Lab Analysis Report

Cardiovascular Disease Detection **Akarsh Chandrashekar**

Problem Statement:

Health issues are a major concern in the 21st century. The aim of this project is to determine which variables are related to the disease and use different machine learning models to predict whether the patient has cardiovascular disease or not. using features like Systolic blood pressure, Diastolic blood pressure, height, weight and 8 other features.

Data Description:

The dataset has 1000 records(including cardio-train and cardio-validation) and 13 features(including target variable) with missing values. The available features are as follows:

1. **Id| int:**
   Unique id of each individual
2. **age | int (days):**
   Age of person in days
3. **height | int (cm) :**
   Height of person in cm
4. **weight | float (kg) :**
   Weight of person in kg
5. **gender | categorical:**
   Gender as male or female
6. **ap_hi | int:**
   Systolic blood pressure of a person
7. **ap_lo | int**:
   Diastolic blood pressure of a person
8. **cholesterol | Normal, Above normal, High:**
   Cholesterol of a person as Normal, Above normal and high
9. **gluc | Normal, Above normal, High:**
   Glucose level as Normal, Above normal and high
10. **smoke | binary:**
    Does the person smoke or not
11. **alco | binary:**
    Does the person has alcohol intake or not
12. **active | binary:**
    Does the person indulge in some physical activity or not

13. **<u>cardio | binary:</u>**

Presence or absence of cardiovascular disease

The dataset also contains missing values which will be discussed later.

<u>Data Analysis and Visualization</u>

<u>Training data:</u>

Categorical features: gender, cholesterol, gluc, smoke, alco, active

Continuous features: id, age, height, weight, ap_hi, ap_lo

```
print(Table)
```

Feature Name	Category
id	nominal
age	continous
gender	categorical
height	continous
weight	continous
ap_hi	continous
ap_lo	continous
cholesterol	Ordinal
gluc	Ordinal
smoke	categorical
alco	categorical
active	categorical
cardio	categorical

feature categories

The dataset has missing values

```
df_train.isnull().sum()
```

```
id               0
age            165
gender         171
height         302
weight         164
ap_hi          153
ap_lo          168
cholesterol    167
gluc           167
smoke          174
alco           165
active         157
cardio           0
dtype: int64
```

Null values

We can notice that height has the most missing values with the lowest in active.

Description of data frame to get a better understanding of mean and standard deviations.

```
df.describe()
```

	id	age	height	weight	ap_hi	ap_lo	smoke	alco	active	cardio
count	500.000000	335.000000	198.000000	336.000000	347.000000	332.000000	326.000000	335.000000	343.000000	500.000000
mean	50279.916000	53.399689	163.934343	74.347321	128.685879	90.060241	0.092025	0.065672	0.813411	0.502000
std	29913.623631	6.758089	8.258559	14.335964	18.490176	87.396945	0.289505	0.248078	0.390150	0.500497
min	38.000000	39.271233	120.000000	45.000000	12.000000	60.000000	0.000000	0.000000	0.000000	0.000000
25%	23446.500000	49.283562	159.250000	65.000000	120.000000	80.000000	0.000000	0.000000	1.000000	0.000000
50%	51913.500000	54.024658	165.000000	72.000000	120.000000	80.000000	0.000000	0.000000	1.000000	1.000000
75%	78656.000000	59.171233	168.000000	82.000000	140.000000	90.000000	0.000000	0.000000	1.000000	1.000000
max	99662.000000	64.326027	187.000000	155.000000	190.000000	1000.000000	1.000000	1.000000	1.000000	1.000000

Table 1: Basic information about data

Oldest and youngest patient:

```
df_train['age_in_years'].describe()
```

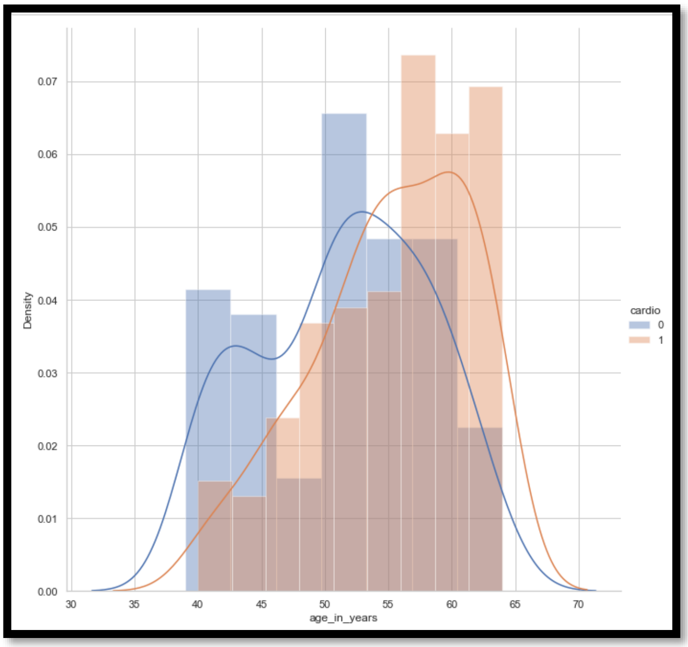
```
count    335.000000
mean      53.388060
std        6.750815
min       39.000000
25%       49.000000
50%       54.000000
75%       59.000000
max       64.000000
Name: age_in_years, dtype: float64
```

Oldest person and youngest person

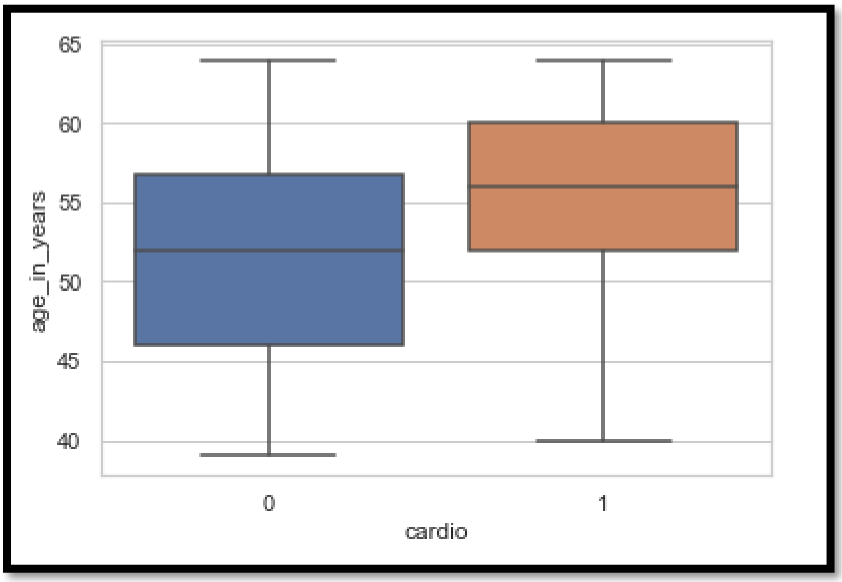
Data Visualization:

Let us analyse different features and check how they are related to our target variable.

1. **Relationship between the cardio and ages:**



Age vs Cardio kde plot



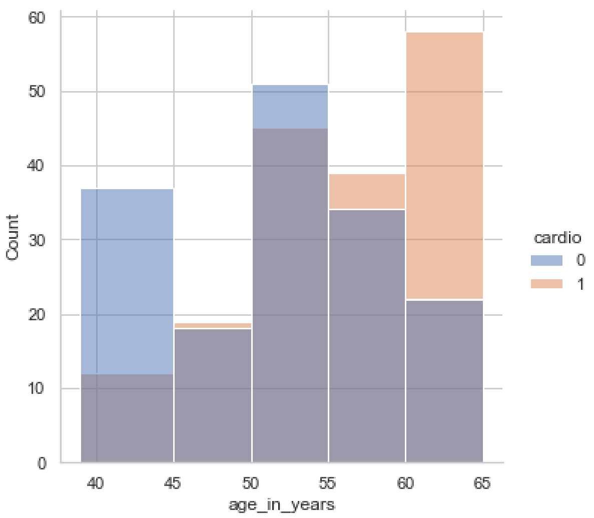
In the above figure, I've used seaborn boxplot to understand the relationship between age and cardio. The blue boxplot and corresponding whisker are age groups who do not have cardio-vascular disease whereas orange plot shows ages that have cardiovascular diseases.

From the plot, it is clearly visible that people 50% of people who do not have cardiovascular disease are below 51 years and 75% are below 57 years of age. 50% of people who have cardiovascular disease are above 55 years of age This gives us an understanding that as you get older, there is a higher chance you might have cardio-vascular disease. But since there is overlap this feature alone is not sufficient for classification

2 Age groups with lowest rate of cardiovascular disease

```
sns.displot(df_train, x="age_in_years", hue="cardio",bins=[39,45,50,55,60,65])
```

<seaborn.axisgrid.FacetGrid at 0x1e7e0761ef0>



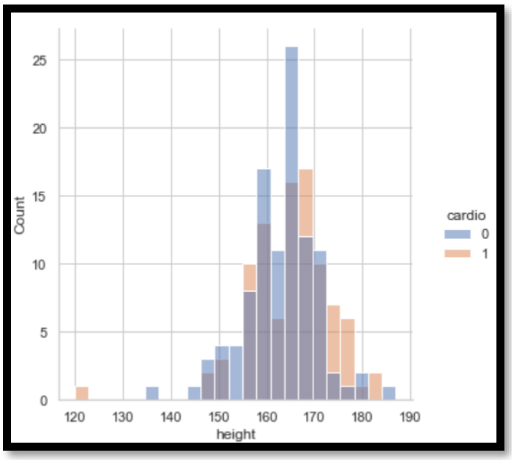
The rate of not having a cardio-vascular disease is largest in age groups of 39-45 years of age. We see that as the age of a person increases, the chances of not having cardiovascular disease decreases

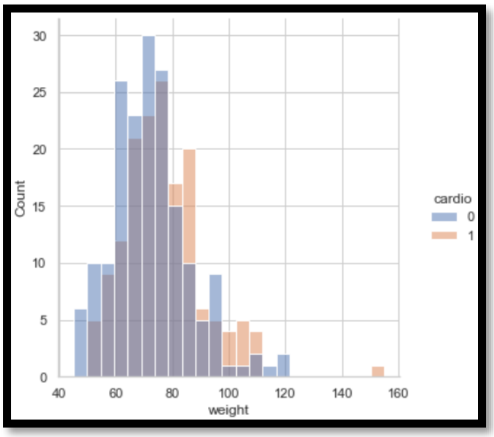
Gender Attribute:-

Men/Woman for different samples	Ratio
Gender Ratio Total Samples	2.074766355140187
Gender Ratio Healthy Samples	2.3333333333333335
Gender Ratio Disease Samples	1.8644067796610169

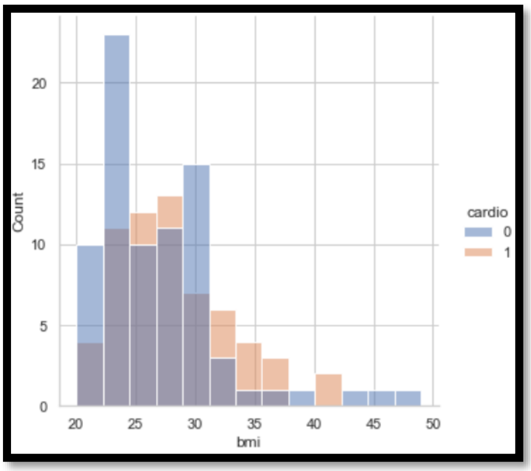
The Men to Women ratio for samples having/not having cardiovascular disease remains similar to the ratio of men and woman in the dataset no specific correlation can be found just comparing these features.

Height and Weight :-



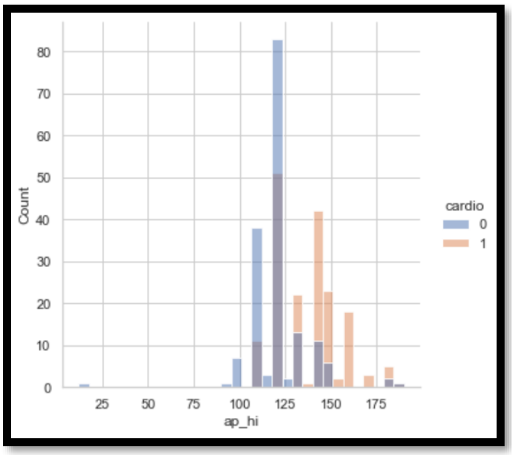


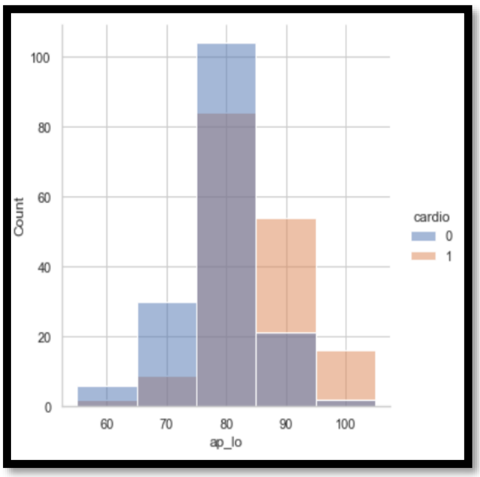
No specific pattern can be found hence combine them to get BMI feature



The chance of not having the disease is more when BMI is low and increase when BMI is high.

Blood Pressure





Looking at the above plots we can infer that as systolic or dsistolic blood pressure increase there is a higher chance of having cardio-vascular disease

Cholesterol:-

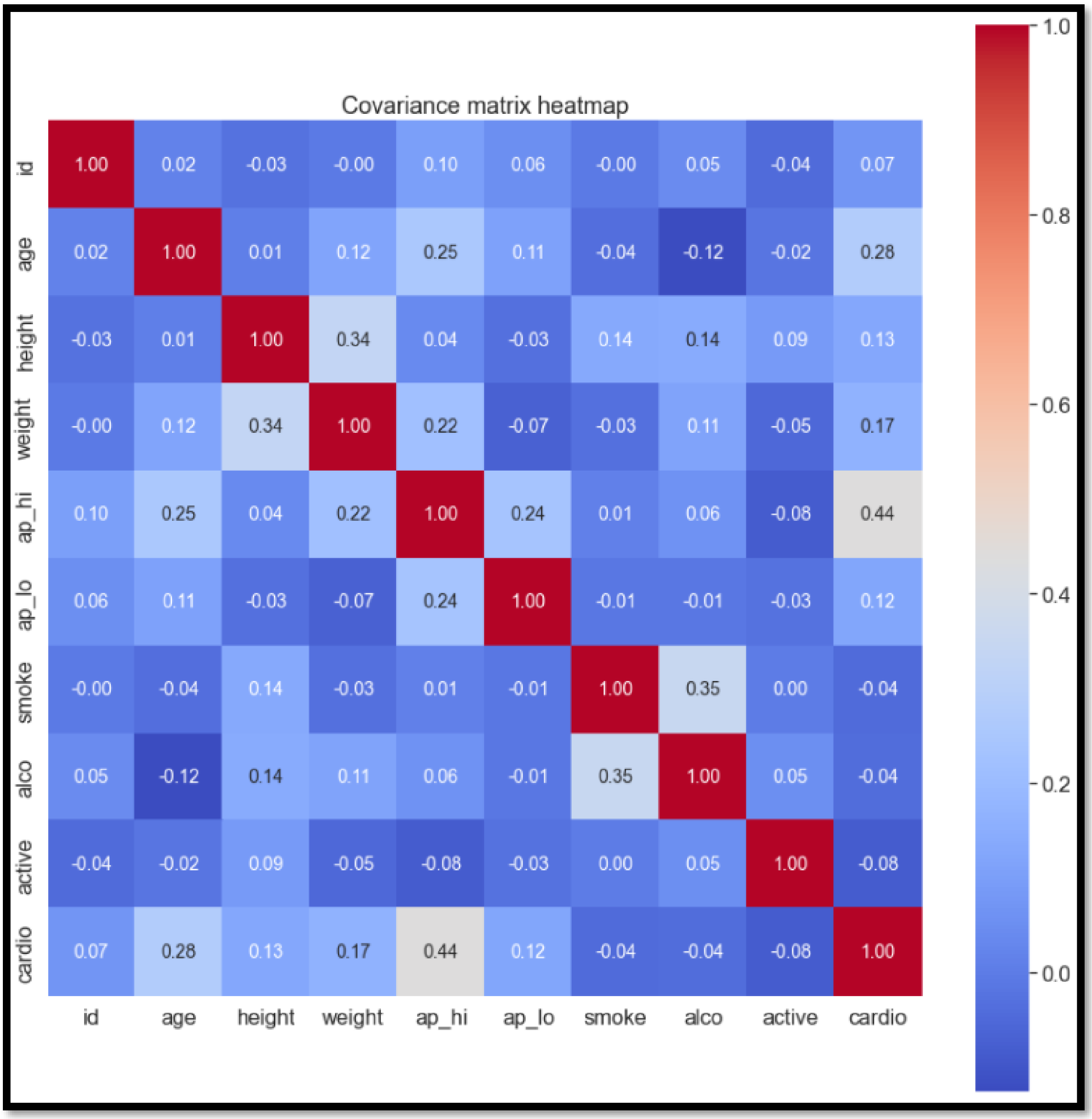
label	Probability
P(Disease/Cholestrol Normal)	0.4074074074074074
P(Healthy/Cholestrol Normal)	0.5925925925925926
P(Disease/Cholestrol Above Normal)	0.6415094339622641
P(Healthy/Cholestrol Above Normal)	0.3584905660377358
P(Disease/Cholestrol High)	0.7837837837837838
P(Healthy/Cholestrol High)	0.21621621621621623

Looking at the probability values we can say we can clearly see positive correlation between higher cholesterol and higher probabilty of having the disease.

Heatmap:

Heatmap gives the correlation between different attributes and can help us understand if a particular feature will have an impact on the target variable. The correlation values close to 1 or -1 mean there is a correlation between two variables and correlation value close to 0 means no correlation.

Plotted using seaborn heatmap



Heatmap of training data

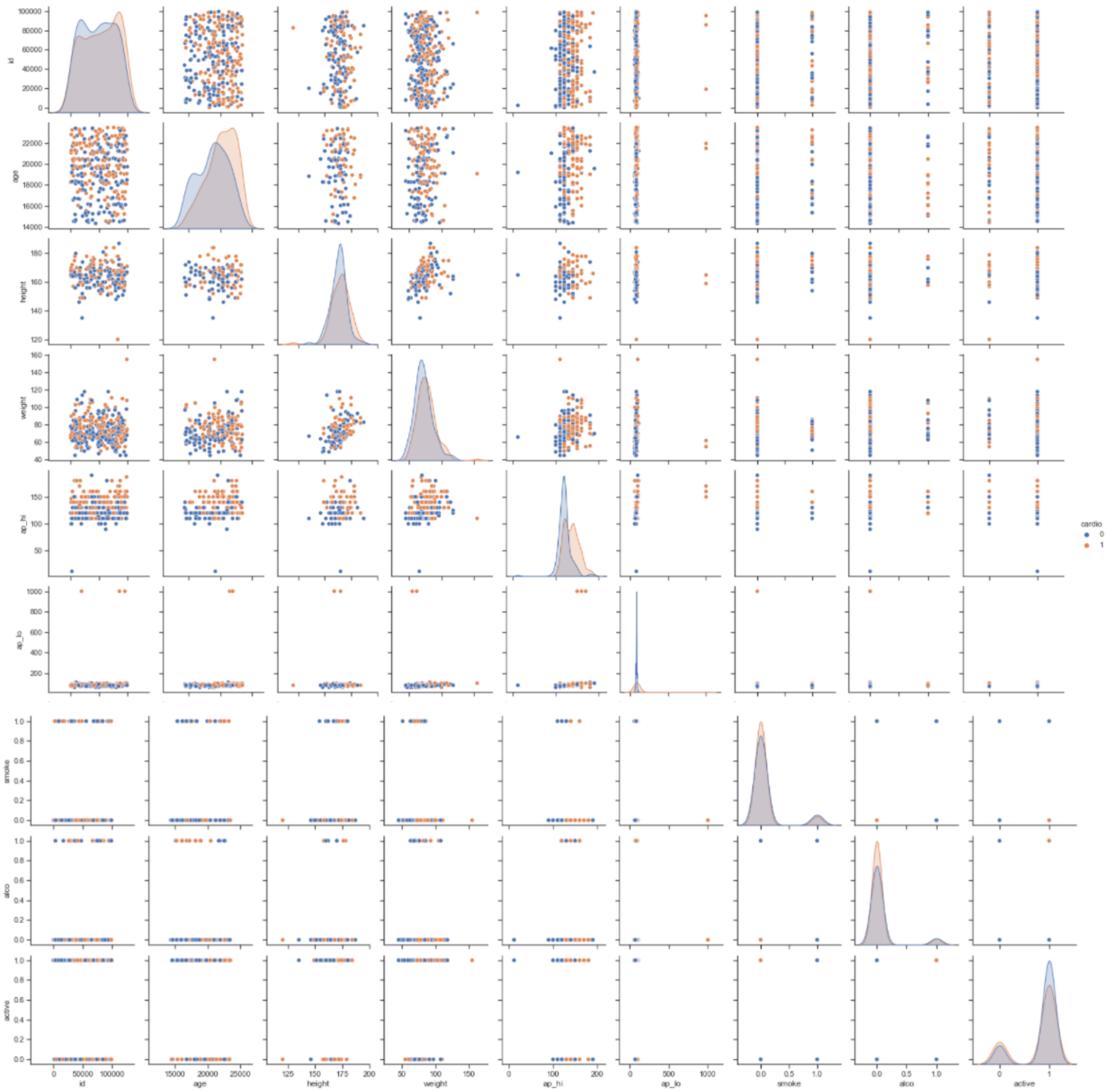
age, weight, ap_hi, ap_lo, cholesterol, gluc are the features that have a correlation value with cardio more than 0.1. These features can have a higher impact on prediction.

5. Scatter Matrix

Scatter matrix gives scatter plots of all the features with respect to each other. The diagonal plots give a kde for that particular feature.

In the plot below, we can see that there isn't much change of weight with respect to age. People generally don't gain much weight as they get older.

```
<seaborn.axisgrid.PairGrid at 0x1e7df679dd8>
```



: Scatter plot of data

Finding Missing Values

The dataset has missing values:

```
df_train.isnull().sum()

id              0
age           165
gender        171
height        302
weight        164
ap_hi         153
ap_lo         168
cholesterol   167
gluc          167
smoke         174
alco          165
active        157
cardio          0
dtype: int64
```

figure 12:missing values in each column

<u>Plotting a heatmap for missing values:</u>

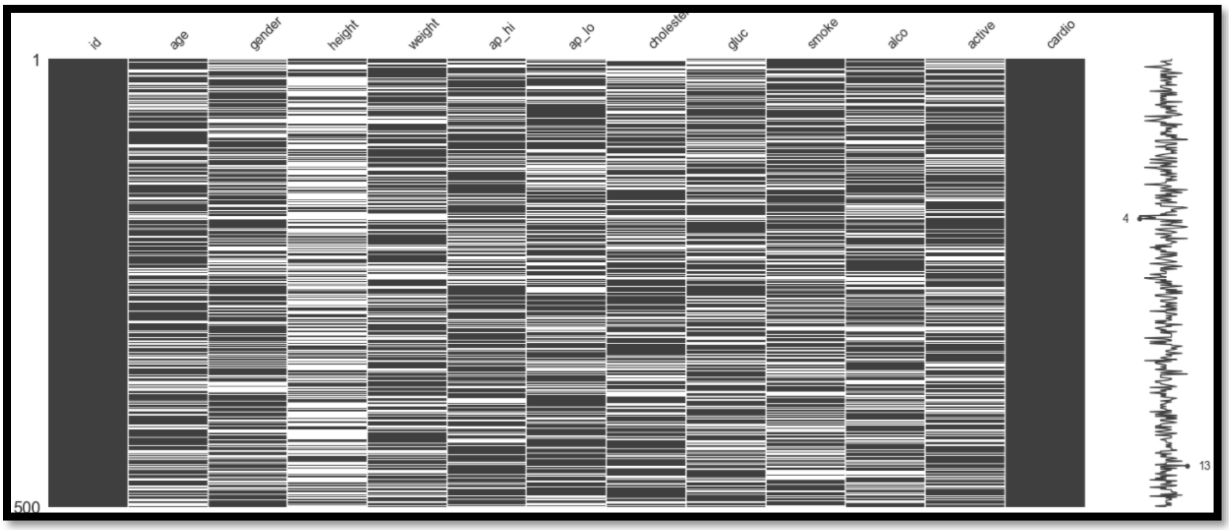


figure 13:Missing values heatmap(1)

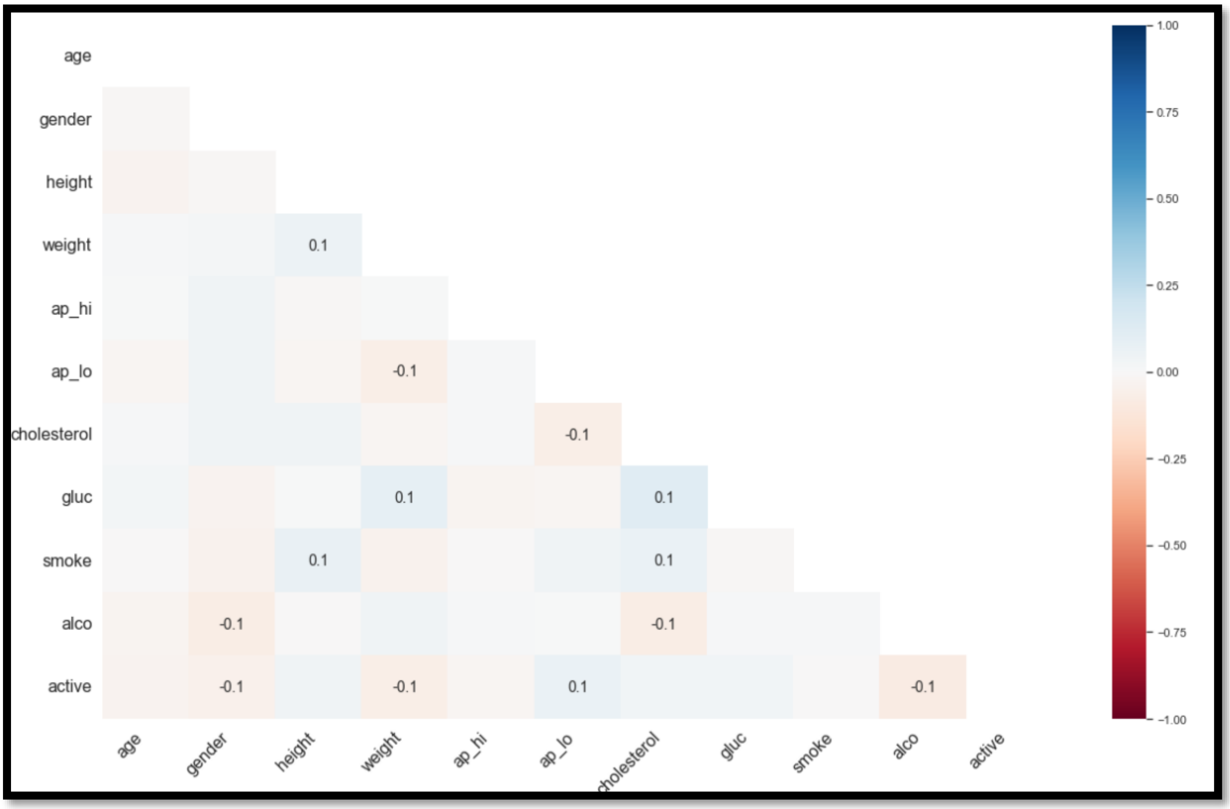


figure 14:Missing values heatmap(2)

Observing the above figure and missing value matrix we can deduce that there is no specific pattern among the missing values this is also corroborated by the missing value correlation heatmap where we don't see any large positive or negative correlation.

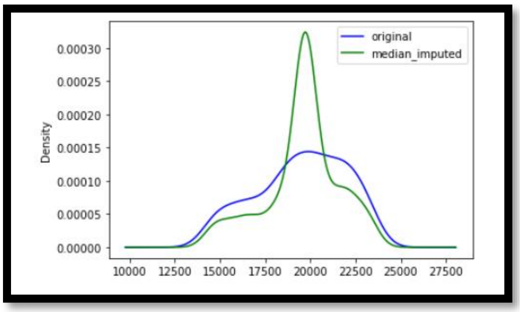
<u>Techniques to fill missing values:</u>

1. **Drop all Nan values**
2. **Replace all Nan values by zero**
3. **Replace all Nan values with mean**
4. **Replace Nan values with Random imputer**
5. **Using Iterative imputer**
6. **:Using KNN imputer**

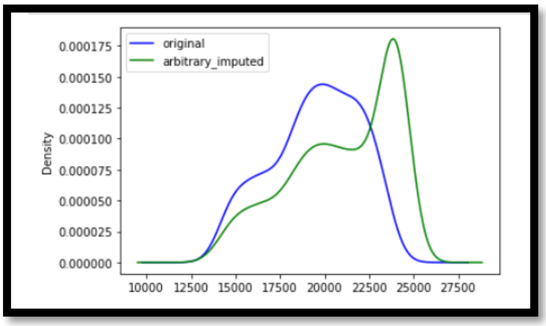
We can say that a strategy is good if it worked best when it preserves the distribution we can see this difference of distribution in the below curves where blue is original and green is after imputation.

We can see that imputing by one single number whether it is mean or arbitrary number this skews the distribution whereas Random no imputation preserves the distribution well here we take n random numbers and impute the missing values.

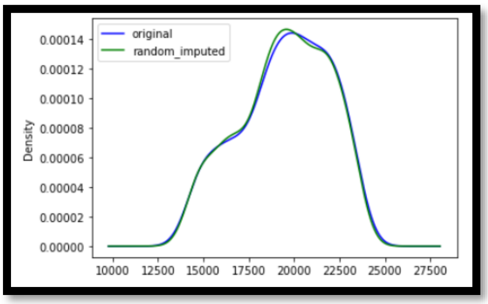
Hence going forward I considered Random no imputation and model imputation such as iterative and knn.



Median imputed



arbitrary imputed



Random no imputed

Outlier removal Techniques:-

1) Local Outlier Factor(LOF)
2) Remove values based on IQR

The reason behind choosing these two combination is that LOF considers the instance of the point which includes all the features whereas removing values considering IQR involves removing values less than q1-1.5*IQR or q3+1.5*IQR it is based on the distribution of feature rather than the instance hence for variability I chose these two outlier removal techniques.

Scaling:-

1) Standard scaler.
2) Minmax scaler.

After imputation and outlier removal the data is scaled using the above mentioned approaches

Feature Engineering:

Dropped the column 'id' as it is unique to every data entry and will not affect the decision-making process.

Convert Age in days to years.

Working on Test dataset(cardio-validation.csv):

The testing data needs to be refined so that the distribution is the same as our training dataset. So I have done the following computations:

1. Dropped 'id' column
2. Converting age from days to years
3. Mapping categorical string features into categorical numerical
4. Filling Missing values
5. Outlier removal
6. Scaling

Train-Tst data:

Now we have the perfect data we want to use for our model.

1) **Random imputed standard scaled data**

```
#df_train_ranna_stdsc,y_train
#df_test_ranna_stdsc,y_test
```

2) **Random imputed IQR outlier removed standard scaled data**

```
#df_train_ranna_out_stdsc,y_train
#df_test_ranna_out_stdsc,y_test
```

3) **Random imputed LOR outlier removed standard scaled data**

```
#df_train_ranna_outlof_stdsc,y_train_ranna_outlof
#df_test_ranna_outlof_stdsc,y_test_ranna_outlof
```

Same exists for min max scaled data

```
#df_train_ranna_minmaxsc,y_train
#df_test_ranna_minmaxsc,y_test

#df_train_ranna_out_minmaxsc,y_train
#df_test_ranna_out_minmaxsc,y_test

#df_train_ranna_outlof_minmaxsc,y_train_lof
#df_test_ranna_outlof_minmaxsc,y_test_lof
```

Same no exists for iterative imputed and knn imputed data

```
#df_train_itena_stdsc,y_train
#df_test_itena_stdsc,y_test

#df_train_itena_out_stdsc,y_train
#df_test_itena_out_stdsc,y_test

#df_train_itena_outlof_stdsc,y_train_itena_lof
#df_test_itena_outlof_stdsc,y_test_itena_lof

#df_train_itena_minmaxsc,y_train
#df_test_itena_minmaxsc,y_test

#df_train_itena_out_minmaxsc,y_train
#df_test_itena_out_minmaxsc,y_test

#df_train_itena_outlof_minmaxsc,y_train_itena_lof
#df_test_itena_outlof_minmaxsc,y_test_itena_lof
```

```
#df_train_3_knnna_stdsc,y_train
#df_test_3_knnna_stdsc,y_test

#df_train_3_knnna_out_stdsc,y_train
#df_test_3_knnna_out_stdsc,y_test

#df_train_3_knnna_outlof_stdsc,y_train_knn3_lof
#df_test_3_knnna_outlof_stdsc,y_test_knn3_lof

#df_train_3_knnna_minmaxsc,y_train
#df_test_3_knnna_minmaxsc,y_test

#df_train_3_knnna_out_minmaxsc,y_train
#df_test_3_knnna_out_minmaxsc,y_test

#df_train_3_knnna_outlof_minmaxsc,y_train_knn3_lof
#df_test_3_knnna_outlof_minmaxsc,y_test_knn3_lof
```

18 sets of train and test data.

Models:

Below are all the models and max accuracy

Iterative imputed data :-

Logistic Regression (l1) =0.69

Logistic Regression (l2) =0.69

Random imputed data :-

Logistic Regression (l1) =0.66

Logistic Regression (l2) =0.65

KNN imputed data :-

Logistic Regression (l1) =0.68

Logistic Regression (l2) =0.63

SVM with linear and RBF kernel results in similar performance

Looking at multiple modules considering accuracy values and importantly using ROC-AUC curves for test and cross-validation data chose logistic regression with l1 regularization for overall performance

Trained many models on each of the 18 Dataset's

Results in jupyter notebooks.

Kaggle Upload:

The prediction that needs to be uploaded on kaggle is on a different dataset which is 'cardio-test.csv'. In order to predict on this data, there are some changes that need to be done:

1. Dropping 'id'
2. Converting age from days to years
3. Label encoding all string categorical values
4. Standardizing the data

   Kaggle RMSE =0.7040

Task 2:

Check the predictions of our model on cardio-complete dataset which doesn't have any missing values:

Data PreProcessing:

1. Dropping column 'id'
2. Label encoder for gender, cholesterol and gluc
3. Changing 'age' from days to years
4. Split the data into 70:30 training testing

Logistic Regresssion :

Logistic Regression is applied using Hyperparameter tuning and the best parameters were as follows:

C:0.01

Penalty:l1

Solver: liblinear

The data is fitted on these parameters

Comparing Task 2 and Task 1 prediction scores:

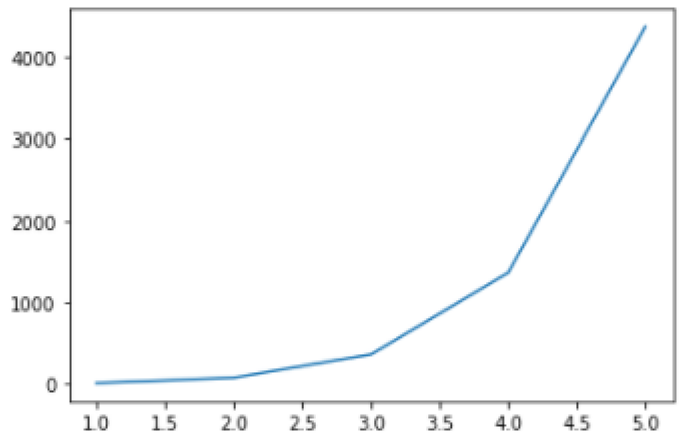
	Accuracy	F1-Score	Precision	Recall
Task1	0.642	0.58	0.66	0.52
Task2	0.67	0.64	0.75	0.56

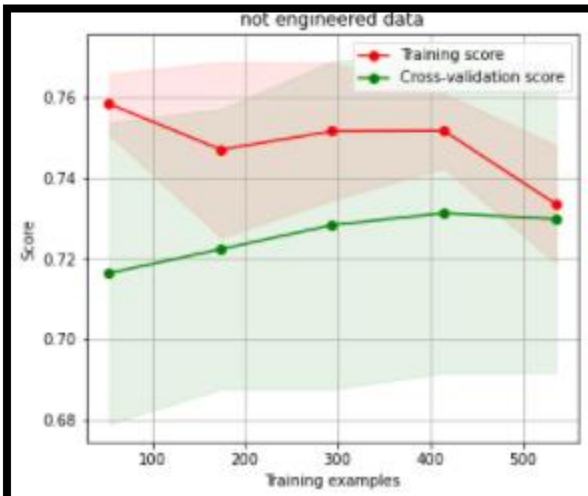
The table above shows the precision, recall, f1, accuracy scores on the testing datasets for task 1 and task 2. We can see overall improvement in all the metrics is improved we can see this especially for logreg models with L1 regularization precision improves the most.This Might be because there is no missing values hence the variability errors and randomness introduced by imputation is not seen here like in case of TASK 1

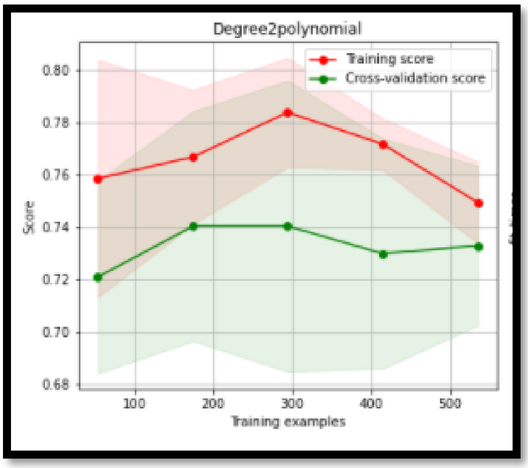
Task 3:

Here I’ve transformed the training features from task 1 into 2,3,4 degree polynomials and computed Accuracy the best accuracy was observed for degrees =2 and it reduced as it increased ,this may be because the higher the degree more nonlinear the decision surface is which might not be optimum for classifying ground truth points which are linear or almost linear

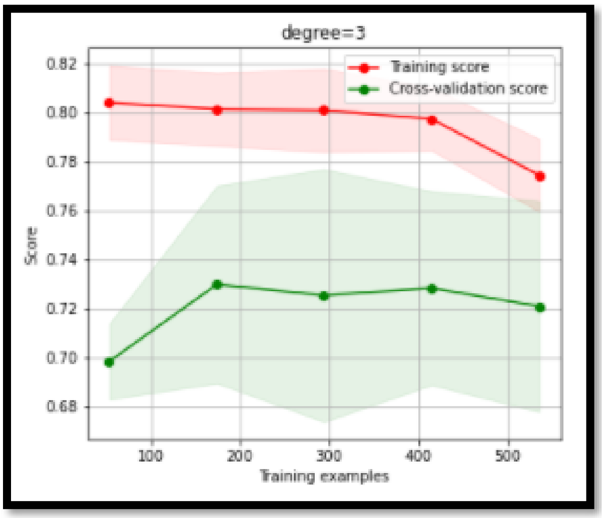
```
Degree: 1, Features: 12
Degree: 2, Features: 78
Degree: 3, Features: 364
Degree: 4, Features: 1365
Degree: 5, Features: 4368
```







Degree 2 polynomial



. Degree 3 polynomial

As we can see from the graph that Training score is higher than cross-validation score which means that there is high variance. We can also notice that as we increase the degree of polynomial features the model overfits the data even more.

Conclusion:

Task 1 included data visualization and prediction on kaggle(cardio-test) dataset using the best model which was Logistic Regression. We used cardio-train and cardio-validation to predict on the test dataset. Missing values were filled using knn std scaler was used .

Task 2 was testing the model used in Task 1 on the cardio-complete dataset. The given dataset didn’t have any missing values. Results of task 1 and task 2 were compared.

Task 3 gave the understanding of overfitting the data when complex features are added. Training and cross validation scores were plotted for polynomial features with degree 2 and 3.