**Case Study Assignment – Data Mining**

**DLH GROUP085**

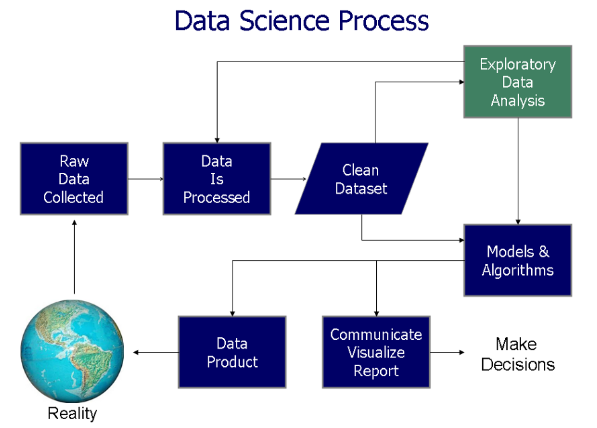
1. **Problem Statement-**

A classification model to be created on Health record’s data containing information about vitals that will help to identify if person would be having Heart disease in next 10 years.

1. **Exploratory Data Analysis (EDA):**

EDA is an approach to [analyzing](https://en.wikipedia.org/wiki/Data_analysis) [data sets](https://en.wikipedia.org/wiki/Data_set) to summarize main characteristics with visual methods. EDA helped to depict beyond formal modeling. EDA focuses on checking assumptions required for model fitting and hypothesis testing and handling missing values and making transformations of variables as needed. Further below are the insights we got from data-

1. Maximize insight into a data set
2. Uncover underlying structure
3. Extract important variables
4. Detect outliers and anomalies
5. Test underlying assumptions
6. Develop parsimonious models
7. Determine optimal factor settings



What we did in EDA:

1. Importing the required libraries for EDA

Below are the libraries that are used to perform EDA

import pandas as pd

import numpy as np

from matplotlib import pyplot as plt

import seaborn as sns

1. Loading the data into the data frame

originalDataSet = pd.read\_csv('Problem2\_Data.csv')

1. Checking the types of data

originalDataSet.info()

originalDataSet.head(1)

originalDataSet.describe()

Here we check for the datatypes because sometimes data stored as a string or object, if in that case, we need to convert string to integer data, only then we can plot the data via a graph. Here, in this case, the data is already in integer format so nothing to worry.

1. **Preprocess the data**
2. Dropping irrelevant columns

Step 1: Selected those columns which have good correlation with Target column

Step 2: Dropping those columns what are highly correlated

print("Heat map of patient original data")

correlation\_matrix = originalDataSet.corr()

fig = plt.figure(figsize=(12, 9))

sns.heatmap(correlation\_matrix, vmax=0.8, square=True)

Step 3: Dropping the missing or null values and dropping ID column

# find column which hold any null value and replace them with mean of the column

nullcolumnsList = originalDataSet.columns[originalDataSet.isnull().any()]

nullrowsbasedonthecolumnName = originalDataSet[nullcolumnsList].isnull().sum()

# fill null rows with mean of the respective column

nba = originalDataSet.fillna(originalDataSet.mean())

# replace NA with mean of the respective column

nba.replace('NA', originalDataSet.mean())

# make a copy of dataframe and replace -99 with 0 and calculate mean so this mean will use to

#replace with -99 in original dataframe

wba = nba.copy()

wba[wba == -99] = 0

# Now replace dataframe with mean of the respective column from dataset which has zero in place of -99

nba.replace(-99, wba.mean(),inplace=True)

1. Perform Data Cleaning
2. Detecting Outliers
3. Pre-process the data (Data Normalization)

scaler = Normalizer()

nba.loc[:, nba.columns != 'Target'] = scaler.fit\_transform(nba.loc[:, nba.columns != 'Target'])

1. Applying PCA- for reducing correlation between independent variables

A close up of a map

Description automatically generated

1. Remove Outliers with help of variance\_inflation\_factor
2. **Select Training data, test data**

Data Splitting

70% for test – 30% for training’

X = FinalDataFrame.drop(['Target'], axis=1)

Y = FinalDataFrame ['Target']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, Y, test\_size=0.3 )

1. **Test and Train the Model (Cross Validation)**
2. #Create Logistics Model, train it and test the model

# Create Model

logModel = LogisticRegression()

# Train Model

logModel.fit(X\_train,y\_train)

#Test Model

predictions = logModel.predict(X\_test)

Logistics regression and fit the train data and then test and predict and it gives accuracy of 80 to 85 %

1. #Create KNN (KNeighborsClassifier) Model , train it and test the model

# Create Model

knn = KNeighborsClassifier(n\_neighbors=1)

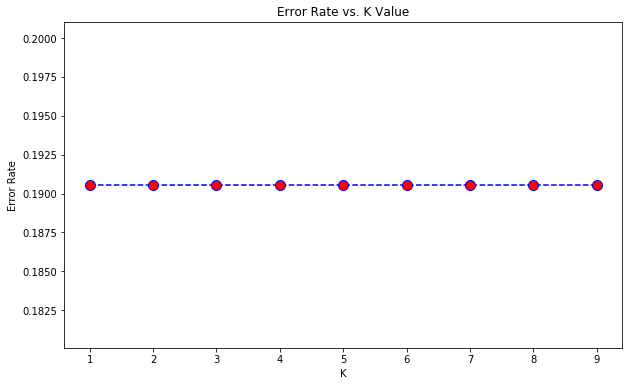
# Train Model

knn.fit(X\_train,y\_train)

# Test Model

predictions = knn.predict(X\_test)

Create and train KNN model with k ranges from 1 to 10 and select k =1 as the error rate is same for all k value then test and predict and found that the accuracy of this model is also in range of 80 to 85 %



1. .1 Evaluate the model performance (Logistics Regression)  
   Confusion Matrix LogisticRegression:

[[6124 778]

[ 962 2421]]

Classification report LogisticRegression:

precision recall f1-score support

0 0.86 0.89 0.88 6902

1 0.76 0.72 0.74 3383

accuracy 0.83 10285

macro avg 0.81 0.80 0.81 10285

weighted avg 0.83 0.83 0.83 10285

A.2Evaluate the model performance (KNN Regression)

Confusion Matrix KNN Model:

[[5874 1028]

[ 927 2456]]

Note: From above image:'Error Rate vs. K Value' The Error rate is same accross all k value so we will pick k = 0 as final value

precision recall f1-score support

0 0.86 0.85 0.86 6902

1 0.70 0.73 0.72 3383

accuracy 0.81 10285

macro avg 0.78 0.79 0.79 10285

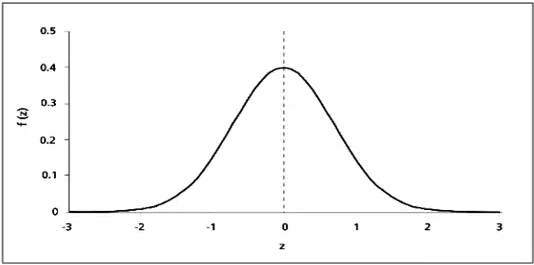
weighted avg 0.81 0.81 0.81 10285

1. Suggest ways of improving the model

We can try calculating z score for Outlier Detection-

The standard score (more commonly referred to as a z-score) is a very useful statistic because it

1. allows us to calculate the probability of a score occurring within our normal distribution
2. enables us to compare two scores that are from different normal distributions



The modified z score is a standardized score that measures outlier strength or how much a score differs from the typical score. Using standard deviation units, it approximates the difference of the score from the median.

The modified z score might be more robust than the standard z score because it relies on the median for calculating the z score. It is less influenced by outliers when compared to the standard z score.

The standard z score is calculated by dividing the difference from the mean by the standard deviation. The modified z score is calculated from the mean absolute deviation (MeanAD) or median absolute deviation (MAD). These values must be multiplied by a constant to approximate the standard deviation.

Depending on the value of MAD, the modified z score is calculated in one of 2 ways:

* **If MAD does equal 0.** Subtract the median from the score and divide by 1.253314\*MeanAD. 1.253314\*MeanAD approximately equals the standard deviation: (X-MED)/(1.253314\*MeanAD).
* **If MAD does not equal 0.** Subtract the median from the score and divide by 1.486\*MAD: (X-MED)/(1.486\*MAD). 1.486\*MAD approximately equals the standard deviation.

1. Any interesting observations
2. Challenges faced and how you mitigated the challenges

Health data comes with many challenges, including security, visualization, and a number of data integrity concerns.

|  |  |
| --- | --- |
| **Challenges** | **Mitigation** |
| **Dirty data** – Blank, NA, Negative values can quickly derail our project | Data Cleaning-   1. find column which hold any null value and replace them with mean of the column 2. replace NA with mean of the respective column 3. replace -99 with mean of the column 4. delete id column as this was index column of the dataset |
| Highly Correlated data | drop highly corelated columns found in variable uniquecolumn\_name\_to\_be\_removed |
| Unlabeled Data | Labeled data can be normalized with better understanding of data instead we used standard normalizer |

1. Assumptions (if any)  
     
   Missing Data may be represented by either NAs, Blanks or values such as -99/-999 etc
2. Conclusion  
     
   LogisticRegression and KNN both models are predicting whether patient has coronary heart disease or

not with 81% to 85% of accuracy