# Model for early prediction of coronary Heart disease based on vitals.

**I have also written these notes in the notebook also.**

## Problem Statement

We have collected some health vitals that can be used to predict if the patient will acquire a coronary heart disease in the next 10 years, we can call these data points as X. For the collected samples we have the ground truth stating that whether the corresponding case acquired that disease in the next 10 years or not, we can call this data pointer Y. We now want to train a model that can take in those data points (X) and predict whether that case is vulnerable to acquire a coronary Heart disease (ŷ). We can apply this model on any other patient and classify him as vulnerable to that disease or not, doing so will help us take necessary steps in advance.

## Exploratory Data Analysis

In this step we would read the dataset and try to understand our data, we would usually start with checking the dataypes and data ranges.

Looking at the data we got some interesting insights,

- some columns like IV has negative values,

- column A15 and A16 has -99 values which can be invalid values, or no data values.

- Target column has 33% positive samples and 77% negative samples.

- Column A13, A18 have very less STD there can be very less information in those columns

## Data Pre-Processing

In this step we would like to read the data and understand format of data and prepare it in a format that can be used to train the model. Our model is as good as our data and I am a believer of 70:30 ratio 70% of my model is data and 30% is the architecture or algorithm. Anyway, better algorithm is always good, but my point is we should be very careful while preparing our dataset to avoid pitfalls.

## Select Training data, test data

Selection of training and testing set is done to check our model’s performance on unseen data usually the accuracy which we print for our model is derived on validation set only (I am pronouncing our testing set as validation set here). W9e would like to keep 80% of our total data as training set and remaining 20% as our validation set (testing set). We cannot simply keep aside 80% and 20% set, the validation set should reflect as a reduced or thinned down version of training set. We used sklearn’s `train\_test\_split()` function to split our data frame to training and testing set.

## Train the model

## 

## Test the model (Predictions and reporting)

## Evaluate the model performance

## Suggest ways of improving the model

The model can be further improved with more data and a better algorithm that can handle biased data. Also, if we apply focal loss over these types of problem, we would get better results.

## Any interesting observations

This problem is so simple that even without any tricks we were able to achieve 80% accuracy, these seems to be directly coming from the Bayes theory. The dataset is having a smaller number of positive samples which further makes our model bias towards predicting fewer positive samples. Even if we have a testing set, to test the robustness of our model needs to be tested with a different dataset with different number of positive over negative samples.

## Challenges faced and how you mitigated the challenges

We initially used an SVM and did not normalize our data, we fed the data as it is inside the model, doing this we were able to get an accuracy of 80%. Then we started experimenting with the data pipeline and hyperparameters. We started with reducing the no of parameters by checking for the most important ones, then we normalized the dataset and finally we used a random forest classifier. With each of these changes we were able to improve our model accuracy to nearly 90%.

## Assumptions if any

No, I am not aware of.