborn helps you explore and understand your data. Its plotting functions operate on data frames and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots. Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them. Behind the scenes, seaborn uses matplotlib to draw its plots


seaborn

Conclusion:

- It’s quite obvious that smokers have a high risk of 10-year CHD.
- Patients with no history of hypertensive, stroke, diabetes have less risk of 10-year CHD.
- By using simple Logistic Regressor algorithm we were able to get the Precision Recall AUC score of 68 %
- Very little improvement in Precision Recall AUC score after using Random Forest classifier because of data imbalance.
- Data Imbalance is addressed by Random Over sampling technique.
- We got the best Precision Recall AUC score of 97.8 percent using Random Forest Classifier with cross validation and hyper parameter tunning.
- Top five most important features are Sys_BP, Glucose, Totchol, age, CigsPerDay
- For the given dataset, Random Forest Classifier has proven to be the best fit model with,
    - Train Precision Recall AUC score: 99.9 %
    - Test Precision Recall AUC score : 97.8%
    - Train ROC AUC score : 99.9 %
    - Test ROC AUC score : 97.4%