Table of Contents

Abstract

In Today,s generation Many Peoples's major concern is Health.This Project is concerned About Health by Taking various Peoples health aspects.The Aspects are BMI, whether they Drink alcohol, whether they smoke how much Do they workout.

In general, datasets which contain Health data can be used :

1. Prediction of the results of the HealthCare will tell whether the person is Healthy Or not
2. Prediction of These results will tell Whether the person have any serious Health Issues and Whether They undergo Major operations or not.

This dataset containing bank health data and we can use it to tell health condition of various People

In order to optimize healthcare data with the help of a dataset, we will have to take following steps:

1. Import data from datasets and perform initial high level analysis
2. Clean the Data
3. Use Machine Learning Techniques

Bank Marketing process steps:

- Collect The persons healthdata
- Analyse Them
- Make a Decision
- Deliver the results

CHAPTER 1
INTRODUCTION

With the increasing power of computer technology, companies and institutions can nowadays store large amounts of data at a reduced cost. The amount of available data is increasing exponentially and cheap disk storage makes it easy to store data that previously was thrown away. There is a huge amount of information locked up in databases that is potentially important but has not yet been explored. The growing size and complexity of the databases make it hard to analyze the data manually, so it is important to have automated systems to support the process. Hence there is a need for computational tools able to treat these large amounts of data and extract valuable information.

In this context, Data Mining provides automated systems capable of processing large amounts of data that are already present in databases. Data Mining is used to automatically extract important patterns and trends from databases seeking regularities or patterns that can reveal the structure of the data and answer business problems. Data mining includes learning techniques that fall into the field of Machine learning. The growth of databases in recent years brings data mining at the forefront of new business technologies.

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, domicile, etc. However, this contest focused on the relationship between claims and vehicle characteristics well as other characteristics associated with bank marketing policies.

1.1. What are the different types of Machine Learning?

How does Machine Learning work?

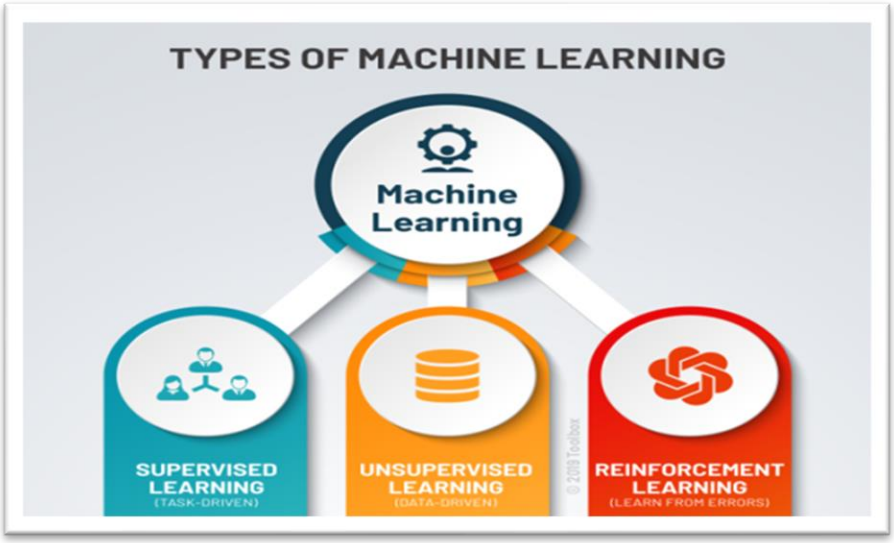

Fig 1.1 Types of machine learning

SUPERVISED LEARNING:

Data sets include their desired outputs or labels so that a function can calculate an error for any given prediction Data sets include their desired outputs or labels so that a function can calculate an error for any given prediction. The supervision part comes into play when a prediction is created, and an error is produced to change the function and learn the mapping. Supervised learning's goal is to create a function that effectively generalizes over data it has never seen.

Classification:

Classification predicts the class or type of an object according to a finite number ofoptions. The classification output variable is always a category.

- **Linear Models**
  - Logistic Regression
  - Support Vector Machines

- **Non-linear Models**
  - K-Nearest Neighbours
  - Naïve Bayes
  - Decision Tree Classification
  - Random Forest Classification
  - XGBoost
  - LightGBM

Regression Models:

Regression is a technique for investigating the relationship between independent variables or features and a dependent variable or outcome. It's used as a method for predictive modelling in machine learning, in which an algorithm is used to predict continuous outcomes.. Regression models get further split into:

- Linear Regression.
- Neural Network Regression.
- Decision Tree Regression.
- Random Forest.

Semi-supervised learning:

Semi-supervised learning is similar to supervised learning but instead uses both labelled and unlabelled data. Labelled data is essential 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.

UNSUPERVISED LEARNING:

There are cases where a data set doesn't have the desired output, so there's no means of supervising the function. Instead, the process tries to segment the data set into "classes" so that each class has a

segment of the data set with common features. Unsupervised learning aims to build a mapping function that classifies data based on features found within the data.

Clustering:

Clustering involves grouping sets of similar data (based on defined criteria) automatically without human intervention. It's useful for segmenting data into several groups and performing analysis on each data set to find patterns. This model involves gathering similar objects into groups.

- K Means Clustering
- Hierarchical Clustering
- Centroid-based Clustering.
- Density-based Clustering.
- Distribution-based Clustering.

Dimension reduction:

Dimension reduction reduces the number of variables being considered to find the exact information required.

- PCA
- LDA
- ICA

REINFORCEMENT LEARNING:

With reinforcement learning, the algorithm tries to learn actions for a given set of states that lead to a goal state. Thus, errors aren't flagged after each example but rather on receiving a reinforcement signal, like reaching the goal state. This process closely resembles human learning, where feedback isn't provided for every action, only when the situation calls for a reward.

1.2. Benefits of Using Machine Learning in HealthCare

Exploring the Advantages and Disadvantages of Machine Learning:

As health systems gain access to more data and invest in integration and interoperability infrastructure, ML will potentially reach all aspects of medicine. The most accurate ML models will typically come from organizations with big data sets and the supporting infrastructure, including a data platform and ML technology (e.g., **Healthcare.AI** by Health Catalyst). These tools are necessary to aggregate data from enterprise data warehouses, data platforms, and third-party data sources, then identify the relevant data inputs for ML

This process enables comparative effectiveness and research, resulting in unique, accurate algorithms.

The increase in healthcare data and accurate algorithms means ML-generated insights can reach more areas of medicine. For example, a primary care provider (PCP) treating a patient with hypertension could review ML-generated information during the clinic visit. The process could begin with an analysis of home-generated data from connected devices, including blood pressure readings, weight, fitness data, sleep data, compliance with medications, and nutritional data. Future wearable sensors may help collect additional data. An ML algorithm could combine that data with the health system's data, such as additional diagnosis, symptoms, lab tests, imaging, and genomic data. By integrating all available patient data in real time, ML will augment the PCP's ability to better understand the patient's current state and future health risks and enhance medical decision making to improve that patient's long-term outcomes.

This potential to augment care exists across all specialties of medicine as more data is available. For example, oncology will see advances in diagnosis with ML-augmented imaging and pathology, and ML's analysis of complex genetic data will improve clinical care and inform treatment. ML algorithms will be like an additional expert consultation, aggregating and informing oncologists with the latest clinical trial results across a broad spectrum of cancers, allowing easier access to newer treatment options and even helping refer patients to clinical trials with promising investigational drugs.

Machine learning-powered tools can sift through more data, including libraries of similar patients, diagnoses, and genetics, than one person can process. As such, ML opens data resources that include treatment options and predictions for each treatment's effectiveness, mortality rates, side effects, and cost. In this way, ML can put in infinitely more work behind the scenes, delivering real-time, accurate information to the point of care.

.

1.3. About Industry (HealthCare)

Many health systems already leverage ML in everyday clinical practice, advancing medicine into a new realm. While the benefits of ML to augment provider decision making seem endless, health systems first need to understand ML's role in healthcare and then invest in supporting tools and infrastructure. With this process, ML is becoming commonplace in healthcare

.

1.4 AI / ML Role in HealthCare:

1. AI supports medical imaging analysis

AI is used as a tool for case triage. It supports a clinician reviewing images and scans. This enables radiologists or cardiologists to identify essential insights for prioritizing critical cases, to avoid potential errors in reading electronic health records (EHRs) and to establish more precise diagnoses.

2. AI can decrease the cost to develop medicines

Supercomputers have been used to predict from databases of molecular structures which potential medicines would and would not be effective for various diseases. By using convolutional neural networks, a technology similar to the one that makes cars drive by themselves, AtomNet was able to predict the binding of small molecules to proteins by analyzing hints from millions of experimental measurements and thousands of protein structures.

3. AI analyzes unstructured data

Clinicians often struggle to stay updated with the latest medical advances while providing quality patient-centered care due to huge amounts of health data and medical records. EHRs and biomedical data curated by medical units and medical professionals can be quickly scanned by ML technologies to provide prompt, reliable answers to clinicians.

4. AI builds complex and consolidated platforms for drug discovery

AI algorithms are able to identify new drug applications, tracing their toxic potential as well as their mechanisms of action. This technology led to the foundation of a drug discovery platform that enables the company to repurpose existing drugs and bioactive compounds.

5. AI can forecast kidney disease

Acute kidney injury (AKI) can be difficult to detect by clinicians, but can cause patients to deteriorate very fast and become life-threatening. With an estimated 11% of deaths in hospitals following a failure to identify and treat patients, the early prediction and treatment of these cases can have a huge impact to reduce life-long treatment and the cost of kidney dialysis.

CHAPTER 2
HEALTH CARE

The data is related with healtjhcare of people.This data was collected from surveys and phone calls.
In Order to Check whether the person is healthy or not there are 40k+ rows and 16+ columns
The main factors for health check are smoke, alcohol, BMI , cardio

2.1 Main Drivers for AI Bank Marketing

Observational healthcare databases, such as administrative claims and electronic health records, present rich data sources for knowledge discovery from patient longitudinal histories. One such use case is the prediction of various events across the patient treatment journey, such as diagnosis and therapy initiation, progression or discontinuation.
If implemented well, patient event prediction models enable several applications in the commercial (predictive customer targeting, patient services design) and research (target patient universe determination, trial site selection) domains. However, owing to the richness, complexity and nuances in the data, there are several things to get right when it comes to model design. For instance, selection of right data set and sample size, length of medical history, prediction time window, modeling parameters, type of features (recency, frequency, sequence); and mechanism of feature generation (knowledge-driven vs. automatically generated).

2.1 Internship Project - Data Link

The internship project data has taken from Kaggle and the link is:

https://www.kaggle.com/datasets/drateendrajha/health-screening-data

CHAPTER 3

AI / ML MODELLING AND RESULTS

3.1 Your Problem Of Statement

Predictive models are most effective when they are constructed using a company'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 company throughout the process, as well as consideration of industry leading claims practices and benchmarks.

- Predictive modelling can be used to quantify the impact to the claims department resulting from the failure to meet or exceed claim service leading practices. It can also be used to identify the root cause of claim leakage. Proper use of predictive modelling will allow for potential savings across two dimensions:
- 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.

In simple terms, a data science life cycle is nothing but a repetitive set of steps that you need to take to complete and deliver a project/product to your client.

Although the data science projects and the teams involved in deploying and developing the model will be different, every data science life cycle will be slightly different in every other company.

However, most of data science projects happen to follow a somewhat similar process.

In order to start and complete a data science-based project, we need to understand the various roles and responsibilities of the people involved in building, and developing the project.

Let us look at those employees who are involved in a typical data science project:

Who Are Involved in The Projects:

- Business Analyst
- Data Analyst
- Data Scientists
- Data Engineer
- Data Architect
- Machine Learning Engineer

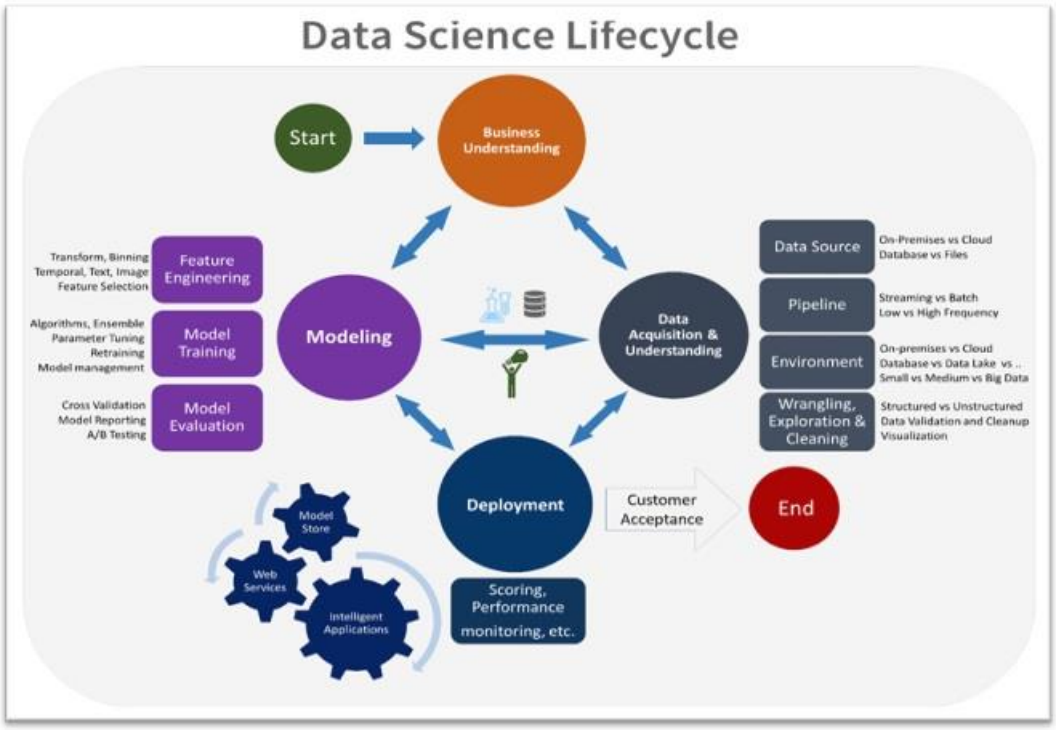

Fig 3.2 Data Science Life Cycle

3.2.1 Data Exploratory Analysis

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

3.2.2 Data Pre Processing

- Data preprocessing, a component of data preparation, describes any type of processing performed on raw data to prepare it for another data processing procedure. It has traditionally been an important preliminary step for the data mining process. More recently, data preprocessing techniques have been adapted for training machine learning models and AI models and for running inferences against them.
- Data preprocessing transforms the data into a format that is more easily and effectively processed in data mining, machine learning and other data science tasks. The techniques are generally used at the earliest stages of the machine learning and AI development pipeline to ensure accurate results.

3.2.2.1 Check The Duplicate and Low Variation Data

Types of Duplicate Features in Machine Learning:

Two things distinguish top data scientists from others in most cases: Feature Creation and Feature Selection. i.e., creating features that capture deeper/hidden insights about the business or customer and then making the right choices about which features to choose for your model.

1. Duplicate Values (Same value for each record)

2. Duplicate Index (value of two features are different but they occur at the same index)

Keeping duplicate features in your dataset introduces the problem of multicollinearity.

In the case of linear models, weights distribution between the two features will be problematic.

If you are using tree-based modes, it won't matter unless you are looking at feature importance.

In the case of distance-based models, it will make that feature count more in the distance.

3.2.2.2 Identify and Address The Missing Variables

- **What are the missing data?**

  Missing data are values that are not recorded in a dataset. They can be a single value missing in a single cell or missing of an entire observation (row). Missing data can occur both in a continuous variable (e.g., height of students) or a categorical variable (e.g., gender of a population).

  Missing data are common in any field of natural or behavioral science, but it is particularly commonplace in social sciences research data.

  In programming languages, missing values are represented as NA or Nan or simply an empty cell in an object.

- **The origins of missing data**

  So where do the missing values come from, and why do they even exist?

  Let's give an example. You are administering a questionnaire survey among a sample of respondents; and in the questionnaire, you are asking a question about household income. Now, what if a respondent refuses to answer that question? Would you make that up or rather leave the field empty? You'd probably leave that cell empty — creating an instance of missing value.

- **Problems caused**

  Missing values are undesirable, but it is difficult to quantify the magnitude of effects in statistical and machine learning projects. If it's a large dataset and a very small percentage of data is missing the effect may not be detectable at all.

However, if the dataset is relatively small, every data point counts. In these situations, a missing data point means loss of valuable information.

In any case, generally missing data creates imbalanced observations, cause biased estimates, and in extreme cases, can even lead to invalid conclusion.

Case deletion: if the dataset is relatively large delete the complete record with a missing value

Substitution: substitute missing cells with (a) column mean, (b) mean of nearest neighbors, (c) moving average, or (c) filling with the last observation

Statistical imputation: a regression can be an effective way to determine the value of missing cell given other information in the dataset

Sensitivity analysis: if the sample is small or missing values are relatively large then conduct a sensitivity analysis with multiple variations of outcomes.

For missing values:

TRAIN DATA:

```
from sklearn.impute import SimpleImputer

imputer_str=SimpleImputer(missing_values=np.nan,strategy='most_frequent',fill_value=None,verbose=0,copy=True,add_indicator=False)

data_test['Arrival Delay in Minutes']=imputer_str.fit_transform(data_test[['Arrival Delay in Minutes']])
```

TEST DATA:

```
from sklearn.impute import SimpleImputer

imputer_str=SimpleImputer(missing_values=np.nan,strategy='most_frequent',fill_value=None,verbose=0,copy=True,add_indicator=False)

data_test['Arrival Delay in Minutes']=imputer_str.fit_transform(data_test[['Arrival Delay in Minutes']])
```

Identify objects and convert them into numerical values:

Defining data types when reading a CSV file

Creating a custom function to convert the data type

as type () vs. to numeric ()

When doing data analysis, it is important to ensure correct data types. Otherwise, you may get unexpected results or errors. In the case of Pandas, it will correctly infer data types in many cases, and you can move on with your analysis without any further thought on the topic.

Despite how well pandas work at some point in your data analysis process you will likely need to explicitly convert data from one type to another. This article will discuss how to change data to a

numeric type. More specifically, you will learn how to use the Pandas built-in methods as type () and to numeric () to deal with the following common problems:

- Converting string/int to int/float
- Converting float to int
- Converting a column of mixed data types
- Handling missing values
- Converting a money column to float
- Converting Boolean to 0/1
- Converting multiple data columns at once

3.2.2.3 Handling of Outlier

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population.

There is, of course, a degree of ambiguity. Qualifying a data point as an anomaly leaves it up to the analyst or model to determine what is abnormal—and what to do with such data points.

- **There are also different degrees of outliers:**

Mild outliers lie beyond an “inner fence” on either side.

Extreme outliers are beyond an “outer fence.”

Why do outliers occur? According to Tom Barenberg, chief economist and data consultant at Unity Marketing, “It can be the result of measurement or recording errors, or the unintended and truthful outcome resulting from the set’s definition.”

Outliers may contain valuable information. Or be meaningless aberrations caused by measurement and recording errors. In any case, they can cause problems with repeatable A/B test results, so it’s important to question and analyze outliers.

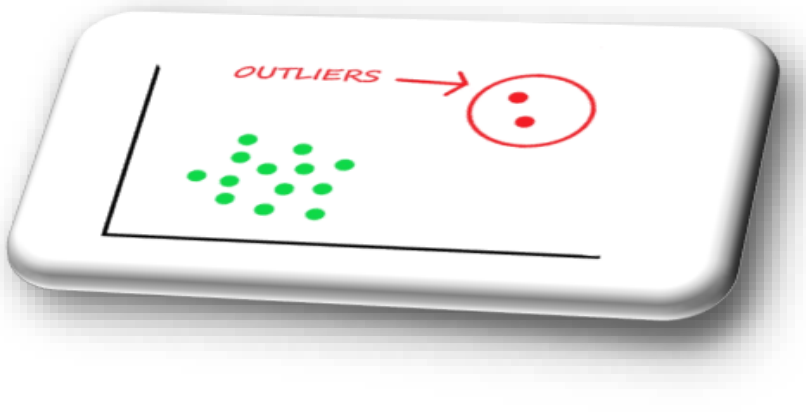

Fig 3.2.2.3 outliers

3.2.2.4 Categorical data and Encoding Techniques

What is Categorical Data?

Since we are going to be working on categorical variables in this article, here is a quick refresher on the same with a couple of examples. Categorical variables are usually represented as 'strings' or 'categories' and are finite in number. Here are a few examples:

1. The city where a person lives: Delhi, Mumbai, Ahmedabad, Bangalore, etc.

2. The department a person works in: Finance, Human resources, IT, Production.

3. The highest degree a person has: High school, Diploma, Bachelors, Masters, Ph.D.

4. The grades of a student: A+, A, B+, B, B- etc.

In the above examples, the variables only have definite possible values. Further, we can see there are two kinds of categorical data-

• Ordinal Data: The categories have an inherent order

• Nominal Data: The categories do not have an inherent order

Label Encoding:

• We use this categorical data encoding technique when the categorical feature is ordinal. In this case, retaining the order is important. Hence encoding should reflect the sequence.

• In Label encoding, each label is converted into an integer value. We will create a variable that contains the categories representing the education qualification of a person.

Binary Encoding:

• Binary encoding is a combination of Hash encoding and one-hot encoding. In this encoding scheme, the categorical feature is first converted into numerical using an ordinal encoder. Then the numbers are transformed in the binary number. After that binary value is split into different columns.

• Binary encoding works well when there are a high number of categories. For example, the cities in a country where a company supplies its products

3.2.2.5 Feature Scaling

Why Feature Scaling?

Real Life Datasets have many features with a wide range of values like for example let's consider the house price prediction dataset. It will have many features like no. of. bedrooms, square feet area of the house, etc.

As you can guess, the no. of bedrooms will vary between 1 and 5, but the square feet area will range from 500-2000. This is a huge difference in the range of both features.

Many machine learning algorithms that are using Euclidean distance as a metric to calculate the similarities will fail to give a reasonable recognition to the smaller feature, in this case, the number of bedrooms, which in the real case can turn out to be an important metric.

E.g.: Linear Regression, Logistic Regression, KNN

There are several ways to do feature scaling. I will be discussing the top 5 of the most used feature scaling techniques.

3.2.3 Selection of Dependent and Independent variables

The dependent or target variable here is satisfaction Target which tells us a

The independent variables are selected after doing exploratory data analysis. This tells us that whether customer is satisfied, neutral or di or ,satisfied.

3.2.4 Data Sampling

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

It can be performed using the 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.

Steps involved in Sampling:

The first stage in the sampling process is to clearly define the target population.

- So, to carry out opinion polls, polling agencies consider only the people who are above 18 years of age and are eligible to vote in the population. Sampling Frame – It is a list of items or people forming a population from which the sample is taken.
- So, the sampling frame would be the list of all the people whose names appear on the voter list of a constituency.
- Generally, probability sampling methods are used because every vote has equal value and any person can be included in the sample irrespective of his caste, community, or religion. Different samples are taken from different regions all over the country.
- Sample Size – It is the number of individuals or items to be taken in a sample that would be enough to make inferences about the population with the desired level of accuracy and precision
- Once the target population, sampling frame, sampling technique, and sample **size** have been established, the next step is to collect data from the sample.

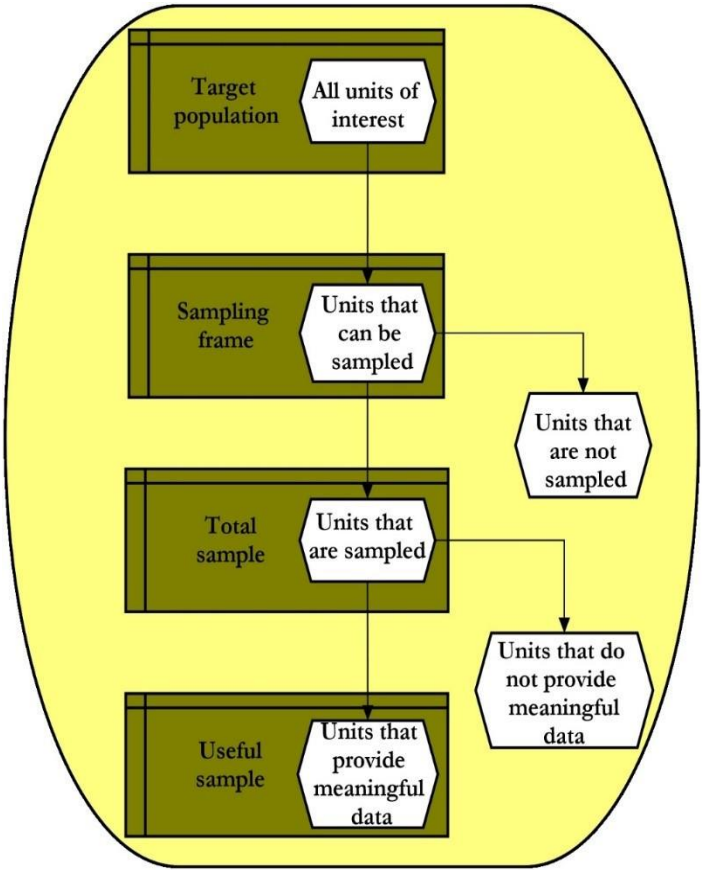

Fig 3.2.4.2 Steps involved in sampling

3.2.5 Models Used for Development

3.2.5.1 Model 01(Logistic regression)

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

3.2.5.2 Model 02(Decision Tree Classifier)

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed based on features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because, like a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

3.2.5.3 Model 03(Random Forest Classifier)

A random forest is an algorithm that consists of many decision trees. It was first developed by Leo Bierman 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 a collection of decision trees with controlled variance. The instance's class is to the class with the highest number of votes, the class that occurs the most within the leaf in which the 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.

3.2.5.4 Model 04(Extra Trees Classifier)

This class implements a meta estimator that fits several randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

3.2.5.5 Model 05(KNN Classifier)

K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on the Supervised Learning technique K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most like the available categories'-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a good suite category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for Classification problems K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much like the new data.

3.2.5.6 Model 06(Naïve Bayes)

Naïve Bayes is a probabilistic machine learning algorithm used for many classification functions and is based on the Bayes theorem. Gaussian Naïve Bayes is the extension of naïve Bayes. While other functions are used to estimate data distribution, Gaussian or normal distribution is the simplest to implement as you will need to calculate the mean and standard deviation for the training data.

3.2.5.7 Model 07(XG Boost)

XG Boost is an implementation of Gradient Boosted decision trees. This library was written in C++. It is a type of Software library that was designed basically to improve speed and model performance. It has recently been dominating in applied machine learning. XG Boost models majorly dominate in many Kaggle Competitions. In this algorithm, decision trees are created in sequential form. Weights play an important role in XG Boost. Weights are

assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and the variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

3.2.5.8 Model 08(Light GBM)

Light GBM is a gradient boosting framework based on decision trees to increases the efficiency of the model and reduces memory usage.

It uses two novel techniques: Gradient-based One Side Sampling and Exclusive Feature Bundling (EFB) which fulfills the limitations of histogram-based algorithm that is primarily used in all GBDT (Gradient Boosting Decision Tree) frameworks. The two techniques of GOSS and EFB described below form the characteristics of Light GBM Algorithm. They comprise together to make the model work efficiently and provide it a cutting edge over other GBDT frameworks.

Gradient-based One Side Sampling Technique for Light GBM:

Different data instances have varied roles in the computation of information gain. The instances with larger gradients (i.e., under-trained instances) will contribute more to the information gain. GOSS keeps those instances with large gradients (e.g., larger than a predefined threshold, or among the top percentiles), and only randomly drop those instances with small gradients to retain the accuracy of information gain estimation. This treatment can lead to a more accurate gain estimation than uniformly random sampling, with the same target sampling rate, especially when the value of information gain has a large range.

3.2.5.9 Model 09 (SVC)

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression

problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

3.2.5.10 Model 10 (K-Means)

k-means clusterings an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

3.3 AI / ML Models Analysis 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

This section in which we used different types of model code as follows

```
# Build the Calssification models 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 sklearn.svm import SVC
```

```
from sklearn.ensemble import BaggingClassifier

from sklearn.ensemble import GradientBoostingClassifier
import lightgbm as lgb

# Create objects of classification algorithm with default hyper-parameters

ModelLR = LogisticRegression()
ModelDC = DecisionTreeClassifier()
ModelRF = RandomForestClassifier()
ModelET = ExtraTreesClassifier()
ModelKNN = KNeighborsClassifier(n_neighbors=5)
ModelSVM = SVC(probability=True)

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)

ModelGB = GradientBoostingClassifier(loss='deviance', learning_rate=0.1, n_estimators=100, subsample=1.0,
                                     criterion='friedman_mse', min_samples_split=2, min_samples_leaf=1,
                                     min_weight_fraction_leaf=0.0, max_depth=3, min_impurity_decrease=0.0,
                                    init=None, random_state=None,
                                     max_features=None, verbose=0, max_leaf_nodes=None, warm_start=False,
```

```
                validation_fraction=0.1, n_iter_no_change=None, tol=0.0001,
ccp_alpha=0.0)
    ModelLGB = lgb.LGBMClassifier()
    ModelGNB = GaussianNB()

    # Evalution matrix for all the algorithms

    MM = [ModelLR, ModelDC, ModelRF, ModelET, ModelKNN, ModelSVM, modelBAG,
ModelGB, ModelLGB, ModelGNB]
    for models in MM:

        # Fit the model

        models.fit(x_train, y_train)

        # Prediction

        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


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, predicted), 3))


# ROC Curve


from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(actual, predicted)
fpr, tpr, thresholds = roc_curve(actual, models.predict_proba(x_test)[:,1])
plt.figure()
# plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot(fpr, tpr, label= 'Classification Model' % logit_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, predicted),
               'Balanced Accuracy':balanced_accuracy}
    EMResults = EMResults.append(new_row, ignore_index=True)
    #-------------------------------------------------------------------------------------------------------
```

Logistic Classifier Code

```
# To build the 'Logistic Regression' model with random sampling (Hyper parameter tuning with GridSearchCV)

from sklearn.linear_model import LogisticRegression

# Create model object

ModelLR = LogisticRegression(penalty='l2', dual=False, tol=0.0001, C=100, fit_intercept=True,
```

```
                intercept_scaling=1, class_weight='balanced', random_state=None,
solver='liblinear',
                max_iter=100, multi_class='auto', verbose=0, warm_start=False,
                n_jobs=None, l1_ratio=None)

    # Fit the model

    ModelLR.fit(x_train, y_train)

    # Predict the model with test data set

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

    # 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

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(y_test, y_pred), 3))

# ROC Curve

from sklearn.metrics import roc_auc_score
```

```
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(y_test, y_pred)
fpr, tpr, thresholds = roc_curve(y_test,ModelLR.predict_proba(x_test)[:,1])
plt.figure()
# plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot(fpr, tpr, label= 'Classification Model' % logit_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' : ModelLR,
        '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(y_test, y_pred),
        'Balanced Accuracy':balanced_accuracy}
BDResults = BDResults.append(new_row, ignore_index=True)
#-----------------------------------------------------------------------------------------------------
```

Gradient Boosting Classifier

```
from sklearn.ensemble import BaggingClassifier

Modelbg2 = GradientBoostingClassifier(max_depth=9, min_samples_leaf=30, min_samples_split=1000)

# Train the model with train data

Modelbg2.fit(x_train,y_train)

# Predict the model with test data set

y_pred = Modelbg2.predict(x_test)
y_pred_prob = Modelbg2.predict_proba(x_test)

# 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

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, predicted), 3))


# ROC Curve


from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
logit_roc_auc = roc_auc_score(actual, predicted)
fpr, tpr, thresholds = roc_curve(actual, Modelbg2.predict_proba(x_test)[:,1])
plt.figure()
# plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
plt.plot(fpr, tpr, label= 'Classification Model' % logit_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' : Modelbg2,
           '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, predicted),
        'Balanced Accuracy':balanced_accuracy}
EMResults = EMResults.append(new_row, ignore_index=True)
```

CHAPTER 4

CONCLUSIONS AND FUTURE WORK

So far, the research offered some important insights into the Health Sector .

Among the top factors affecting peoples health People who smoke and drink alcohol have Major health issues. Because Smoking Directly effects the lungs.people who have more BMI have more health issues when compared to people with low BMI.

This research has significant implications for health , as it allows people to focus on the specific factors affecting Heealth. Moreover, it helps us to improve people health aspects of people and makes them live a healthy life.

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

1) Logistic Regression
2) Decision Tree Classifier
3) Decision Tree Classifier Max

We recommend model – **Logistic Regression Classifier** as a best fit for the given dataset. We considered Random Forest because it uses bootstrap aggregation which can reduce bias and variance in the data and can leads to good predictions with HealthCare Prediction.

Extra tress classifier Model got an accuracy of about 90.6% and got a precision and got RocAuc curve of 76.3 in HealthCare

	Model Name	True_Positive	False_Negative	False_Positive	True_Negative	Accuracy	Precision	Recall	F1 Score	Specificity
0	LogisticRegression(max_iter=3000)	409	788	414	11181	0.906	0.497	0.342	0.405	0.964
1	DecisionTreeClassifier()	590	607	1398	10197	0.843	0.297	0.493	0.37	0.879
2	(DecisionTreeClassifier(max_features='sqrt', r...	652	545	794	10801	0.895	0.451	0.545	0.493	0.932
3	(ExtraTreeClassifier(random_state=214727222), ...	595	602	766	10829	0.893	0.437	0.497	0.465	0.934
4	KNeighborsClassifier()	38	1159	149	11446	0.898	0.203	0.032	0.055	0.987
5	SVC(probability=True)	0	1197	0	11595	0.906	NaN	0.0	0.0	1.0
6	(DecisionTreeClassifier(random_state=508666280...	630	567	862	10733	0.888	0.422	0.526	0.469	0.926
7	([DecisionTreeRegressor(criterion='friedman_ms...	633	564	717	10878	0.9	0.469	0.529	0.497	0.938
8	LGBMClassifier()	713	484	776	10819	0.902	0.479	0.596	0.531	0.933
9	GaussianNB()	231	966	351	11244	0.897	0.397	0.193	0.26	0.97

Fig 4 Result

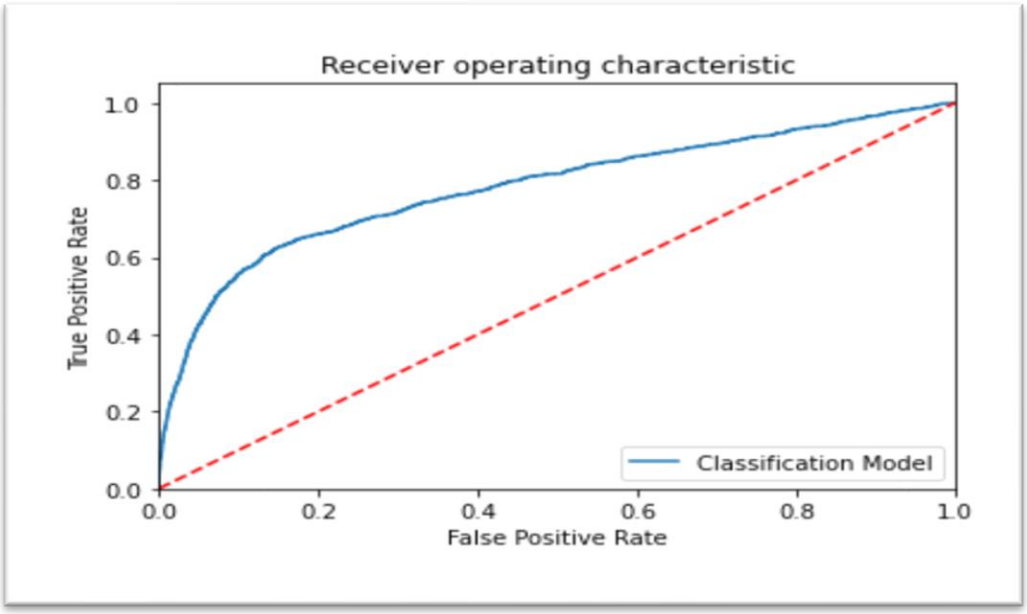

Fig 5 . Logistic Regression

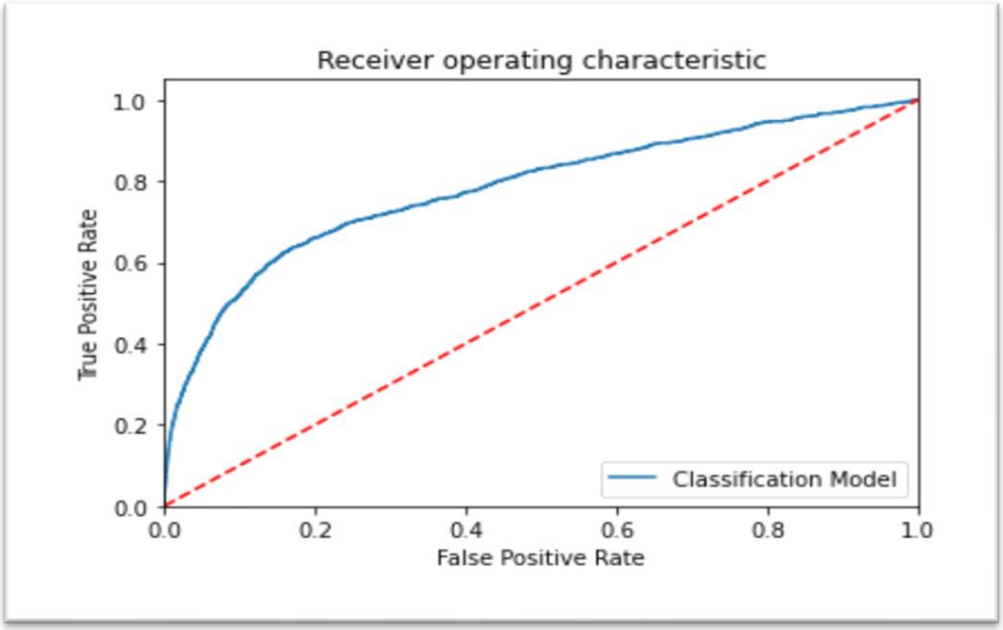

Fig 6 . Decision Tree Classifier

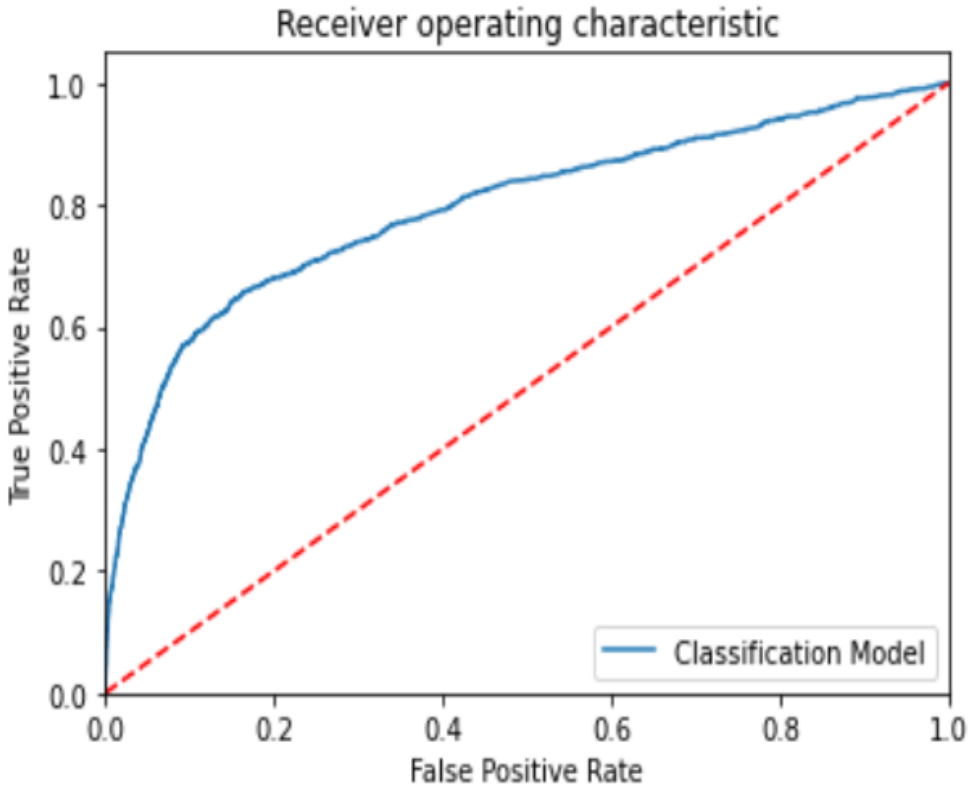

Fig 7. Decision Tree Classifier Max

CHAPTER 5

REFERENCES

Anyan, F. (2013). The influence of power shifts in data collection and analysis stages: A focus on qualitative research interview. The Qualitative Report, 18(18), 1.

Chen, C. F. (2008). Investigating structural relationships between service quality, perceived value, satisfaction, and behavioral intentions for air passengers: Evidence from Taiwan. Transportation Research Part A: Policy and Practice, 42(4), 709-717.

Cho, W., Windle, R. J., & Dresner, M. E. (2017). The impact of operational exposure and value-of-time on customer choice: Evidence from the airline industry. Transportation Research Part A: Policy and Practice, 103, 455–471.

https://www.javatpoint.com/types-of-machine-learning

https://www.kaggle.com

https://www.kaggle.com/datasets/drateendrajha/health-screening-data

APPENDICES

- Kernel methods
- Graphical models
- Reinforcement learning
- Convex analysis
- Optimization
- Bio informatics
- Minimal description length principle
- Topics in information theory
- Decision theory
- Network algorithms
- Computational Social Science
- Natural Language Processing Recent trends in deep learning and representation learning
- Facial recognition
- Voice Recognition
- Virtual Personal Assistants

A.1 Python Code Result

	Model Name	True_Positive	False_Negative	False_Positive	True_Negative	Accuracy	Precision	Recall	F1 Score	Specificity
0	LogisticRegression(max_iter=3000)	409	788	414	11181	0.906	0.497	0.342	0.405	0.964
1	DecisionTreeClassifier()	590	607	1398	10197	0.843	0.297	0.493	0.37	0.879
2	(DecisionTreeClassifier(max_features='sqrt', r...	652	545	794	10801	0.895	0.451	0.545	0.493	0.932
3	(ExtraTreeClassifier(random_state=214727222), ...	595	602	766	10829	0.893	0.437	0.497	0.465	0.934
4	KNeighborsClassifier()	38	1159	149	11446	0.898	0.203	0.032	0.055	0.987
5	SVC(probability=True)	0	1197	0	11595	0.906	NaN	0.0	0.0	1.0
6	(DecisionTreeClassifier(random_state=508666280...	630	567	862	10733	0.888	0.422	0.526	0.469	0.926
7	([DecisionTreeRegressor(criterion='friedman_ms...	633	564	717	10878	0.9	0.469	0.529	0.497	0.938
8	LGBMClassifier()	713	484	776	10819	0.902	0.479	0.596	0.531	0.933
9	GaussianNB()	231	966	351	11244	0.897	0.397	0.193	0.26	0.97

Fig A.1 python code

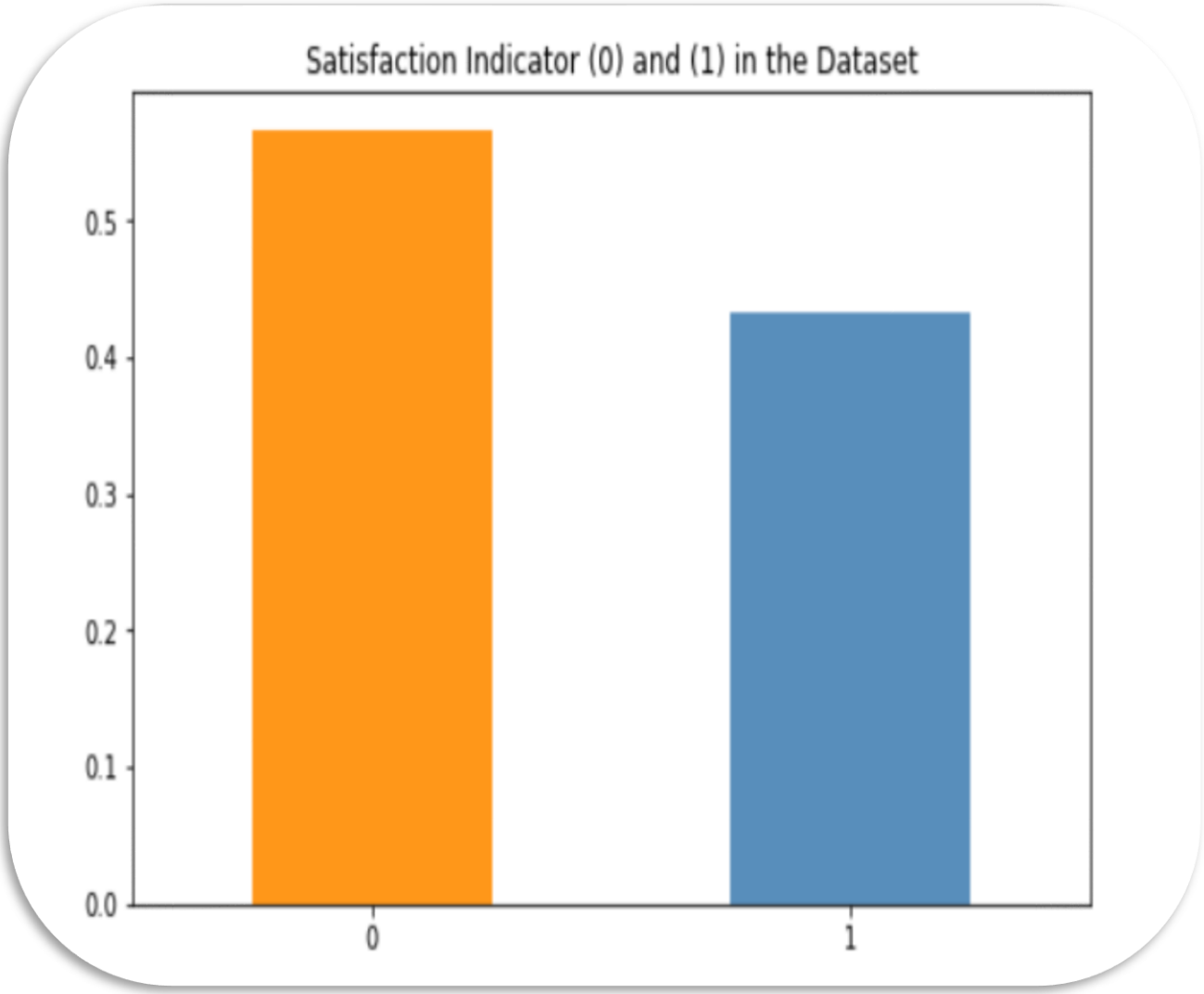

Fig A.2 Bar Graph

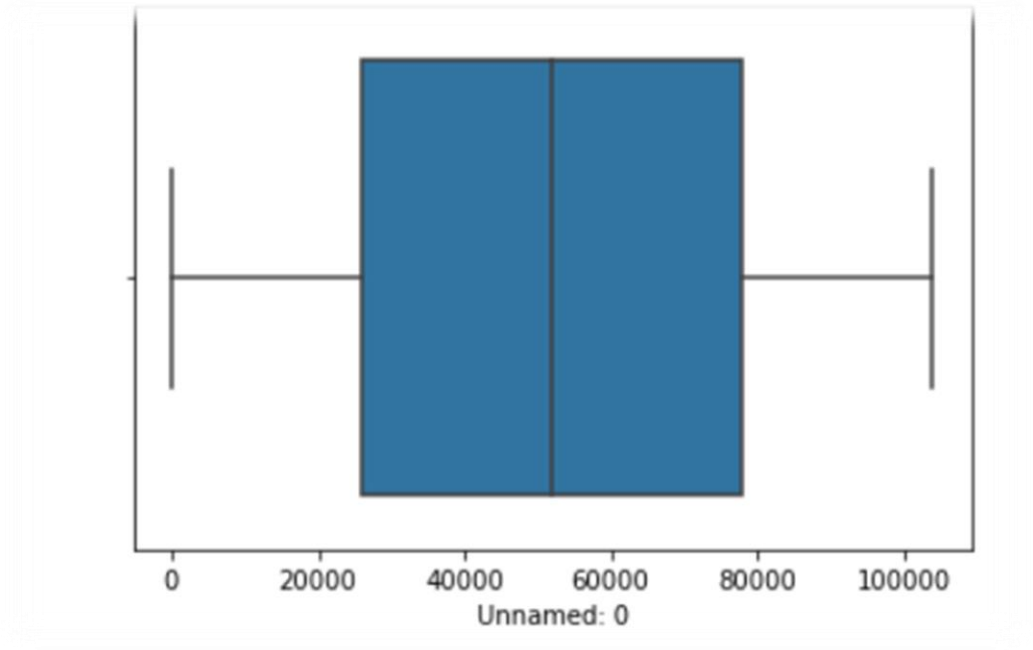

Fig A.2.3 Box plot1

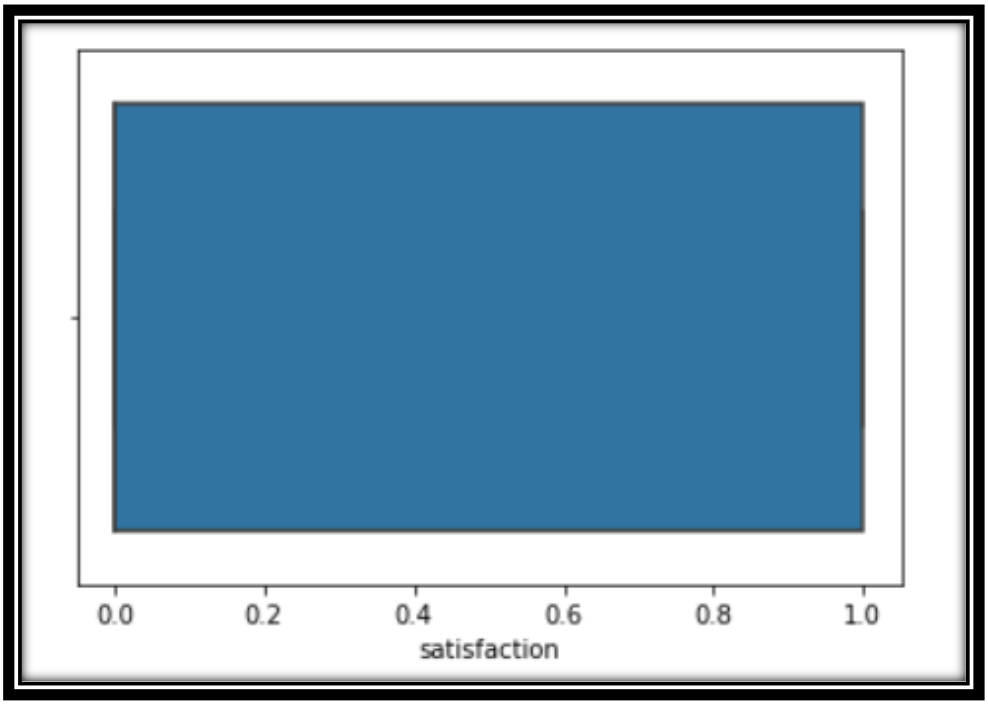

Fig A.2.4 Box Plot2

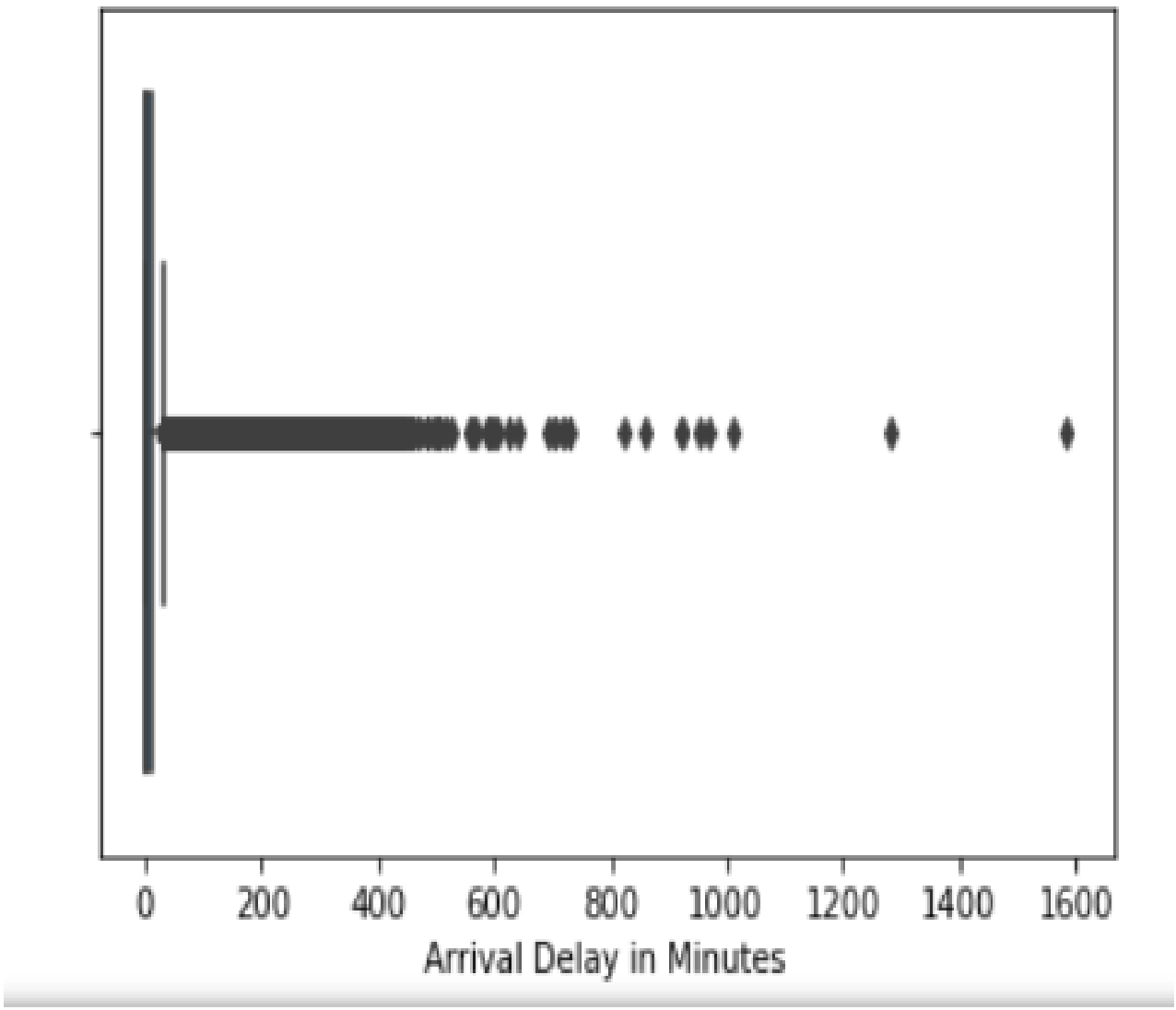

Fig A.2.5 Outliers

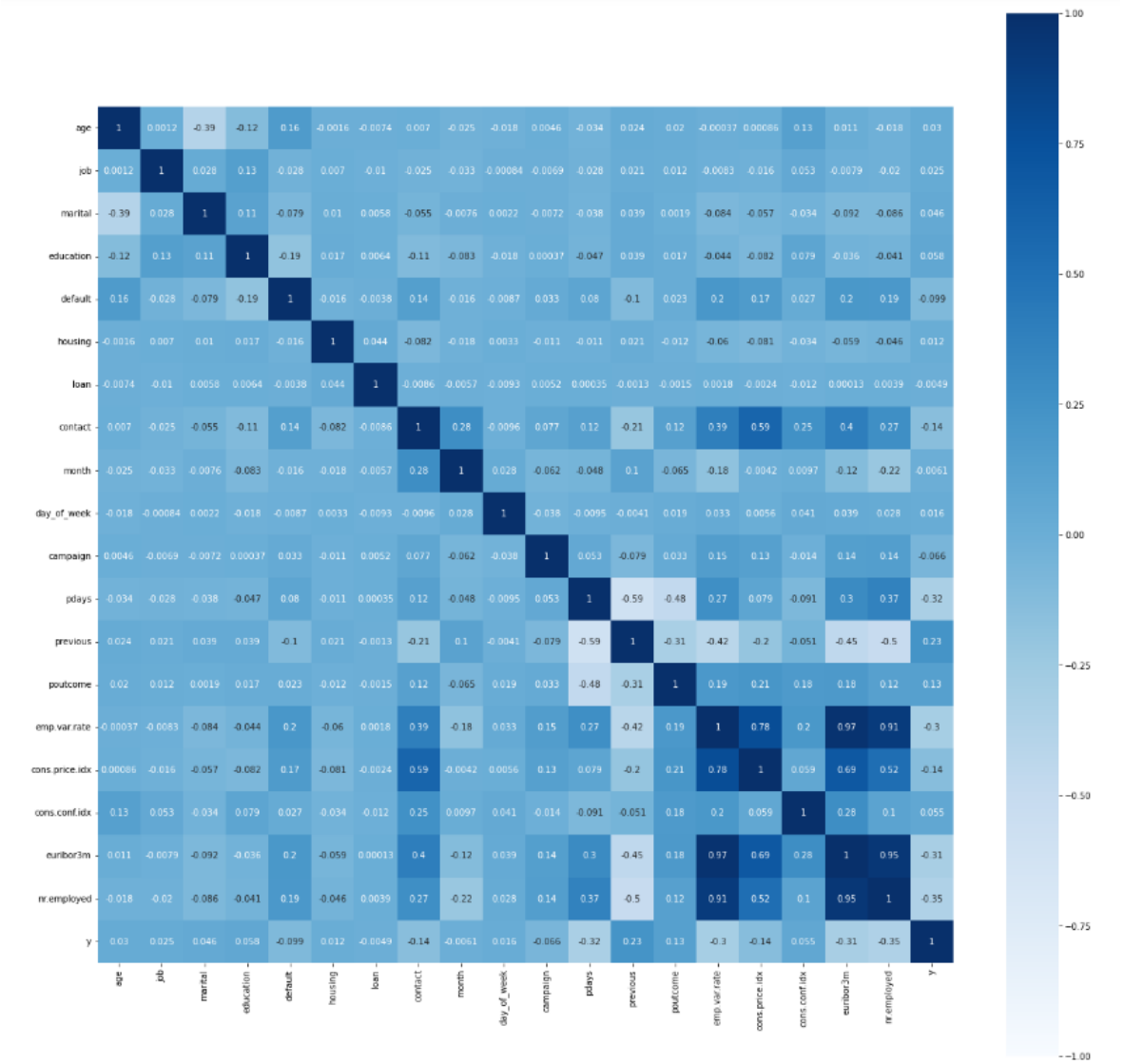

Fig A.2.6 Heat Map