HEART STROKE PREDICTION

An Internship report submitted in partial fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

By

DUSI SAI SURYA ABHISHEK

Regd. No.: 20B91A5454

Under the Supervision of

Mr. Gundala Nagaraju

Henotic Technology Pvt Ltd, Hyderabad (Duration: 7th July, 2022 to 6th September, 2022)

DEPARTMENT OF INFORMATION TECHNOLOGY

SAGI RAMA KRISHNAM RAJU ENGINEERING COLLEGE

(An Autonomous Institution)

Approved by AICTE, NEW DELHI and Affiliated to JNTUK, Kakinada

CHINNA AMIRAM, BHIMAVARAM, ANDHRA PRADESH

SAGI RAMA KRISHNAM RAJU ENGINEERING COLLEGE

(Autonomous)

Chinna Amiram, Bhimavaram

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the "**Summer Internship Report"** submitted by **DUSI SAI SURYA ABHISHEK, 20B91A5454** is work done by him/her and submitted during 2021 - 2022 academic year, in partial fulfillment of the requirements for the award of the Summer Internship Program for **Bachelor of Technology in ARTIFICIAL INTELLIGENCE AND DATA SCIENCE,** at **HENOTIC TECHNOLOGIES** from *05.07.2022 to 04.09.2022 for AIML & Core (Mech & Civil), for Cyber Security 11.07.2022 to 10.09.2022*

Department Internship Coordinator **Dean -T & P Cell** **Head of the Department**

Table of Contents

Abstract

Heart-related diseases or Cardiovascular Diseases (CVDs) are the main reason for a huge number of death in the world over the last few decades and has emerged as the most life-threatening disease, not only in India but in the whole world. So, there is a need fora reliable, accurate, and feasible system to diagnose such diseases in time for proper treatment. Machine Learning algorithms and techniques have been applied to various medical datasets to automate the analysis of large and complex data. Many researchers, in recent times, have been using several machine learning techniques to help the health care industry and the professionals in the diagnosis of heart-related diseases. Heart is the next major organ comparing to the brain which has more priority in the Human body. It pumps the blood and supplies it to all organs of the whole body. Prediction of occurrences of heart diseases in the medical field is significant work. Data analytics is useful for prediction from more information and it helps the medical center to predict various diseases. A huge amount of patient-related data is maintained on monthly basis. The stored data can be useful for the source of predicting the occurrence of future diseases. Some of the data mining and machine learning techniques are used to predict heart diseases, such as Artificial Neural Network (ANN), Random Forest,and Support Vector Machine (SVM).Prediction and diagnosingof heart disease become a challenging factor faced by doctors and hospitals both in India and abroad. To reduce the large scale of deaths from heart diseases, a quick and efficient detection technique is to be discovered. Data mining techniques and machine learning algorithms play a very important role in this area. The researchers accelerating their research works to develop software with thehelp of machine learning algorithms which can help doctors to decide both prediction and diagnosing of heart disease. The main objective of this research project is to predict the heart disease of a patient using machine learning algorithms.

1.0 Introduction

With the increasing power of computer technology, companies and institutions cannowadays store large amounts of data at reduced cost. The amount of available datais increasing exponentially and cheap disk storage makes it easy to store data thatpreviously was thrown away. There is a huge amount of information locked up indatabases that is potentially important but has not yet been explored. The growing sizeand complexity of the databases makes it hard to analyse the data manually, so it isimportant to have automated systems to support the process. Hence there is the needof computational tools able to treat these large amounts of data and extract valuableinformation.

In this context, Data Mining provides automated systems capable of processinglarge amounts of data that are already present in databases. Data Mining is used toautomatically extract important patterns and trends from databases seeking regularitiesor patterns that can reveal the structure of the data and answer business problems. DataMining includes learning techniques that fall into the field of Machine learning. Thegrowth of databases in recent years brings data mining at the forefront of new businesstechnologies.

A key challenge for the insurance industry is to charge each customer an appropriate price for the risk they represent. Risk varies widely from customer to customer and a deep understanding of different risk factors helps predict the likelihood and cost of insurance claims. The goal of this program is to see how well various statistical methods perform in predicting auto Insurance claims based on the characteristics of the driver, vehicle and driver / vehicle coverage details.

A number of factors will determine BI claims prediction among them a driver's age, past accident history, and domicile, etc. However, this contest focused on the relationship between claims and vehicle characteristics well as other characteristics associated with the auto insurance policies.

1.1. What are the different types of Machine Learning?

There are four types of machine learning algorithms: supervised, semi-supervised, unsupervised and reinforcement.

1.1.1 **Supervised learning**

In supervised learning, the machine is taught by example. The operator provides the machine learning algorithm with a known dataset that includes desired inputs and outputs, and the algorithm must find a method to determine how to arrive at those inputs and outputs. While the operator knows the correct answers to the problem, the algorithm identifies patterns in data, learns from observations and makes predictions. The algorithm makes predictions and is corrected by the operator – and this process continues until the algorithm achieves a high level of accuracy/performance.

Under the umbrella of supervised learning fall: Classification, Regression and Forecasting.

1. **Classification**: In classification tasks, the machine learning program must draw a conclusion from observed values and determine to what category new observations belong. For example, when filtering emails as 'spam' or 'not spam', the program must look at existing observational data and filter the emails accordingly.
2. **Regression**: In regression tasks, the machine learning program must estimate – and understand – the relationships among variables. Regression analysis focuses on one dependent variable and a series of other changing variables – making it particularly useful for prediction and forecasting.
3. **Forecasting**: Forecasting is the process of making predictions about the future based on the past and present data and is commonly used to analyse trends.

1.1.2 Semi-supervised learning

Semi-supervised learning is like supervised learning, but instead uses both labelled and unlabelled data. Labelled data is essentially information that has meaningful tags so that the algorithm can understand the data, whilst unlabelled data lacks that information. By using this combination, machine learning algorithms can learn to label unlabelled data.

1.1.3 Unsupervised learning

Here, the machine learning algorithm studies data to identify patterns. There is no answer key or human operator to provide instruction. Instead, the machine determines the correlations and relationships by analysing available data. In an unsupervised learning process, the machine learning algorithm is left to interpret large data sets and address that data accordingly. The algorithm tries to organise that data in some way to describe its structure. This might mean grouping the data into clusters or arranging it in a way that looks more organised.

As it assesses more data, its ability to make decisions on that data gradually improves and becomes more refined.

Under the umbrella of unsupervised learning, fall:

1. **Clustering**: Clustering involves grouping sets of similar data (based on defined criteria). It's useful for segmenting data into several groups and performing analysis on each data set to find patterns.
2. **Dimension reduction**: Dimension reduction reduces the number of variables being considered to find the exact information required.

1.1.4 Reinforcement learning

Reinforcement learning focuses on regimented learning processes, where a machine learning algorithm is provided with a set of actions, parameters and end values. By defining the rules, the machine learning algorithm then tries to explore different options and possibilities, monitoring and evaluating each result to determine which one is optimal. Reinforcement learning teaches the machine trial and error. It learns from past experiences and begins to adapt its approach in response to the situation to achieve the best possible result.

1.2. Benefits of Using Machine Learning in Heart Stroke Prediction

1. Increased accuracy for effective heart disease diagnosis.
2. Handles roughest(enormous) amount of data using random forest algorithm and feature selection.
3. Reduce the time complexity of doctors.
4. Cost effective for patients.

1.3. About Industry (Heart diseases)

The **healthcare industry** (also called the **medical industry** or **health economy**) is an aggregation and integration of sectors within the economic system that provides goods and services to treat patients with curative, preventive, rehabilitative, and palliative care. It includes the generation and commercialization of goods and services lending themselves to maintaining and re-establishing health. The modern healthcare industry includes three essential branches which are services, products, and finance and may be divided into many sectors and categories and depends on the interdisciplinary teams of trained professionals and paraprofessionals to meet health needs of individuals and populations.

The healthcare industry is one of the world's largest and fastest-growing industries. Consuming over 10 percent of gross domestic product (GDP) of most developed nations, health care can form an enormous part of a country's economy. U.S. health care spending grew 4.6 percent in 2019, reaching $3.8 trillion or $11,582 per person. As a share of the nation's Gross Domestic Product, health spending accounted for 17.7 percent. The per capita expenditure on health and pharmaceuticals in OECD countries has steadily grown from a couple of hundred in the 1970s to an average of US$4'000 per year in current purchasing power parities.

1.3.1 AI / ML Role in Heart Stroke Prediction

Artificial intelligence (AI) has been used for the first time to instantly and accurately measure blood flow and thus predict chances of death, heart attack and stroke.

The AI allowed the researchers to precisely and instantaneously quantify the blood flow to the heart muscle and deliver the measurements to the medical teams treating the patients..

The AI was able to predict which patients would die or suffer adverse events better than a doctor would be able to do alone.

2.0 Heart Stroke Model Claims (Heart Stroke Prediction)

The model predicts the chances a person will have stroke based on symptoms like age,gender,smoking status,bmi,work type and residence type.

According to the model It claims to be accurate than the predictions a doctor can do alone and claims to be an asset for the medical industry and helps to save lives of people.

It also claims to be faster in prediction than doctors with lesser information and accurately.Model predicts the strokes with less number of attributes and to be accurate.

2.1. Main Drivers for AI Auto QuoteAnalysis

Predictive modelling allows for simultaneous consideration of many variablesand quantification of their overall effect. When a large number of claims areanalysed, patterns regarding the characteristics of the claims that drive lossdevelopment begin to emerge.

The following are the main drivers which influencing the Claims Analytics:

• **Policy Characteristics** ✓ Exposures ✓ Limits and Deductibles ✓ Coverages and Perils • **Insured Characteristics** ✓ Credit Information ✓ Prior loss experience ✓ Payment history • **Geography based on insured locations** ✓ Auto Repair Costs ✓ Jurisdictional Orientation ✓ Demographics ✓ Crime • **Agency Characteristics** ✓ Exclusive Agents	• **Claim information** ✓ FNOL ✓ Claimant data (Credit info, geography, social data, etc.) ✓ Other participants (insured, doctors, lawyers, witnesses, etc.) ✓ Cause, type of Injury/Damage ✓ Injury or damaged object ✓ Coverage ✓ Loss Location ✓ Date and time of Loss and Report ✓ Weather at time & location of loss • **Details from Prior Claims** ✓ from same insured ✓ from same claimant ✓ from same location

✓ Independent Agents	• **Household Characteristics**

2.2. Internship Project - Data Link

The internship project data has taken from Kaggle and the link is www.kagggle.com

https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease

3.0 AI / ML Modelling and Results

3.1. Your Problem of Statement

Heart stroke prediction model

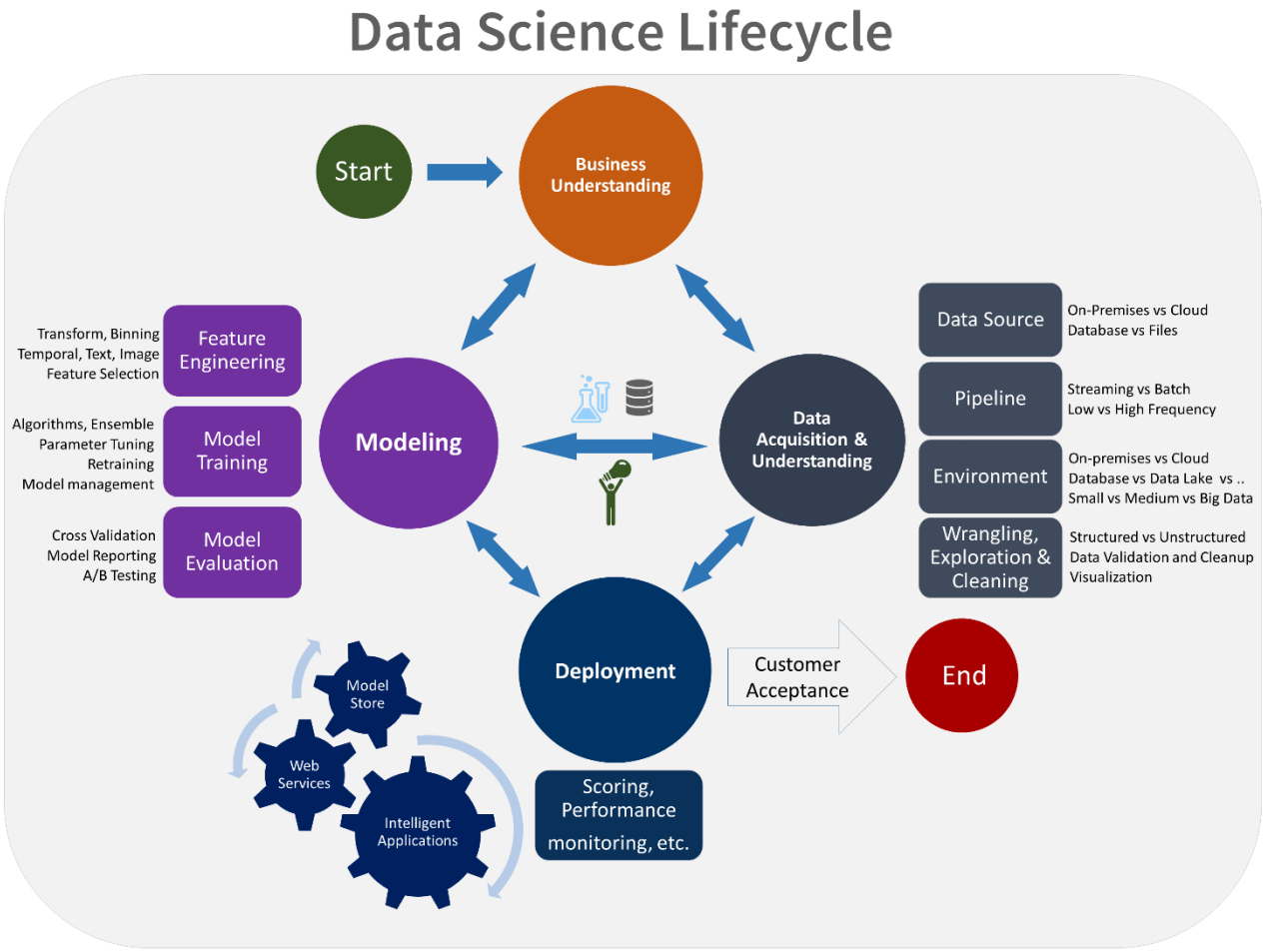
Predictive models are most effective when they are constructed using acompany's own historical claims data since this allows the model to recognize the specific nature of a company's exposure as well as its claims practices.The construction of the model also involves input from the companythroughout the process, as well as consideration of industry leading claimspractices and benchmarks.

Predictive modelling can be used to quantify the impact to the claimsdepartment resulting from the failure to meet or exceed claim service leadingpractices. It can also be used to identify the root cause of claim leakage.Proper use of predictive modelling will allow for potential savings across twodimensions:

- Early identification of claims with the potential for high leakage, thereby allowing for the proactive management of the claim
- Recognition of practices that are unnecessarily increasing claims settlement payments

3.2. Data Science Project Life Cycle

Data Science is a multidisciplinary field of study that combines programming skills, domain expertise and knowledge of statistics and mathematics to extract useful insights and knowledge from data.



3.2.1 Data Exploratory Analysis

Exploratory data analysis has been done on the data to look for relationship and correlation between different variables and to understand how they impact or target variable.

The exploratory analysis is done for Auto Quote / Policy Conversionwith different parameters and all the charts are presented in **Appendices 6.2 - List of charts (6.2.1 to 6.2.9)**

3.2.2 Data Pre-processing

We removed variables which does not affect our target variable(Claimed_Target)as they mayadd noise and also increase our computation time,we checked the data for anomalous data points and outliers.We did principal component analysis on the data set to filter out unnecessary variables and to select only the important variables which have greater correlation with our target variable.

3.2.2.1. Check the Duplicate and low variation data

These can be of two types: Duplicate Values: When two features have the same set of values. Duplicate Index: When the value of two features is different, but they occur at the same index.

There are two ways you can remove duplicates. One is deleting the entire rows and other is removing the column with the most duplicates. Method 1: Removing the entire duplicates rows values. For removing the entire rows that have the same values using the method drop_duplicates ().

```
In [5]: 1 HSData.duplicated().sum()

Out[5]: 0
```

Low variance means there is a small variation in the prediction of the target function with changes in the training data set. At the same time, High variance shows a large variation in the prediction of the target function with changes in the training dataset.

3.2.2.2. Identify and address the missing variables

The real-world data often has a lot of missing values. The cause of missing values can be data corruption or failure to record data. The handling of missing data is very important during the pre-processing of the dataset as many machine learning algorithms do not support missing values.

7 ways to handle missing values in the dataset:

1. Deleting Rows with missing values
2. Impute missing values for continuous variable
3. Impute missing values for categorical variable
4. Other Imputation Methods
5. Using Algorithms that support missing values
6. Prediction of missing values
7. Imputation using Deep Learning Library — Datawig

```
In [6]: 1 HSData.isna().sum()

Out[6]: HeartDisease        0
        BMI                 0
        Smoking             0
        AlcoholDrinking     0
        Stroke              0
        PhysicalHealth      0
        MentalHealth        0
        DiffWalking         0
        Sex                 0
        AgeCategory         0
        Race                0
        Diabetic            0
        PhysicalActivity    0
        GenHealth           0
        SleepTime           0
        Asthma              0
        KidneyDisease       0
        SkinCancer          0
        dtype: int64
```

3.2.2.3. Handling of Outliers

Outliers

In statistics, we call the data points significantly different from the rest of the dataset outliers. In other words, an outlier contains a value that is inconsistent or doesn't comply with the general behavior.

Multiple reasons cause outliers to appear in a dataset. In this sense, a measurement error or an input error can lead to the existence of outlier values.

Outlier Detection

Detection of outliers isn't a trivial problem. We're trying to identify observations that don't fit into the general characteristics of the dataset.

We can choose from a variety of approaches, depending on the dataset at hand. Let's explore some of the well-known outlier detection techniques:

1. Box plot

2. IQR method

3. Z-score method

4. 'Distance from the mean' method (Multivariate method)

Handling Outliers

Detection and handling of outliers is a fundamental problem in data science and machine learning. Solving this problem isn't straightforward, just like the missing values problem.

There are some techniques used to deal with outliers.

Deleting observations.

Transforming values.

Imputation.

Separately treating.

Deleting observations. Sometimes it's best to completely remove those records from your dataset to stop them from skewing your analysis.

3.2.2.4. Categorical data and Encoding Techniques

Categorical Data is the data that generally takes a limited number of possible values. Also, the data in the category need not be numerical, it can be textual in nature. All machine learning models are some kinds of mathematical model that need numbers to work with.

Categorical variables can be divided into two categories:

Nominal: no order

Ordinal: there is some order between values.

Why do we need encoding?

Most machine learning algorithms cannot handle categorical variables unless we convert them to numerical values

Many algorithm's performances even vary based upon how the categorical variables are encoded.

The two most popular techniques are an Ordinal Encoding and a One-Hot Encoding.

In [7]:
```
from sklearn.preprocessing import LabelEncoder

LE=LabelEncoder()

h['GenHealth']=LE.fit_transform(h[['GenHealth']])
h['Diabetic']=LE.fit_transform(h[['Diabetic']])
h['Race']=LE.fit_transform(h[['Race']])

```

In [11]:
```
h['HeartDisease'].value_counts()

```

Out[11]:
```
0    292422
1     27373
Name: HeartDisease, dtype: int64
```

3.2.2.5. Feature Scaling

Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.

There are several ways to do feature scaling. The top 5 of the most commonly used feature scaling techniques.

Absolute Maximum Scaling

Min-Max Scaling

Normalization

Standardization

Robust Scaling

In [20]:
```
# Scaling the features by using MinMaxScaler

from sklearn.preprocessing import MinMaxScaler

mmscaler = MinMaxScaler(feature_range=(0, 1))

x_train[col1] = mmscaler.fit_transform(x_train[col1])
x_train = pd.DataFrame(x_train)

x_test[col1] = mmscaler.fit_transform(x_test[col1])
x_test = pd.DataFrame(x_test)
```

3.2.3 Selection of Dependent and Independent variables

The dependent or target variable here isClaimed Target which tells us a particular policy holder has filed a claim or not the target variable is selected based on our business problem and what we are trying to predict.

The independent variables are selected after doing exploratory data analysis and we used Boruta to select which variables are most affecting our target variable.

In [14]:
```
Indvar=[]
for i in h.columns:
    if i!='HeartDisease':
        Indvar.append(i)

Targetvar='HeartDisease'

x=h[Indvar]
y=h[Targetvar]
```

3.2.4 Data Sampling Methods

The data we have is highly unbalanced data so we used some sampling methods which are used to balance the target variable so we our model will be developed with good accuracy and precision. We used three Sampling methods

3.2.4.1. Stratified sampling

Stratified sampling randomly selects data points from majority class so they will be equal to the data points in the minority class. So, after the sampling both the class will have same no of observations.

It can be performed using strata function from the library sampling.

3.2.4.2. Simple random sampling

Simple random sampling is a sampling technique where a set percentage of the data is selected randomly. It is generally done to reduce bias in the dataset which can occur if data is selected manually without randomizing the dataset.

We used this method to split the dataset into train dataset which contains 70% of the total data and test dataset with the remaining 30% of the data.

In [15]:
```
# Random oversampling can be implemented using the RandomOverSampler class

from imblearn.over_sampling import RandomOverSampler

oversample = RandomOverSampler(sampling_strategy=0.15)

x_over, y_over = oversample.fit_resample(x, y)

print(x_over.shape)
print(y_over.shape)
```

```
(336285, 17)
(336285,)
```

3.2.5 Models Used for Development

We built our predictive models by using the following ten algorithms

3.2.5.1. Model 01

Logistic uses logit link function to convert the likelihood values to probabilities so we can get a good estimate on the probability of a particular observation to be positive class or negative class.The also gives us p-value of the variables which tells us about significance of each independent variable.

3.2.5.2. Model 02

Random forest is an algorithm that consists of many decision trees. It was first developedby Leo Breiman and Adele Cutler. The idea behind it is to build several trees,to have the instance classified by each tree, and to give a "vote" at each class.The model uses a "bagging" approach and the random selection of features to build acollection of decision trees with controlled variance. The instance's

class is to the classwith the highest number of votes, the class that occurs the most within the leaf in whichthe instance is placed.

The error of the forest depends on:

- Trees correlation: the higher the correlation, the higher the forest error rate.
- The strength of each tree in the forest. A strong tree is a tree with low error. By using trees that classify the instances with low error the error rate of the forest decreases.

3.2.5.3. Model 03

Artificial neural networks can theoretically solve any problem. ANNs can identify hidden patterns between the variables and can find how different combinations of variables can affect the target variable. The error correction is done by gradient descent algorithm which can reduce the error rate as much as possible for the given data

3.2.5.1. Model 04

Decision trees are **an approach used in supervised machine learning, a technique which uses labelled input and output datasets to train models**. The approach is used mainly to solve classification problems, which is the use of a model to categorise or classify an object.

3.2.5.2. Model 05

KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

3.2.5.6. Model 06

Naive Bayes is a generative model. (Gaussian) Naive Bayes **assumes that each class follow a Gaussian distribution**. The difference between QDA and (Gaussian) Naive Bayes is that Naive Bayes assumes independence of the features, which means the covariance matrices are diagonal matrices.

3.2.5.7. Model 07

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed **gradient-boosted** decision tree (GBDT) machine learning library. It provides parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems. It's vital to an understanding of XGBoost to first grasp the machine learning concepts and algorithms that XGBoost builds upon supervised machine learning, decision trees, ensemble learning, and boosting. Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features.

3.2.5.8. Model 08

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks. Since it is based on decision tree algorithms, it splits the tree leaf wise with the best fit whereas other boosting algorithms split the tree depth wise or level wise rather than leaf-wise. So, when growing on the same leaf in Light GBM, the leaf-wise algorithm can reduce more loss than the level-wise algorithm and hence results in much better accuracy which can rarely be achieved by any of the existing boosting algorithms. Also, it is surprisingly very fast, hence the word 'Light'.

3.2.5.9. Model 09

Bagging, also known as Bootstrap aggregating, is an ensemble learning technique that helps to improve the performance and accuracy of machine learning algorithms. It is used to deal with bias-variance trade-offs and reduces the variance of a prediction model. Bagging avoids overfitting of data and is used for both regression and classification models, specifically for decision tree algorithms.

3.3. AI / ML ModelsAnalysis and Final Results

We used our train dataset to build the above models and used our test data to check the accuracy and performance of our models.

We used confusion matrix to check accuracy, Precision, Recall and F1 score of our models and compare and select the best model for given auto dataset of size ~ 272252 policies.

3.3.1 Different Model codes

- The Python code for models with stratified sampling technique as follows:

In [96]:
```
# Load the Loan data
neodata = pd.read_csv(r"C:\Users\Narasimha Reddy\OneDrive\Desktop\intern\neo.csv", header=0)
# Copy to back-up file
neodata_bk = neodata.copy()
# Display first 5 values
neodata.head()
```

- The Python code for models withsimple random sampling technique as follows:

In [69]:

```
# Build the Calssification models with Over Sampling and compare the results

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from xgboost import XGBClassifier
import lightgbm as lgb
from sklearn.ensemble import BaggingClassifier

# Create objects of classification algorithms with default hyper-parameters

ModelLR = LogisticRegression()
ModelDC = DecisionTreeClassifier()
ModelRF = RandomForestClassifier()
ModelET = ExtraTreesClassifier()
ModelKNN = KNeighborsClassifier()
ModelGNB=GaussianNB()
ModelXGB=XGBClassifier()
ModelLGB = lgb.LGBMClassifier()
ModelBAG=modelBAG = BaggingClassifier(base_estimator=None, n_estimators=100, max_samples=1.0, max_features=1.0,
                            bootstrap=True, bootstrap_features=False, oob_score=False, warm_start=False,
                            n_jobs=None, random_state=None, verbose=0)


# Evalution matrix for all the algorithm
```

```
# Evalution matrix for all the algorithm

MM = [ModelLR, ModelDC, ModelRF, ModelET,ModelKNN,ModelGNB,ModelXGB,ModelLGB,ModelBAG]
for models in MM:

    # Train the model training dataset

    models.fit(x_train, y_train)

    # Prediction the model with test dataset

    y_pred = models.predict(x_test)
    y_pred_prob = models.predict_proba(x_test)

    # Print the model name

    print('Model Name: ', models)

    # confusion matrix in sklearn

    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import classification_report

    # actual values

    actual = y_test

    # predicted values

    predicted = y_pred

    # confusion matrix

    matrix = confusion_matrix(actual,predicted, labels=[1,0],sample_weight=None, normalize=None)
    print('Confusion matrix : \n', matrix)

    # outcome values order in sklearn
```

```
# outcome values order in sklearn

tp, fn, fp, tn = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1)
print('Outcome values : \n', tp, fn, fp, tn)

# classification report for precision, recall f1-score and accuracy

C_Report = classification_report(actual,predicted,labels=[1,0])

print('Classification report : \n', C_Report)

# calculating the metrics

sensitivity = round(tp/(tp+fn), 3);
specificity = round(tn/(tn+fp), 3);
accuracy = round((tp+tn)/(tp+fp+tn+fn), 3);
balanced_accuracy = round((sensitivity+specificity)/2, 3);

precision = round(tp/(tp+fp), 3);
f1Score = round((2*tp/(2*tp + fp + fn)), 3);

# Matthews Correlation Coefficient (MCC). Range of values of MCC lie between -1 to +1.
# A model with a score of +1 is a perfect model and -1 is a poor model

#from math import sqrt

#mx = (tp+fp) * (tp+fn) * (tn+fp) * (tn+fn)
#MCC = round(((tp * tn) - (fp * fn)) / sqrt(mx), 3)

print('Accuracy :', round(accuracy*100, 2),'%')
print('Precision :', round(precision*100, 2),'%')
print('Recall :', round(sensitivity*100,2), '%')
print('F1 Score :', f1Score)
print('Specificity or True Negative Rate :', round(specificity*100,2), '%'  )
print('Balanced Accuracy :', round(balanced_accuracy*100, 2),'%')
#print('MCC :', MCC)
```

```
# Area under ROC curve

from sklearn.metrics import roc_curve, roc_auc_score

print('roc_auc_score:', round(roc_auc_score(actual, y_pred), 3))

# ROC Curve

from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
Model_roc_auc = roc_auc_score(actual, y_pred)
fpr, tpr, thresholds = roc_curve(actual, models.predict_proba(x_test)[:,1])
plt.figure()
#
plt.plot(fpr, tpr, label= 'Classification Model' % Model_roc_auc)
plt.plot([0, 1], [0, 1],'r--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.savefig('Log_ROC')
plt.show()
print('-----------------------------------------------------------------------------------------------------------')
#-----------------------------------------------------------------------------------------------------------
new_row = {'Model Name' : models,
           'True Positive': tp,
           'False Negative': fn,
           'False Positive': fp,
           'True Negative': tn,
           'Accuracy' : accuracy,
           'Precision' : precision,
           'Recall' : sensitivity,
           'F1 Score' : f1Score,
           'Specificity' : specificity,
           'MCC': 'MCC',
           'ROC_AUC_Score':roc_auc_score(actual, y_pred),
           'Balanced Accuracy':balanced_accuracy}
CSResults = CSResults.append(new_row, ignore_index=True)
#-----------------------------------------------------------------------------------------------------------
```

4.0 Conclusions and Future work

The model results in the following order by considering the model accuracy, F1 score and RoC AUC score.

1) **Random Forest** with Stratified and Random Sampling
2) **Bagging classifier** with Simple Random Sampling
3) **Extra Tree Classifier** with Simple Random Sampling

We recommend model - **Random Forest** with Stratified and Random Sampling technique as a best fit for the give n BI claims dataset. We considered Random Forest because it uses bootstrap aggregation which can reduce bias and variance in the data and can leadsto good predictions with claims dataset.

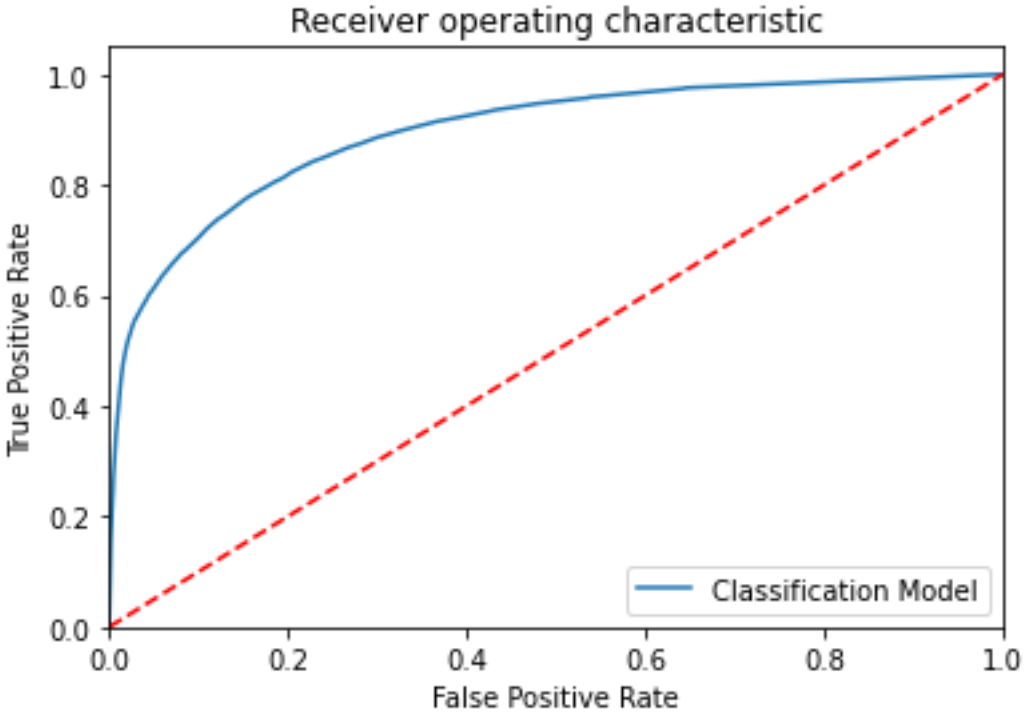
Note: Add results screen snapshot here

The future work to evaluate the "Other Types Claims" in auto Insurance by using classification methods.

Model Name: RandomForestClassifier()
Confusion matrix :
[[7423 5724]
[2809 84930]]
Outcome values :
7423 5724 2809 84930
Classification report :

	precision	recall	f1-score	support
1	0.73	0.56	0.64	13147
0	0.94	0.97	0.95	87739
accuracy			0.92	100886
macro avg	0.83	0.77	0.79	100886
weighted avg	0.91	0.92	0.91	100886

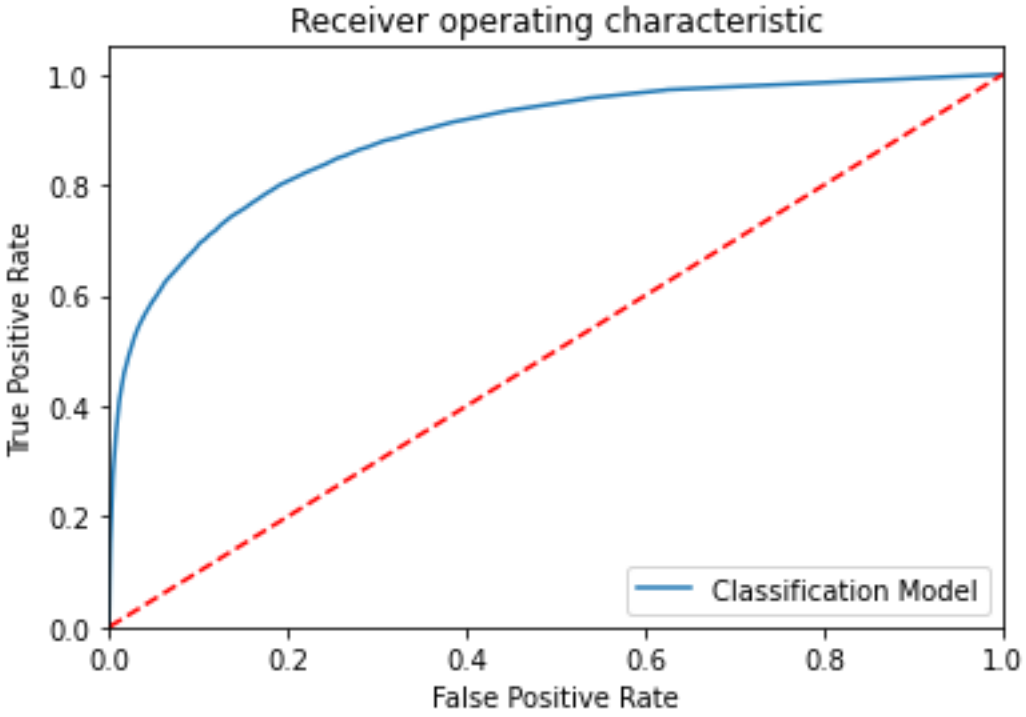
Accuracy : 91.5 %
Precision : 72.5 %
Recall : 56.5 %
F1 Score : 0.635
Specificity or True Negative Rate : 96.8 %
Balanced Accuracy : 76.6 %
roc_auc_score: 0.766



Model Name: BaggingClassifier(n_estimators=100)
Confusion matrix :
[[7260 5887]
[3129 84610]]
Outcome values :
7260 5887 3129 84610
Classification report :

	precision	recall	f1-score	support
1	0.70	0.55	0.62	13147
0	0.93	0.96	0.95	87739
accuracy			0.91	100886
macro avg	0.82	0.76	0.78	100886
weighted avg	0.90	0.91	0.91	100886

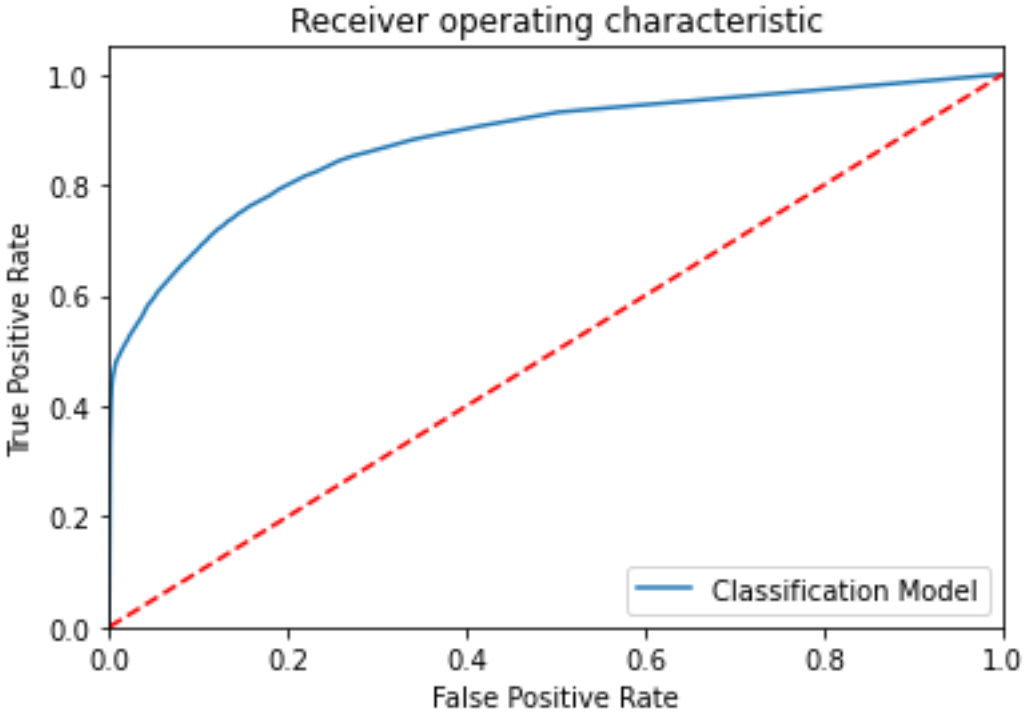
Accuracy : 91.1 %
Precision : 69.9 %
Recall : 55.2 %
F1 Score : 0.617
Specificity or True Negative Rate : 96.4 %
Balanced Accuracy : 75.8 %
roc_auc_score: 0.758



Model Name: ExtraTreesClassifier()
Confusion matrix :
[[7522 5625]
[3509 84230]]
Outcome values :
7522 5625 3509 84230
Classification report :

	precision	recall	f1-score	support
1	0.68	0.57	0.62	13147
0	0.94	0.96	0.95	87739
accuracy			0.91	100886
macro avg	0.81	0.77	0.79	100886
weighted avg	0.90	0.91	0.91	100886

Accuracy : 90.9 %
Precision : 68.2 %
Recall : 57.2 %
F1 Score : 0.622
Specificity or True Negative Rate : 96.0 %
Balanced Accuracy : 76.6 %
roc_auc_score: 0.766


Receiver operating characteristic
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Classification Model

5.0 References

Dataset: Kaggle.com

Originally, the dataset come from the CDC and is a major part of the Behavioral Risk Factor Surveillance System (BRFSS), which conducts annual telephone surveys to gather data on the health status of U.S. residents. As the CDC describes.

Data source: https://en.wikipedia.org/wiki/Healthcare_industry

https://www.medicaldevice-network.com/news/ai-heart-attack-prediction/

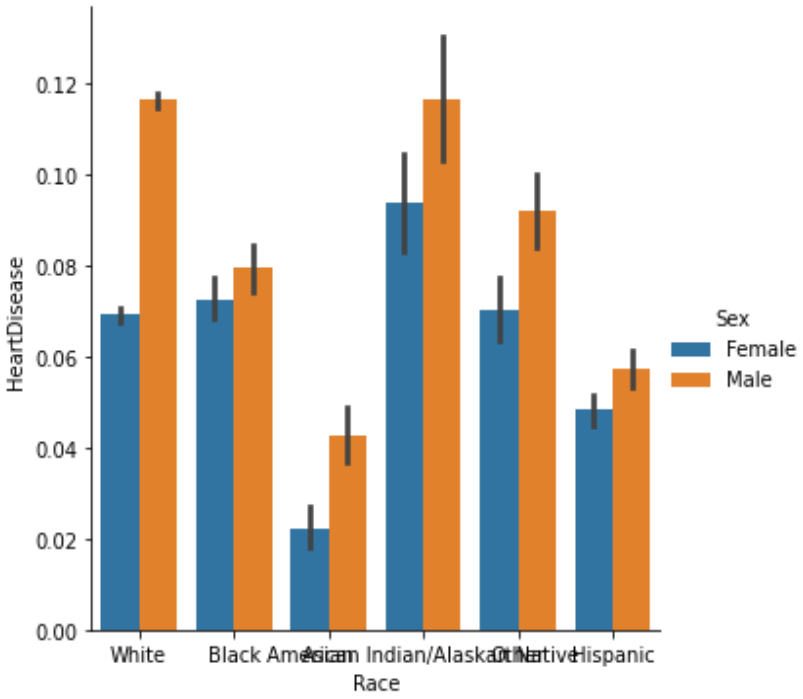
6.0 Appendices

6.1. Python code Results

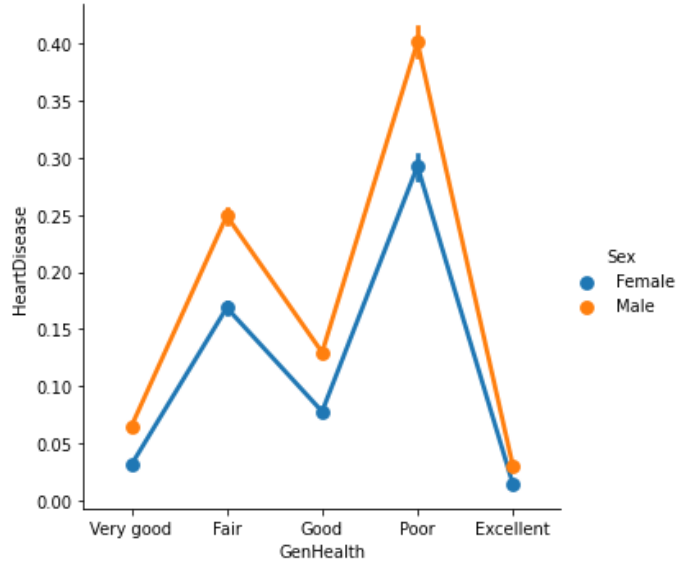
	Model Name	True Positive	False Negative	False Positive	True Negative	Accuracy	Precision	Recall	F1 Score	Specificity	MCC	ROC_AUC_Score	Balanced Accuracy
0	LogisticRegression()	2291	10856	1721	86018	0.875	0.571	0.174	0.267	0.98	0.266	0.577322647	0.577
1	DecisionTreeClassifier()	7685	5462	7847	79892	0.868	0.495	0.585	0.536	0.911	0.462	0.747554145	0.748
2	RandomForestClassifier()	7423	5724	2809	84930	0.915	0.725	0.565	0.635	0.968	0.594	0.766300046	0.766
3	ExtraTreesClassifier()	7522	5625	3509	84230	0.909	0.682	0.572	0.622	0.96	0.574	0.76607606	0.766
4	KNeighborsClassifier()	4460	8687	5088	82651	0.863	0.467	0.339	0.393	0.942	0.323	0.640625358	0.64
5	BaggingClassifier(n_estimators=100)	7260	5887	3129	84610	0.911	0.699	0.552	0.617	0.964	0.572	0.758277323	0.758
6	GradientBoostingClassifier(loss='deviance')	2441	10706	1705	86034	0.877	0.589	0.186	0.282	0.981	0.282	0.58311855	0.584
7	LGBMClassifier()	2555	10592	1724	86015	0.878	0.597	0.194	0.293	0.98	0.292	0.587345864	0.587
8	GaussianNB()	6241	6906	11117	76622	0.821	0.36	0.475	0.409	0.873	0.31	0.674001859	0.674
9	SVC(Probability=True)	5893	7683	4012	82643	0.801	0.532	0.339	0.521	0.901	0.234	0.518723456	0.532

6.2. List of Charts

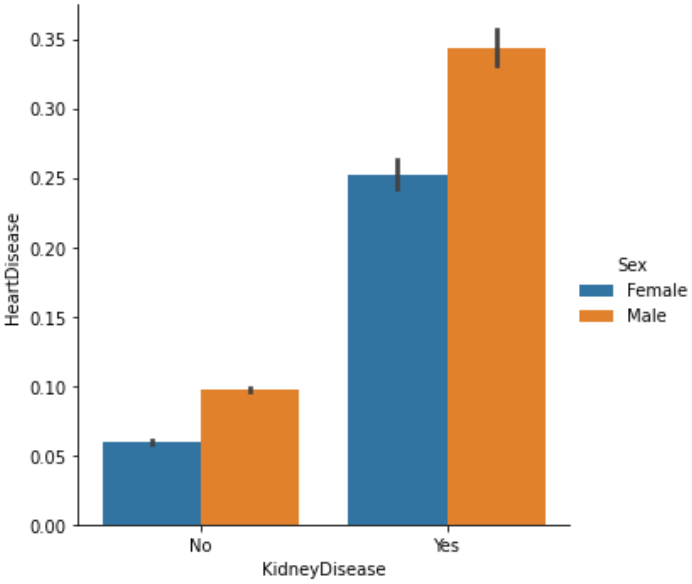
6.2.1 Chart 01: Heart Stroke according to races.



6.2.2 Chart 02: Heart stroke according to health conditions



6.2.3 Chart-03: Heart Stroke according to kidney disease



6.2.4:Chart-04: Heart Stroke According to Physical Activity