Capstone Project
Cardiovascular Risk Prediction

Team Members:
Vidit Ghelani
Mahesh Lakum

Problem Statement ?

A dataset of patients' information of over 4,000 records and 15 attributes is available from an ongoing cardiovascular study on residents of the town of Framingham, Massachusetts. The classification goal is to predict whether the patient has a 10-year risk of future coronary heart disease (CHD).

"Predict if a patient will be at Risk of CHD."

How to Approach:

Approach the problem in three simple steps:

1. Pre- Processing
   - 1.1 View the data
   - 1.2 Inspect the data
   - 1.3 Clean the data
2. Performing exploratory data analysis (EDA)
   - 2.1 Univariate and Bivariate Analysis
   - 2.2 Feature Engineering and Dealing with Class Imbalance
   - 2.3 Normalizing the data
3. Training the Models
4. Observation and Conclusion

Pre-Processing

In just few simple steps:

1. View the data
2. Clean the data

   3.1 Remove/ replace data for missing values

   3.2 Remove duplicate values (here none)

   3.3 Change column names (if needed)

   3.4 Add derived features (sysBP, diaBP □ Pulse Pressure)

   3.5 Drop the redundant features

Pre-Processing (Cont.):

The dataset contains 15 features. They could be classified as demographics, behavioral, medical (history), medical condition (present), Risk to CHD (target variable).

Features:

Dependent Variable

Demographic:

1. Sex: male or female("M" or "F")
2. Age: Age of the patient
3. Education: Level of education. ("1", "2", "3" and "4".)

Behavioural:

4. is_smoking: whether the patient is a smoker ("YES" or "NO")
5. Cigs Per Day: number of cigarettes smoked on average in one day.

Medical(history):

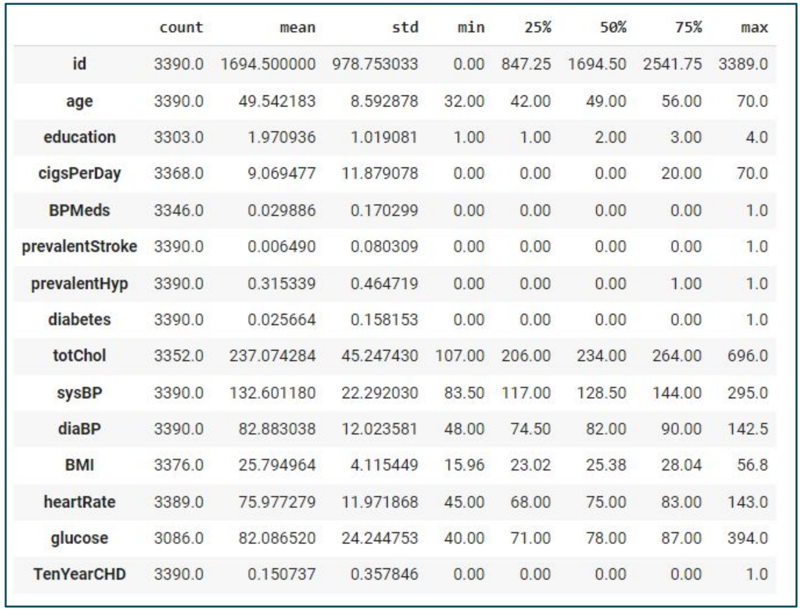
6. BP Meds: whether the patient was on blood pressure medication
7. Prevalent Stroke: whether the patient had previously had a stroke
8. Prevalent Hyp: whether the patient was hypertensive
9. Diabetes: whether the patient had diabetes

Medical(current):

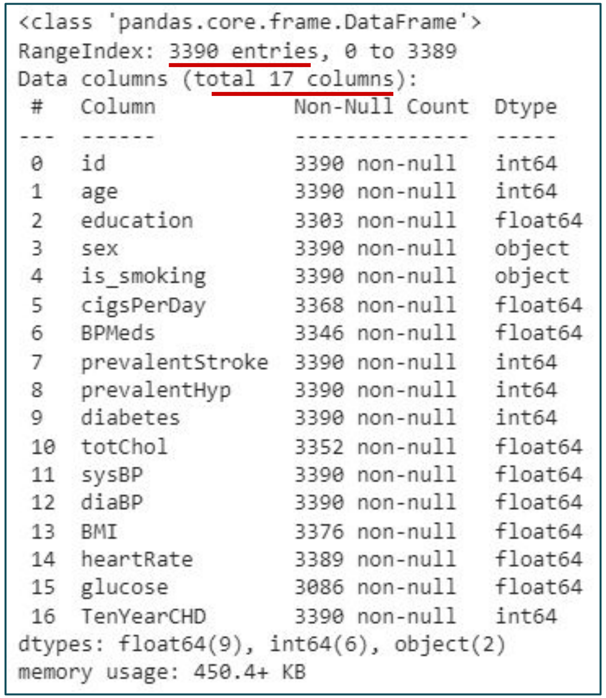
10. Tot Chol: total cholesterol level (Continuous)
11. Sys BP: systolic blood pressure (Continuous)
12. Dia BP: diastolic blood pressure (Continuous)
13. BMI: Body Mass Index (Continuous)
14. Heart Rate: heart rate (Continuous)
15. Glucose: glucose level (Continuous)
16. TenYearCHD: 10-year risk of coronary heart disease CHD

Pre-Processing (Cont.):

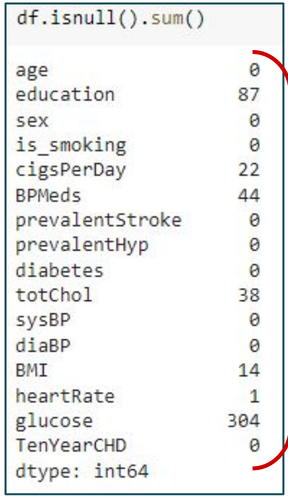
Using the basic commands like info, describe, unique, isnull, etc. we try to learn basic stats of the data:



	count	mean	std	min	25%	50%	75%	max
id	3390.0	1694.500000	978.753033	0.00	847.25	1694.50	2541.75	3389.0
age	3390.0	49.542183	8.592878	32.00	42.00	49.00	56.00	70.0
education	3303.0	1.970936	1.019081	1.00	1.00	2.00	3.00	4.0
cigsPerDay	3368.0	9.069477	11.879078	0.00	0.00	0.00	20.00	70.0
BPMeds	3346.0	0.029886	0.170299	0.00	0.00	0.00	0.00	1.0
prevalentStroke	3390.0	0.006490	0.080309	0.00	0.00	0.00	0.00	1.0
prevalentHyp	3390.0	0.315339	0.464719	0.00	0.00	0.00	1.00	1.0
diabetes	3390.0	0.025664	0.158153	0.00	0.00	0.00	0.00	1.0
totChol	3352.0	237.074284	45.247430	107.00	206.00	234.00	264.00	696.0
sysBP	3390.0	132.601180	22.292030	83.50	117.00	128.50	144.00	295.0
diaBP	3390.0	82.883038	12.023581	48.00	74.50	82.00	90.00	142.5
BMI	3376.0	25.794964	4.115449	15.96	23.02	25.38	28.04	56.8
heartRate	3389.0	75.977279	11.971868	45.00	68.00	75.00	83.00	143.0
glucose	3086.0	82.086520	24.244753	40.00	71.00	78.00	87.00	394.0
TenYearCHD	3390.0	0.150737	0.357846	0.00	0.00	0.00	0.00	1.0



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3390 entries, 0 to 3389
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   id               3390 non-null   int64  
 1   age              3390 non-null   int64  
 2   education        3303 non-null   float64
 3   sex              3390 non-null   object 
 4   is_smoking       3390 non-null   object 
 5   cigsPerDay       3368 non-null   float64
 6   BPMeds           3346 non-null   float64
 7   prevalentStroke  3390 non-null   int64  
 8   prevalentHyp     3390 non-null   int64  
 9   diabetes         3390 non-null   int64  
 10  totChol          3352 non-null   float64
 11  sysBP            3390 non-null   float64
 12  diaBP            3390 non-null   float64
 13  BMI              3376 non-null   float64
 14  heartRate        3389 non-null   float64
 15  glucose          3086 non-null   float64
 16  TenYearCHD       3390 non-null   int64  
dtypes: float64(9), int64(6), object(2)
memory usage: 450.4+ KB
```



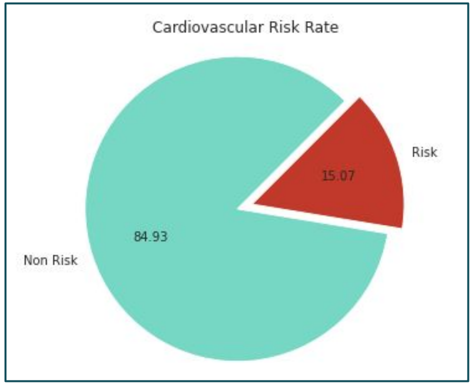
```
df.isnull().sum()

age                  0
education           87
sex                  0
is_smoking           0
cigsPerDay          22
BPMeds              44
prevalentStroke      0
prevalentHyp         0
diabetes             0
totChol             38
sysBP                0
diaBP                0
BMI                 14
heartRate            1
glucose            304
TenYearCHD           0
dtype: int64
```

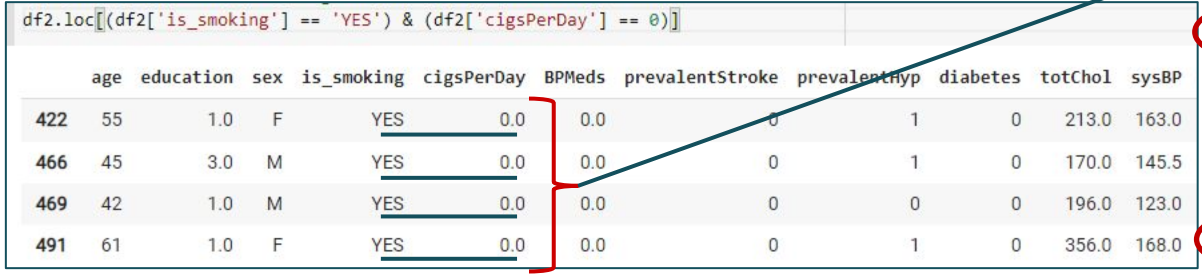
Missing values: 510

Pre-Processing (Cont.):

Summary about the data:

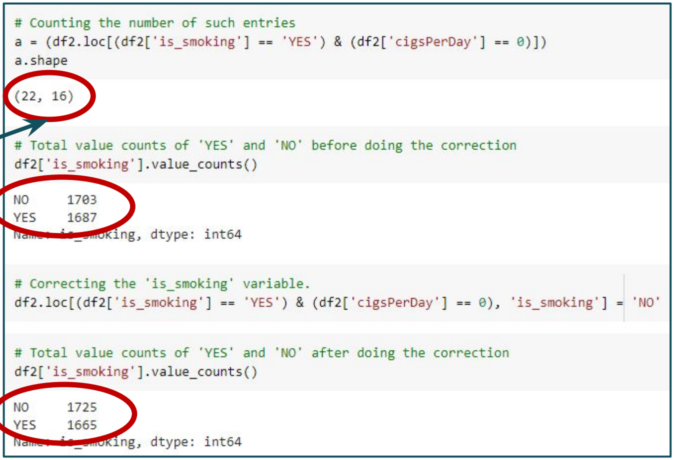
- Shape □ (3390,16) □ 16 features and 3390 data
- Dtypes □ object, float and integer. Here, the data types are already relevant. So, no change.
- Duplicate values □ None
- Missing values □ 510
- Checking the strength of minority class (Risk) □ 511/ 3390
- The dependent variable is categorical in nature.
- Dealing with irrelevant values. □ 22
- Looking at the percentage of classes, we can say that there is a huge class imbalance. We will need to deal with this before training the machine learning models.
- Label Encoding: Converting the data of 'sex' & 'is_smoking' feature from categorical to numerical where, M = 1, F = 0 and YES = 1, NO = 0, respectively.





```
df2.loc[(df2['is_smoking'] == 'YES') & (df2['cigsPerDay'] == 0)]
```

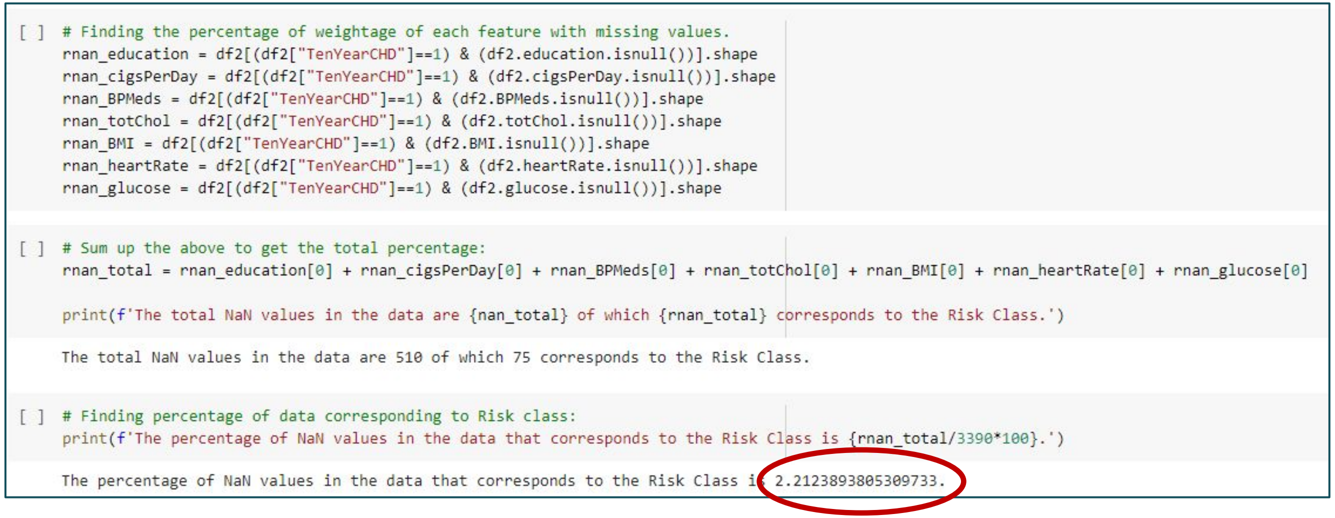
	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP
422	55	1.0	F	YES	0.0	0.0	0	1	0	213.0	163.0
466	45	3.0	M	YES	0.0	0.0	0	1	0	170.0	145.5
469	42	1.0	M	YES	0.0	0.0	0	0	0	196.0	123.0
491	61	1.0	F	YES	0.0	0.0	0	1	0	356.0	168.0



```
# Counting the number of such entries
a = (df2.loc[(df2['is_smoking'] == 'YES') & (df2['cigsPerDay'] == 0)])
a.shape

(22, 16)

# Total value counts of 'YES' and 'NO' before doing the correction
df2['is_smoking'].value_counts()

NO     1703
YES    1687
Name: is_smoking, dtype: int64

# Correcting the 'is_smoking' variable.
df2.loc[(df2['is_smoking'] == 'YES') & (df2['cigsPerDay'] == 0), 'is_smoking'] = 'NO'

# Total value counts of 'YES' and 'NO' after doing the correction
df2['is_smoking'].value_counts()

NO     1725
YES    1665
Name: is_smoking, dtype: int64
```

Pre-Processing (Cont.):

Now, we check that how much data corresponding to the missing values falls under the minority class of dependent variable. This is important because we want to know the amount of minority data being affected by this.



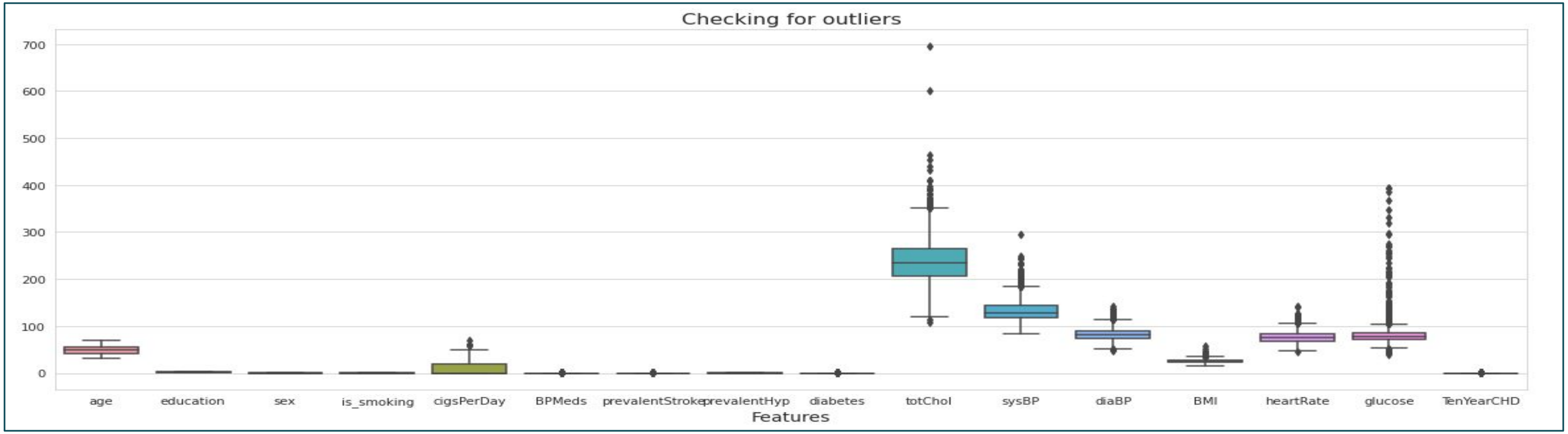
```
[ ]  # Finding the percentage of weightage of each feature with missing values.
     rnan_education = df2[(df2["TenYearCHD"]==1) & (df2.education.isnull())].shape
     rnan_cigsPerDay = df2[(df2["TenYearCHD"]==1) & (df2.cigsPerDay.isnull())].shape
     rnan_BPMeds = df2[(df2["TenYearCHD"]==1) & (df2.BPMeds.isnull())].shape
     rnan_totChol = df2[(df2["TenYearCHD"]==1) & (df2.totChol.isnull())].shape
     rnan_BMI = df2[(df2["TenYearCHD"]==1) & (df2.BMI.isnull())].shape
     rnan_heartRate = df2[(df2["TenYearCHD"]==1) & (df2.heartRate.isnull())].shape
     rnan_glucose = df2[(df2["TenYearCHD"]==1) & (df2.glucose.isnull())].shape
```

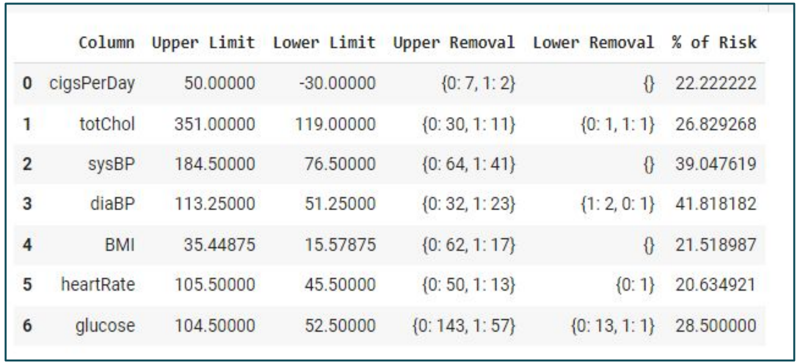
```
[ ]  # Sum up the above to get the total percentage:
     rnan_total = rnan_education[0] + rnan_cigsPerDay[0] + rnan_BPMeds[0] + rnan_totChol[0] + rnan_BMI[0] + rnan_heartRate[0] + rnan_glucose[0]

     print(f'The total NaN values in the data are {nan_total} of which {rnan_total} corresponds to the Risk Class.')

     The total NaN values in the data are 510 of which 75 corresponds to the Risk Class.
```

```
[ ]  # Finding percentage of data corresponding to Risk class:
     print(f'The percentage of NaN values in the data that corresponds to the Risk Class is {rnan_total/3390*100}.')

     The percentage of NaN values in the data that corresponds to the Risk Class is 2.2123893805309733.
```

This means that if we drop the NaN values, we will lose 2.21% of the data corresponding to the Risk class, which consists of just 15% of the entire data. Hence, dropping this might impact the final results. Hence, we will have to replace the data with some relevant values. For categorical variables we will use mode of the data while for numerical variables we will use median of the data.

Pre-Processing (Cont.):





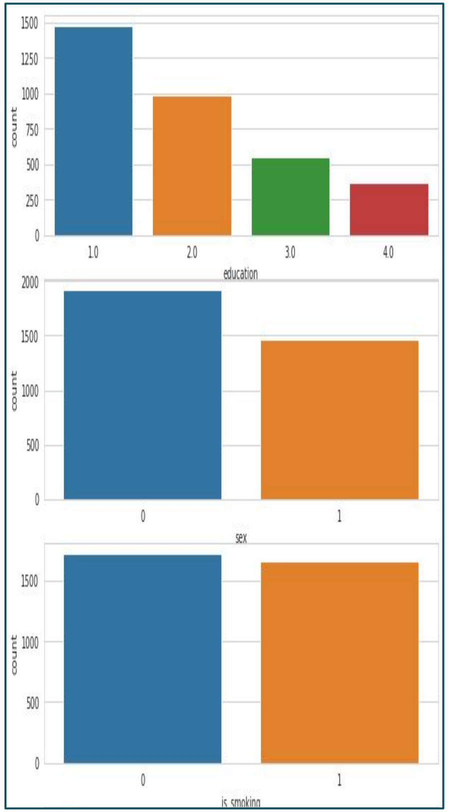
	Column	Upper Limit	Lower Limit	Upper Removal	Lower Removal	% of Risk
0	cigsPerDay	50.00000	-30.00000	{0: 7, 1: 2}	{}	22.222222
1	totChol	351.00000	119.00000	{0: 30, 1: 11}	{0: 1, 1: 1}	26.829268
2	sysBP	184.50000	76.50000	{0: 64, 1: 41}	{}	39.047619
3	diaBP	113.25000	51.25000	{0: 32, 1: 23}	{1: 2, 0: 1}	41.818182
4	BMI	35.44875	15.57875	{0: 62, 1: 17}	{}	21.518987
5	heartRate	105.50000	45.50000	{0: 50, 1: 13}	{0: 1}	20.634921
6	glucose	104.50000	52.50000	{0: 143, 1: 57}	{0: 13, 1: 1}	28.500000

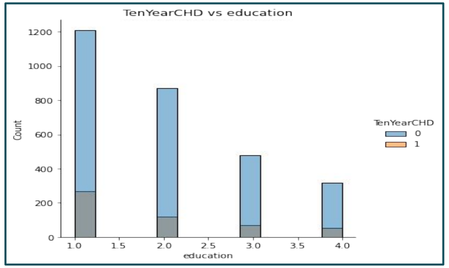
Before removing the outliers from all these features, we need to know the impact of each of it. Hence, we calculated the Risk of outlier removal for all the features.

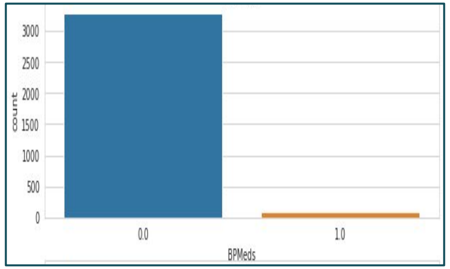
- Upper Removal shows the number of values lying above the Upper limit. The data is represented as a dictionary of 0,1, which means the number of points corresponding to No Risk and Risk class respectively.
- If we decide to remove all these outliers, then a minimum of 57 entries corresponding to the minority class ('Risk' - '1') will be lost.
- Since, there is already prevalent class imbalance, we cannot afford to lose any data corresponding to the minority class.

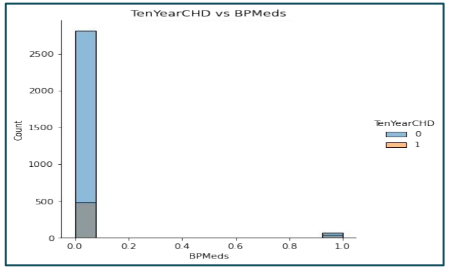
Performing the EDA

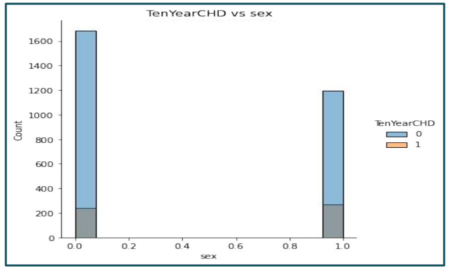
Plot of the Categorical Features:

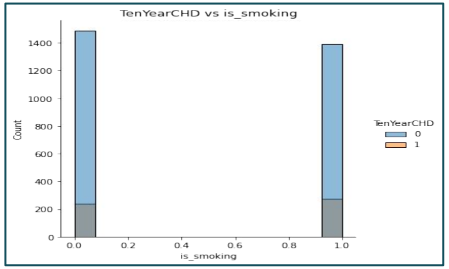








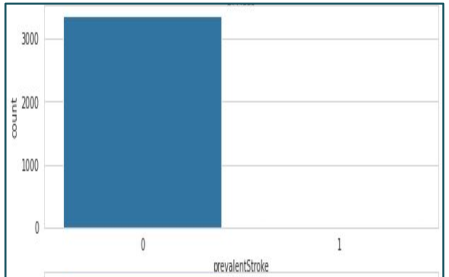


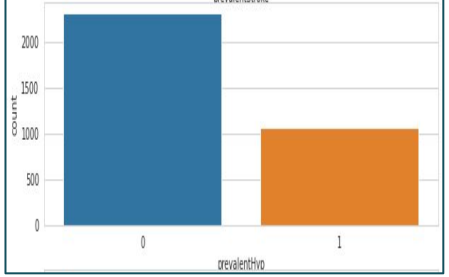


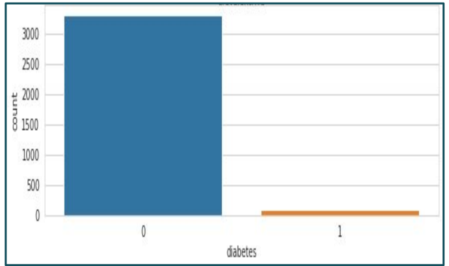
1. Majority of the people in the data belong to the 1st class of education which amounts to about 1500. As the class count increases, the number of people reduces.
2. Number of females in the data is more than men by nearly 500 counts.
3. The data contains nearly same amount of non-smokers a nd smokers.
4. Very few people are on BP medication. Above 3000 people are free from those medicines.

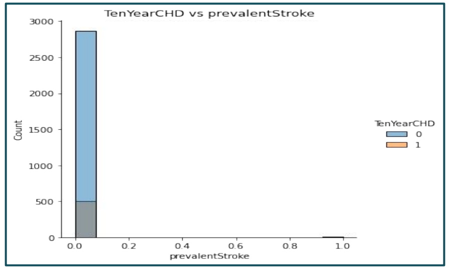
Performing the EDA (Cont.)

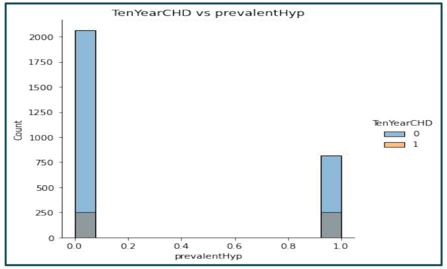
Plot of the Categorical Features:

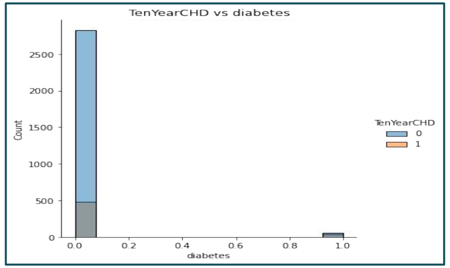








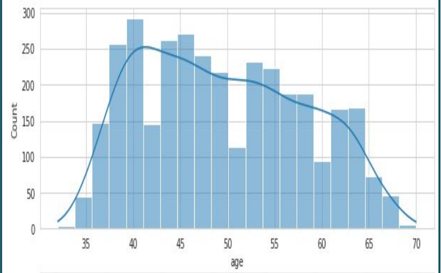


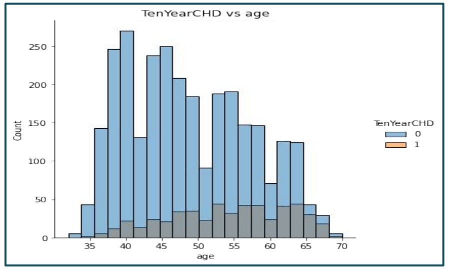


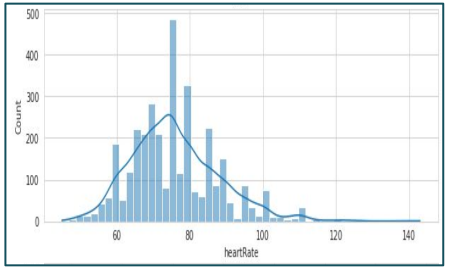
1. Majority of the people have not suffered a stroke previously.
2. About 1/3rd people have hypertension. This data amounts to more than 1000 counts.
3. Very few people have diabetes. Over 3000 people are free from diabetes.

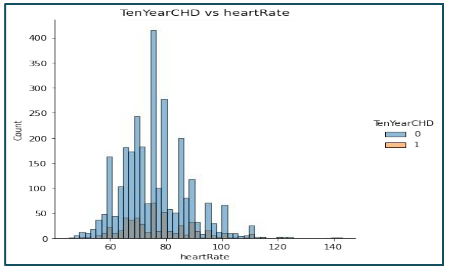
Performing the EDA (Cont.)

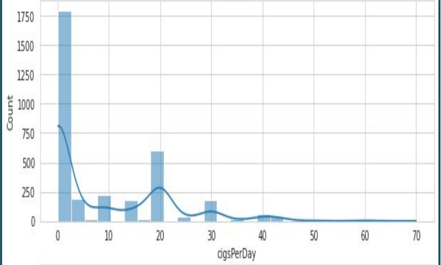
Plot of the Numerical Features:

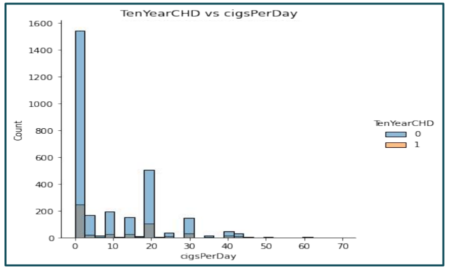


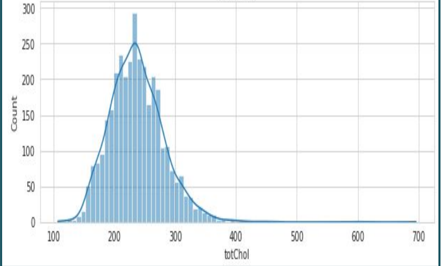


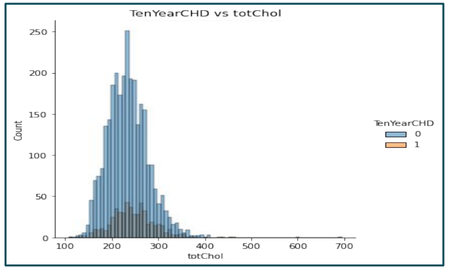








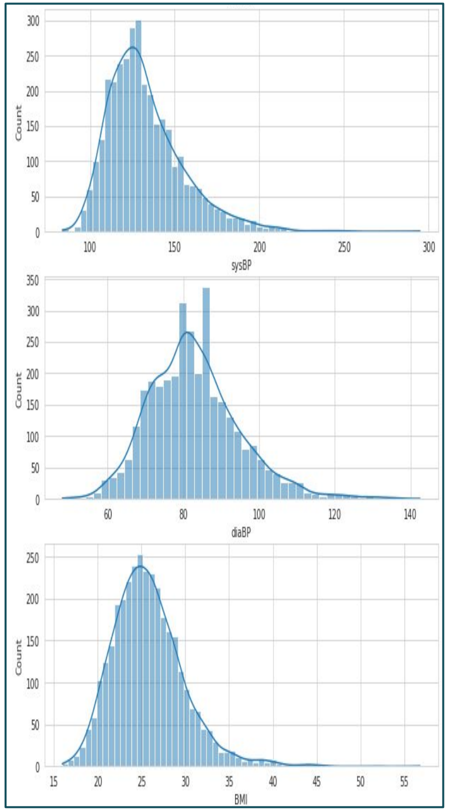


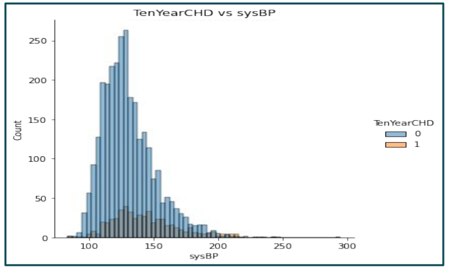


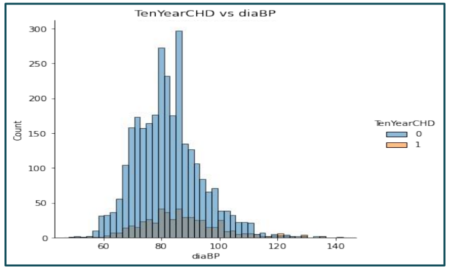
1. In the data, age ranges from 35 years to 70 years, of which majority people belong to the group of 40 - 45. Overall the graph is mostly normally distributed.
2. The data contains majority of the values of non smoker as zero cigarettes has the highest count. Other notable peaks are of 20 and 10 cigarettes per day.
3. The total Cholesterol follows a near normal distribution which ranges from 100 to 400 units, with peak nearing 250 units.
4. The Heart rate ranges from 35 to 110 with a peak at 75.

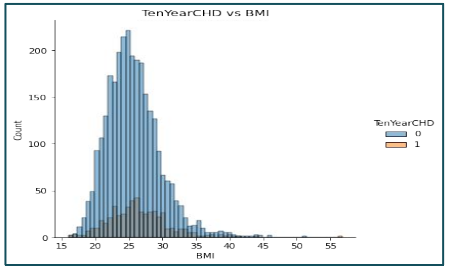
Performing the EDA (Cont.)

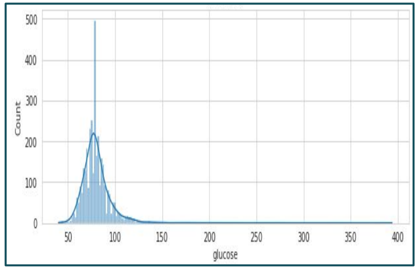
Plot of the Numerical Features:

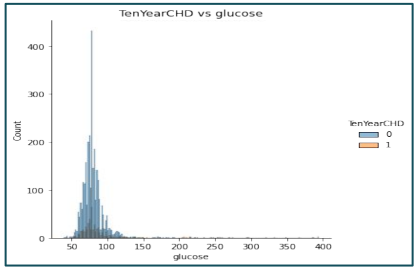








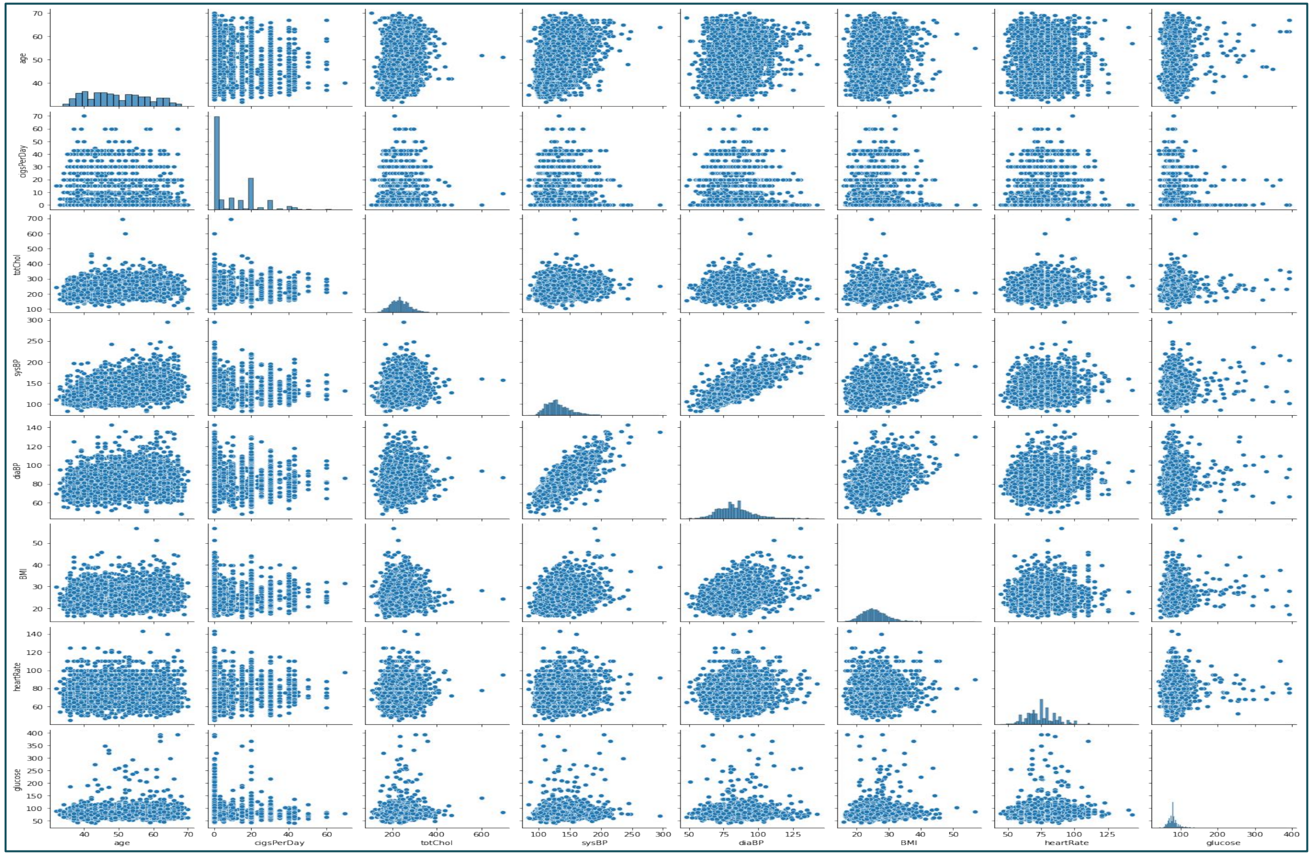




1. Systolic BP appears to be slightly left skewed with range 10 0 to 200 units.
2. Diastolic BP appears to be near normal distribution with range 60 to 120 units.
3. The BMI has a range of 16 to 40 with peak at 25. It follows normal distribution.
4. Glucose is left skewed graph with peak at about 75. It spans from 50 to 125.

Performing the EDA (Cont.)

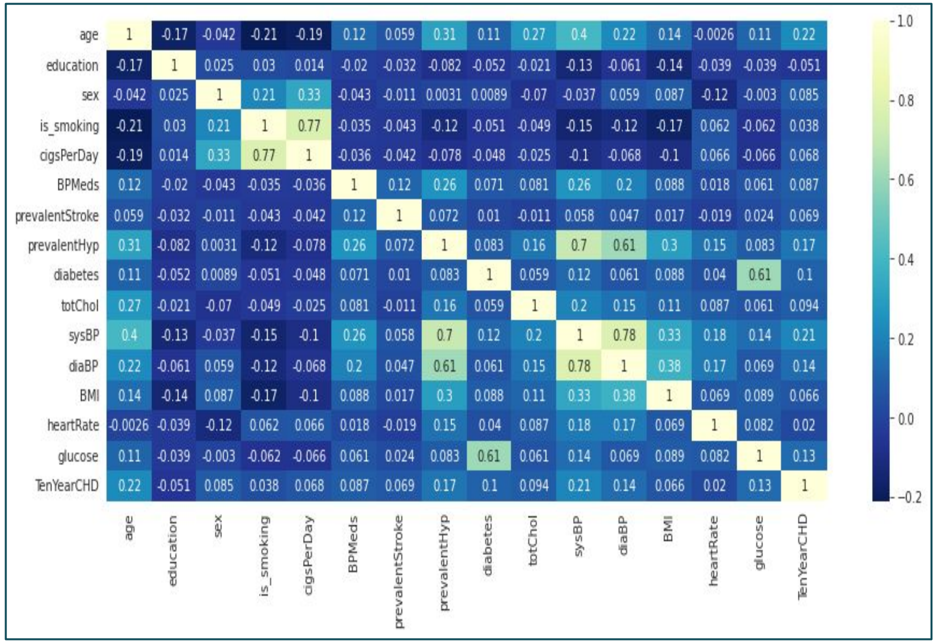
Bi-variate Analysis:



As cholesterol is low the systolic BP also remains low. Systolic BP and diastolic BP have a positive relationship. Most of the cases, less cholesterol means less glucose. Diastolic BP, systolic BP and BMI have a slight positive relationship.

Performing the EDA (Cont.)

Bi-variate Analysis:



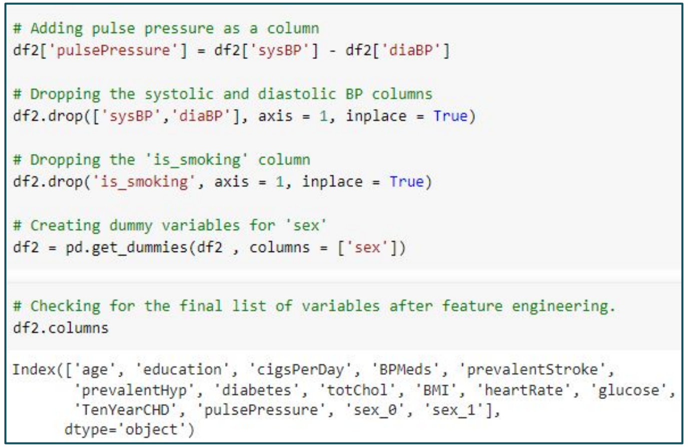
	age	education	sex	is_smoking	cigsPerDay	BPMeds	prevalentStroke	prevalentHyp	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
age	1	-0.17	-0.042	-0.21	-0.19	0.12	0.059	0.31	0.11	0.27	0.4	0.22	0.14	-0.0026	0.11	0.22
education	-0.17	1	0.025	0.03	0.014	-0.02	-0.032	-0.082	-0.052	-0.021	-0.13	-0.061	-0.14	-0.039	-0.039	-0.051
sex	-0.042	0.025	1	0.21	0.33	-0.043	-0.011	0.0031	0.0089	-0.07	-0.037	0.059	0.087	-0.12	-0.003	0.085
is_smoking	-0.21	0.03	0.21	1	0.77	-0.035	-0.043	-0.12	-0.051	-0.049	-0.15	-0.12	-0.17	0.062	-0.062	0.038
cigsPerDay	-0.19	0.014	0.33	0.77	1	-0.036	-0.042	-0.078	-0.048	-0.025	-0.1	-0.068	-0.1	0.066	-0.066	0.068
BPMeds	0.12	-0.02	-0.043	-0.035	-0.036	1	0.12	0.26	0.071	0.081	0.26	0.2	0.088	0.018	0.061	0.087
prevalentStroke	0.059	-0.032	-0.011	-0.043	-0.042	0.12	1	0.072	0.01	-0.011	0.058	0.047	0.017	-0.019	0.024	0.069
prevalentHyp	0.31	-0.082	0.0031	-0.12	-0.078	0.26	0.072	1	0.083	0.16	0.7	0.61	0.3	0.15	0.083	0.17
diabetes	0.11	-0.052	0.0089	-0.051	-0.048	0.071	0.01	0.083	1	0.059	0.12	0.061	0.088	0.04	0.61	0.1
totChol	0.27	-0.021	-0.07	-0.049	-0.025	0.081	-0.011	0.16	0.059	1	0.2	0.15	0.11	0.087	0.061	0.094
sysBP	0.4	-0.13	-0.037	-0.15	-0.1	0.26	0.058	0.7	0.12	0.2	1	0.78	0.33	0.18	0.14	0.21
diaBP	0.22	-0.061	0.059	-0.12	-0.068	0.2	0.047	0.61	0.061	0.15	0.78	1	0.38	0.17	0.069	0.14
BMI	0.14	-0.14	0.087	-0.17	-0.1	0.088	0.017	0.3	0.088	0.11	0.33	0.38	1	0.069	0.089	0.066
heartRate	-0.0026	-0.039	-0.12	0.062	0.066	0.018	-0.019	0.15	0.04	0.087	0.18	0.17	0.069	1	0.082	0.02
glucose	0.11	-0.039	-0.003	-0.062	-0.066	0.061	0.024	0.083	0.61	0.061	0.14	0.069	0.089	0.082	1	0.13
TenYearCHD	0.22	-0.051	0.085	0.038	0.068	0.087	0.069	0.17	0.1	0.094	0.21	0.14	0.066	0.02	0.13	1

From this heatmap we can say:

1. Variables with high correlation (i.e. above 50%):
   - "is_smoking" - "cigsPerDay" ==> This is the obvious correlation as the first one is a categorical variable while the second is a numerical one.
   - "prevalentHyp" - "sysBP" ==> Hypertension is related to Systolic Blood Pressure.
   - "prevalentHyp" - "diaBP" ==> Hypertension is related to Diastolic Blood Pressure.
   - "diabetes" - "glucose" ==> This is the obvious correlation as the first one is a categorical variable while the second is a numerical one.
   - "sysBP" - "diaBP" ==> These two are highly correlated as both of them are blood pressures.
2. Systolic and Diastolic Blood Pressure does influence hypertension and BMI.
3. Systolic BP and age have a positive correlation.
4. Variables such as age, prevalent hypertension, systolic BP, diastolic BP, and glucose have lower but positive correlation with the dependent variable. Hence, we can say that it has influence on the risk of heart disease.
5. Education is the only variable negatively correlated with the dependent variable.

Performing the EDA (Cont.)

Feature Engineering:



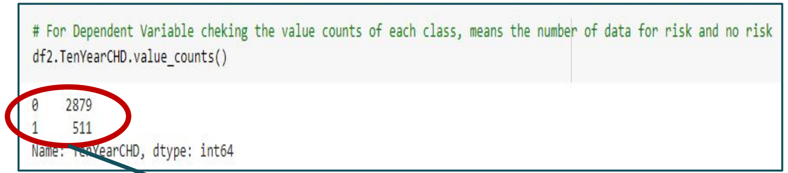
```
# Adding pulse pressure as a column
df2['pulsePressure'] = df2['sysBP'] - df2['diaBP']

# Dropping the systolic and diastolic BP columns
df2.drop(['sysBP','diaBP'], axis = 1, inplace = True)

# Dropping the 'is_smoking' column
df2.drop('is_smoking', axis = 1, inplace = True)

# Creating dummy variables for 'sex'
df2 = pd.get_dummies(df2 , columns = ['sex'])

# Checking for the final list of variables after feature engineering.
df2.columns

Index(['age', 'education', 'cigsPerDay', 'BPMeds', 'prevalentStroke',
       'prevalentHyp', 'diabetes', 'totChol', 'BMI', 'heartRate', 'glucose',
       'TenYearCHD', 'pulsePressure', 'sex_0', 'sex_1'],
      dtype='object')
```

Here, we will add derived variables and drop the highly correlated variables.

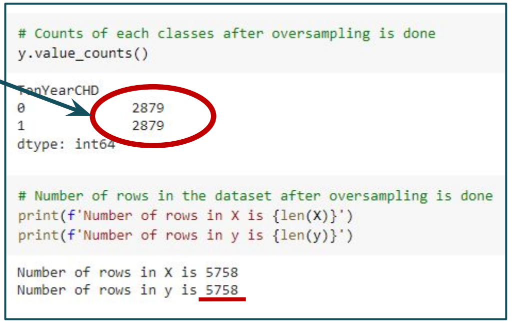
1. "sysBP" and "diaBP" are highly correlated. According to the article these two are linearly related variable hence we can convert them to a single variable "pulsePressure" and drop these two individual variables.
2. We will drop the 'is_smoking' variable, as it is categorical, it gives lesser information compared to the numerical variable "cigsPerDay".
3. "sex" is a categorical variable. So, we will create its dummy variable for the ease of analysis and We will drop this variable. Here, the dummy variables will be "sex_0" corresponding to 'Female' and "sex_1" corresponding to 'Male'.

Dealing with Class Imbalance:

There is a huge class imbalance in the dependent variable (target variable). Hence, we used 'SMOTE' (Synthetic Minority Oversampling Technique) to balance that for better model training.
This is a oversampling technique which will balance the classes of dependent variable.



```
# For Dependent Variable cheking the value counts of each class, means the number of data for risk and no risk
df2.TenYearCHD.value_counts()

0    2879
1     511
Name: TenYearCHD, dtype: int64
```



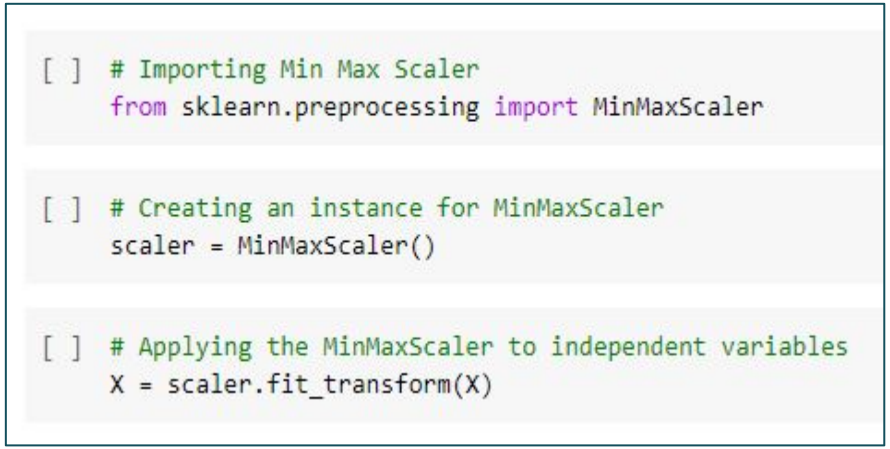
```
# Counts of each classes after oversampling is done
y.value_counts()

TenYearCHD
0        2879
1        2879
dtype: int64

# Number of rows in the dataset after oversampling is done
print(f'Number of rows in X is {len(X)}')
print(f'Number of rows in y is {len(y)}')

Number of rows in X is 5758
Number of rows in y is 5758
```

The original data consisted of 3390 rows. After oversampling the number of rows becomes 5758, hence total 2368 new rows were added.

Performing the EDA (Cont.)

Normalizing the data:

Looking at the count plot earlier for each independent feature, we observed that the majority of the features were skewed and not in same range. Hence we need to Normalize these features. We use **MinMaxScaler** to achieve the normalized data.



```
[ ]  # Importing Min Max Scaler
     from sklearn.preprocessing import MinMaxScaler
```

```
[ ]  # Creating an instance for MinMaxScaler
     scaler = MinMaxScaler()
```

```
[ ]  # Applying the MinMaxScaler to independent variables
     X = scaler.fit_transform(X)
```

With this EDA is completed.
Our data is ready to train the models.

Model Training:

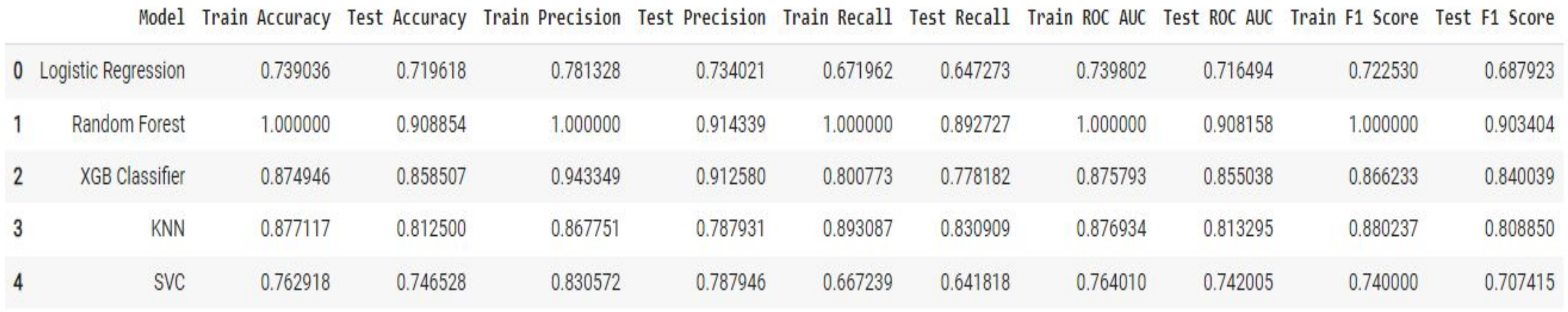
Split the data for Training and Testing: We split the data into 80 : 20 ratio.
Now we train the model and then test the same for various models. Based on that we will select the best suited model for our prediction.

The models used are:

- Logistic Regression
- Random Forest
- XGBoost Classifier
- KNN Classifier
- Support Vector Machine (SVM)

Model Training(Cont.)

Model Performance:



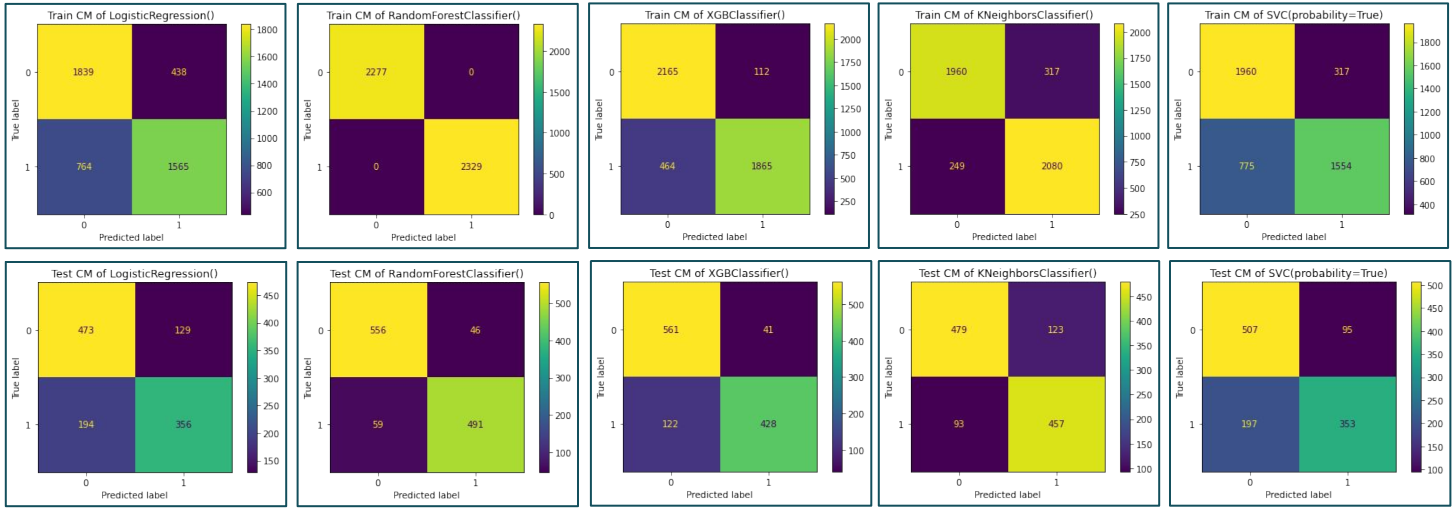
	Model	Train Accuracy	Test Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train ROC AUC	Test ROC AUC	Train F1 Score	Test F1 Score
0	Logistic Regression	0.739036	0.719618	0.781328	0.734021	0.671962	0.647273	0.739802	0.716494	0.722530	0.687923
1	Random Forest	1.000000	0.908854	1.000000	0.914339	1.000000	0.892727	1.000000	0.908158	1.000000	0.903404
2	XGB Classifier	0.874946	0.858507	0.943349	0.912580	0.800773	0.778182	0.875793	0.855038	0.866233	0.840039
3	KNN	0.877117	0.812500	0.867751	0.787931	0.893087	0.830909	0.876934	0.813295	0.880237	0.808850
4	SVC	0.762918	0.746528	0.830572	0.787946	0.667239	0.641818	0.764010	0.742005	0.740000	0.707415

From this we can say that:

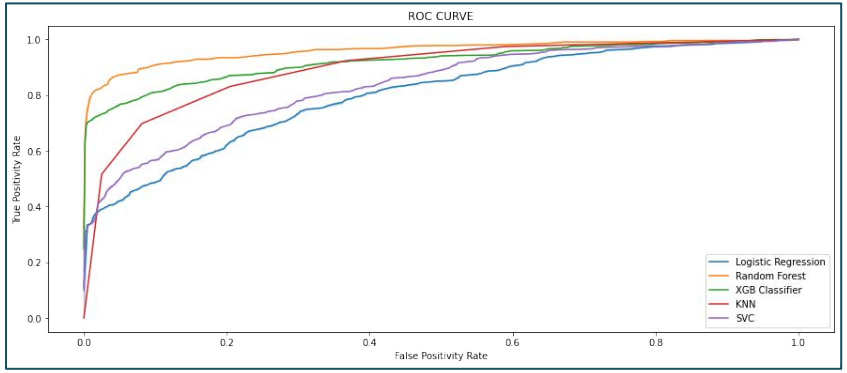
- RandomForest model out performs all the models when it comes to training. Also, the value of all the metrices are fairly high while testing the model. However, if we compare the train vs. test metrices, we see that somewhere, there is a slight occurrences of over fitting. We may consider hyperparameter tuning on this to improve the test results.
- The second best model appears to be the XG Boost Classifier. Presently it appears to have optimal fit as the test and train metrices have the same value. Hence, it is highly preferable to perform better after hyper parameter tuning.
- Logistic regression and SVC/ SVM are performing well, however, accuracy and recall for these are quite lower. Hence, we may not work upon improving these models.
- KNN does appear to have fine train metrices however, it does not quite really perform well for the test dataset. Hence, it also might have some degree of over fitting. So we will not consider it for further improvement.

Model Training (Cont.)

Confusion Matrix & ROC Curve:



Observation (ROC Curve):
Random Forest has a better performance compared to all the other models. The next one to follow Random Forest is XGB Classifier.



Observation (Confusion Matrix):
The performance of a model become critical to evaluate in case of classification problems when the minority classes are present in the data. Hence, we need to focus on the amount of False Negatives generated by the model. Looking at the count of False Negatives, the best model in the decreasing order of performances are Random Forest, KNN, XGB.

Model Training (Cont.)

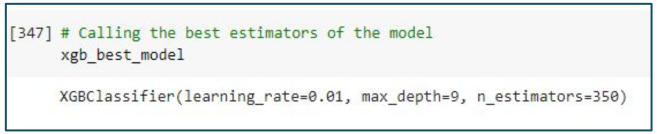
Hyperparameter Tuning:

From the model building section we can understand the best models are Random Forest and XGB Classifier. Between these two, Random Forest is overfitting to some extent compared to other models. XGB Classifier is the second best performer, hence this will be chosen for hyperparameter tuning.

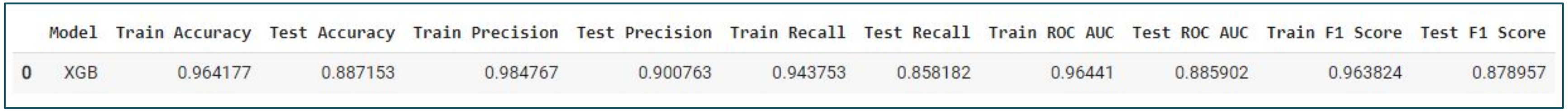
Here, we have used GridSearchCV, with hyperparameters as:

- Number of estimators: 300 & 350
- Max depth: 7, 8 & 9
- Learning Rate: 0.01 & 0.001

After applying this we find the best parameters as:

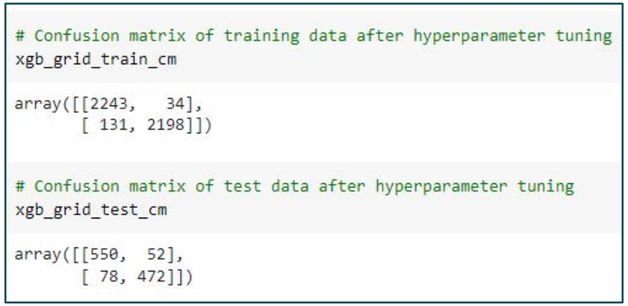


```
[347] # Calling the best estimators of the model
      xgb_best_model

      XGBClassifier(learning_rate=0.01, max_depth=9, n_estimators=350)
```



	Model	Train Accuracy	Test Accuracy	Train Precision	Test Precision	Train Recall	Test Recall	Train ROC AUC	Test ROC AUC	Train F1 Score	Test F1 Score
0	XGB	0.964177	0.887153	0.984767	0.900763	0.943753	0.858182	0.96441	0.885902	0.963824	0.878957

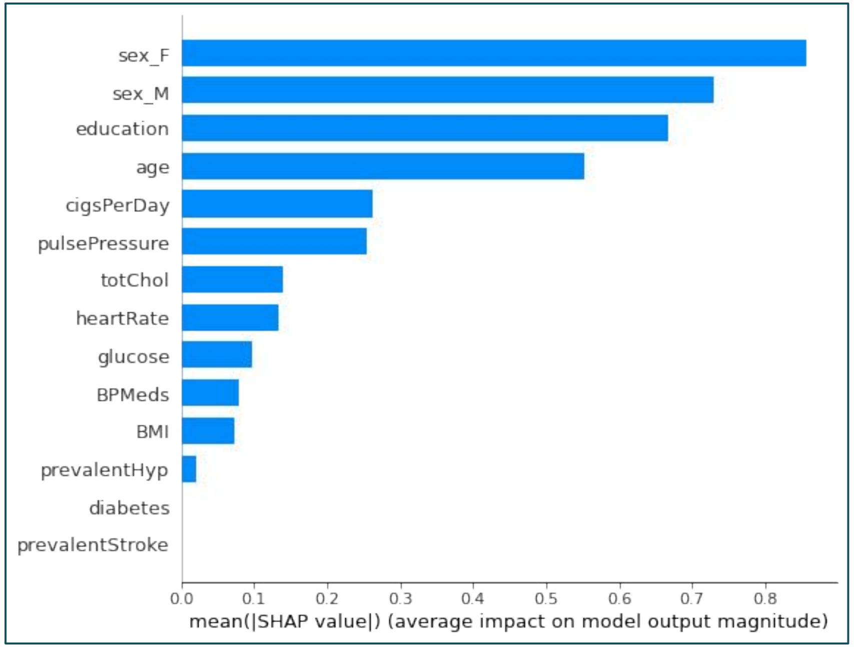
As we can see the hyperparameter tuning has improved the model from the base XGBoost Classifier especially for recall, the parameter we are focusing on.
However, this still doesn't give the model explainability. For this we need to check the feature importance.



```
# Confusion matrix of training data after hyperparameter tuning
xgb_grid_train_cm

array([[2243,   34],
       [ 131, 2198]])

# Confusion matrix of test data after hyperparameter tuning
xgb_grid_test_cm

array([[550,  52],
       [ 78, 472]])
```

Model Training (Cont.)

Feature Importance:



1. Gender, education and age are the main features which help us classify the risk of CHD.
2. Looking at the features related to medical condition or medical history, we could say that they have about 10% influence on the Risk of CHD; which is quite less or insignificant.
3. Cigarettes per day and Blood Pressure (difference of SysBP and DisBP) influences the CHD variable by about 30%.

Conclusions:

1. Keeping the accuracy and performance of Confusion Matrix, performance metrices value (Accuracy, Precision, Recall) and ROC Curve we could say that XGBoost model performed the best as it did not overfit the data as much as done by the Random Forest model.

2. We did hyperparameter tuning on XGB to improve its performance. The overall model performance was not improved, however, the Recall value did improve. Which is an indication that we succeeded in reducing the False Negative counts.

3. Later on we use SHAP method to know the feature importance. We could deduce that gender, education and age appear to be the major influencers. Compared to other medical features, cigarettes per day and Blood Pressure played a significant role with about 30% influence on the dependent variable. Here, it is a bit confusing to relate education with health. However, if enough data is available we could deduce if the students of certain educational background do fall victim of this. To get a better understanding of the dependence of this feature on the Risk factor, we must consult a domain expert and gather more data. This could be another research in itself.

4. We could further improve the model performance by further fine tuning the hyperparameters where we could seek help from an expert with excellent domain knowledge.

Thank you

References of Images:

1. Vector image and graphics of the muse: shutterstock.com.