HEALTH CARE: CHECK HEART ATTACK POSSIBILITY

Data Set used: https://www.kaggle.com/nareshbhat/health-care-data-set-on-heart-attack-possibility
We started working on this project as a part of a hackathon wherein we were asked to build a model that would predict the occurrence of heart attacks.

Firstly, I'll analyze the data set and would rectify if we have any entry missing in the data set by the use of imputers in the sklearn library. The **Imputer** fills missing values with some statistics (e.g. mean, median, ...) of the data. Then I'll plot the histogram and scatter-plot to see the correlation between different fields.

After that, I'll select the best features as obtained after the correlation and proceed with the top 8 features that have more scores than the others with the help of sklearn feature selection library using the SelectKbest and chi2 functions. With these features, I'll standardize my data set using StandardScalers. **StandardScaler** standardizes a feature by subtracting the mean and then scaling to unit variance. I'll also use the get_dummies function to manipulate the data. It converts categorical data into dummy or indicator variables.

Now, I'll split the data into a train set and test set with a suitable split ratio like 0.2. **Partitioning data into training**, **validation**, and testing **sets** allows you to develop highly accurate models that are relevant to **data** that you collect in the future, not just the **data** the model was trained on.

Now, I'll select which model to use by training and testing my algorithm on more available models like KNN, SVM, RF, LR and proceed with the one that gives the best accuracy and higher CV-Score.

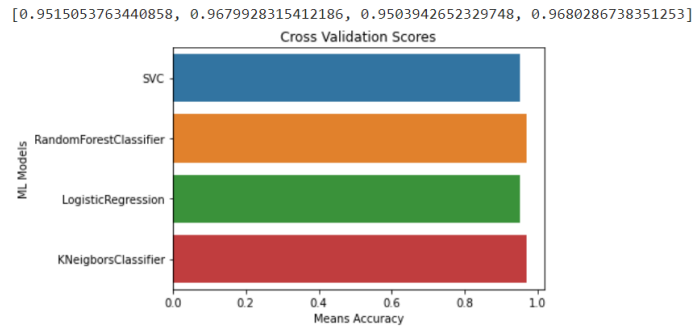
In **KNN**, We can implement a KNN model by following the below steps:

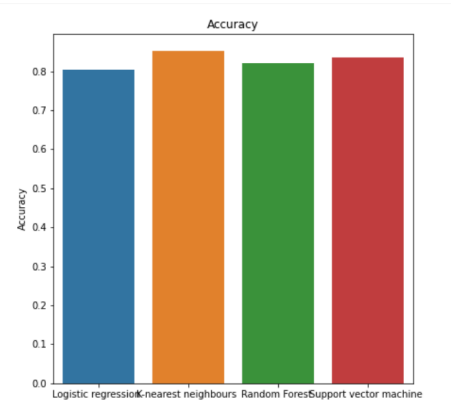
1. Load the data
2. Initialise the value of k
3. For getting the predicted class, iterate from 1 to total number of training data points.
4. I'll plot the error rate vs K value graph and proceed with the value of K that gives the least error.

5. The model after fitting the training sets will give me a prediction over my test set and a confusion matrix that will eventually give me the accuracy of the model.

In **SVM**, I'll use this model to predict my accuracy by fitting the training sets and performing a grid search to optimize the values of C and gamma hyperparameters. A smaller value of C creates a small-margin hyperplane and a larger value of C creates a larger-margin hyperplane. A lower value of Gamma will loosely fit the training dataset, whereas a higher value of gamma will exactly fit the training dataset, which causes overfitting.

Logistic Regression is a classification algorithm that is used to predict the probability of a categorical dependent variable. The decision boundary of logistic regression is a linear binary classifier that separates the two classes we want to predict using a line, a plane, or a hyperplane.
Scikit Learn has a Logistic Regression module which I will be using to build our machine learning model. The model will predict(1) if the heart disease is found and (0) if not.





After all this, I'll compare the CV-scores and accuracy results of all the models and wherever I get the maximum values of these, I'll use that model in my website which will be developed using the Django web framework. It will be a user-friendly web page where the user can input all the values in the fields provided and based on this it'll predict if the user has heart disease or not.