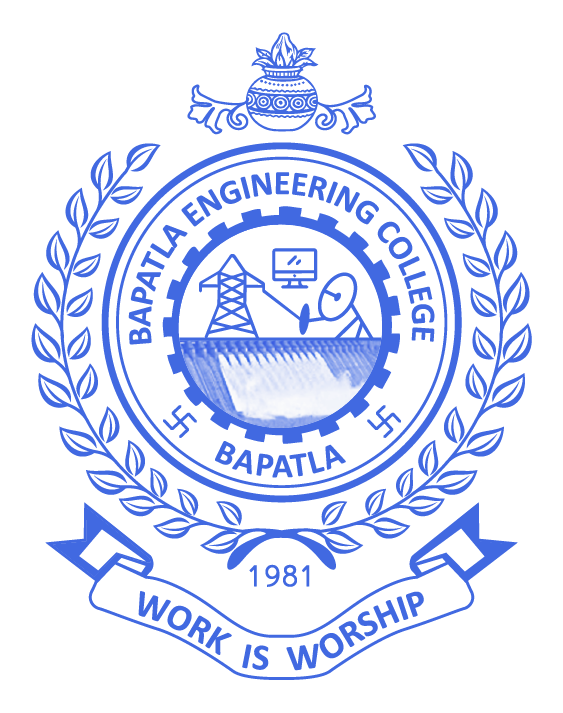
**The Project Report on**

# Analyzing Customers PERSONALITY FOR Business Improvement using Machine Learning

**Submitted To**

**Acharya Nagarjuna University**

****

**Under The Esteemed Guidance Of**

**Sri N. KIRAN KUMAR M. Tech**

**Assistant Professor**

**DEPARTMENT OF MCA**

**BAPATLA ENGINEERING COLLEGE**

**(Approved by A.I.C.T.E)**

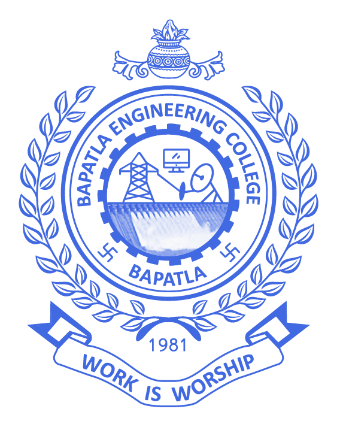
**(Affiliated to ACHARYA NAGARJUNA UNIVERSITY)**

**2019-2021**

**DEPARTMENT OF MCA**

**BAPATLA ENGINEERING COLLEGE**

**BAPATLA-522101**

****

**CERTIFICATE**

This is to certify that this project work entitled “**Secure Cloud Storage based on RLWE Problem”** is the bonafide work carried out by **ARUNURU NAVEEN**, **Reg.No: L20MC23014** submitted in Partial fulfillment of the requirement for the Award of Degree of “**Master of** **Computer Applications**”, during the academic year 2019-2021.

The results submitted in this project have been verified and are found to be satisfactory. The results embodied in this thesis have not been submitted to any other university for the award of the any other degree/diploma.

**Project Guide: Head of the Department:**

**Sri N. Kiran Kumar M. Tech Sri K.N. Prasad M. Tech**

**Assistant Professor Associate Professor & Head**

**Dept. of MCA Dept. of MCA**

**Bapatla Engineering College Bapatla Engineering College**

**Bapatla. Bapatla.**

**Signature of External Examiner**

# ACKNOWLEDGEMENT

I sincerely thank the following distinguished personalities who have given their advice and support for successful completion of this work.

I would like to thank **Dr. V. Damodara Naidu**, Principal, Bapatla Engineering College, Bapatla for providing the online resources and allowing me to carry out this work.

I extended my sincere thanks to **Sri K.N. Prasad**, Head of the Department for extending their cooperation and providing the required resources.

I am deeply indebted to most respect and my project guide **Sri N. Kiran Kumar**, Assistant Professor Dept. of MCA for his valuable and inspiring guidance, comments, suggestions and encouragements.

I extended my sincere thanks to all other teaching and non-teaching staff of the department who helped directly or indirectly for their cooperation and encouragement.

**ARUNURU NAVEEN**

**(L20MC23014)**

**DECLARATION**

This is to declare that the project **“Secure Cloud Storage based on RLWE Problem”** at Bapatla Engineering College has been presented by me during the academic year **2019-2021** in partial fulfillment of the requirements for the **“Master of Computer Application”**.

I also declare that this project is the result of my own efforts and that it has not been submitted to any other universities for the award of degree or diploma.

**ARUNURU NAVEEN**

**(L20MC23014)**

Table of Contents

[ACKNOWLEDGEMENT iii](#_Toc80180563)

[ABSTRACT viii](#_Toc80180564)

[CHAPTER-1 1](#_Toc80180565)

[INTRODUCTION 1](#_Toc80180566)

[1.1 INTRODUCTION 1](#_Toc80180567)

[CHAPTER- 2 5](#_Toc80180568)

[LITERATURE SURVEY 5](#_Toc80180569)

[2.1 LITERATURE REVIEW 5](#_Toc80180570)

[CHAPTER-3 7](#_Toc80180571)

[THEORETICAL BACKGROUND 7](#_Toc80180572)

[3.1 INTRODUCTION: 7](#_Toc80180573)

[3.2 INTRODUCTION TO PYTHON 12](#_Toc80180574)

[3.3 BENFITS OF PYTHON 21](#_Toc80180575)

[CHAPTER-4 30](#_Toc80180576)

[SYSTEM ANALYSIS 30](#_Toc80180577)

[4.1 EXISTING SYSTEM: 30](#_Toc80180578)

[4.1.1 DISADVANTAGES OF EXISTING SYSTEM: 30](#_Toc80180579)

[4.2 PROPOSED SYSTEM: 30](#_Toc80180580)

[4.2.1 ADVANTAGES OF PROPOSED SYSTEM: 31](#_Toc80180581)

[CHAPTER- 5 32](#_Toc80180582)

[SYSTEM DESIGN 32](#_Toc80180583)

[5.1 INTRODUCTION 32](#_Toc80180584)

[5.2 MODULES 32](#_Toc80180585)

[5.2.1 DATASET: 32](#_Toc80180586)

[5.2.2 PREPROCESSING: 32](#_Toc80180587)

[5.2.3 GRAPHS: 32](#_Toc80180588)

[5.2.4 PREDICTION: 32](#_Toc80180589)

[5.3 SYSTEM ARCHITECTURE 33](#_Toc80180590)

[5.4 UML DAIGRAMS 34](#_Toc80180591)

[5.4.1 CONSTRUCTION OF USE CASE DIAGRAMS: 37](#_Toc80180592)

[5.4.2 SEQUENCE DIAGRAMS: 40](#_Toc80180593)

[5.4.3 CLASS DIAGRAM: 42](#_Toc80180594)

[5.4.4 ACTIVITY DIAGRAM: 42](#_Toc80180595)

[CHAPTER-6 44](#_Toc80180596)

[SYSTEM REQUIREMENTS 44](#_Toc80180597)

[6.1 SYSTEM REQUIREMENTS 44](#_Toc80180598)

[6.1.1 HARDWARE REQUIREMENTS: 44](#_Toc80180599)

[6.1.2 SOFTWARE REQUIREMENTS: 44](#_Toc80180600)

[CHAPTER-7 45](#_Toc80180601)

[SYSTEM IMPLEMENTATION 45](#_Toc80180602)

[7.1 INPUT AND OUTPUT DESIGNS 45](#_Toc80180603)

[7.1.1 LOGICAL DESIGN 45](#_Toc80180604)

[7.1.2 PHYSICAL DESIGN 45](#_Toc80180605)

[7.2 INPUT & OUTPUT REPRESENTATION 46](#_Toc80180606)

[7.2.1 INPUT DESIGN 46](#_Toc80180607)

[7.2.2 OBJECTIVES 47](#_Toc80180608)

[7.2.3 OUTPUT DESIGN 47](#_Toc80180609)

[CHAPTER-8 69](#_Toc80180610)

[SYSTEM TESTING 69](#_Toc80180611)

[8.1 INTRODUCTION: 69](#_Toc80180612)

[8.2 LEVELS OF TESTING 69](#_Toc80180613)

[8.2.1 BLACK BOX TESTING 70](#_Toc80180614)

[8.2.2 WHITE BOX TESTING 72](#_Toc80180615)

[CHAPTER-9 74](#_Toc80180616)

[OUTPUT SCREENS 74](#_Toc80180617)

[CONCLUSION 75](#_Toc80180618)

[REFERENCES 76](#_Toc80180619)

**List of Figures**

**Name of the figure Pg.no**

[Figure 3. 1 Structure of cloud computing 7](#_Toc80179715)

[Figure 3. 2 Characteristics of cloud computing 9](#_Toc80179716)

[Figure 3. 3 Structure of service models 10](#_Toc80179717)

[Figure 5. 1 System Architecture 33](#_Toc80179735)

[Figure 5. 2 Use Case Diagram 39](#_Toc80179736)

[Figure 5. 3 Sequence diagram 41](#_Toc80179737)

[Figure 5. 4 Class Diagram 42](#_Toc80179738)

[Figure 5. 5 Activity Diagram 43](#_Toc80179739)

# ABSTRACT

 comprehensive investigation of a company's ideal clients is known as a customer personality analysis. It aids a company's understanding of its consumers and makes it simpler to change goods to meet the demands, habits, and concerns of various sorts of customers. Customer personality analysis enables a company to adapt its product depending on the preferences of its target customers from various customer categories. Instead of paying money to promote a new product to every client in a firm's database, for example, a company may determine which customer group is most likely to buy the product and then market it to that segment solely. Hence, the main motive for the paper is to find the accuracy of the prediction of the personality of the customer who is shopping and improve the research out there using the ensemble technique..

.

.

# CHAPTER-1

# INTRODUCTION

## Introduction

Doing business nowadays is not just waiting in a store and selling items. Day by day customers and their needs are also changing, and so the type of products according to them used to be changed.So, to overcome this problem we take help of Machine Learning algorithms to know more about customers and their needs. By using ML models which can predict a customer's needs depending upon their activities in a store or shopping mall and also on social media.For example, Instagram provides people to promote their business online and provide necessary tools. Tools may include paid promotion of their business, insights to monitor the post and stories activities such as how many people read their story or viewed their post.But now a question arises how this prediction thing works. There are algorithms like ANN, XGBoost and SVN in Machine Learning which works upon the datasets collected and based upon the algorithms make predictions about the customers and likes and dislikes and displays them products according to it.Even prediction can also be made through the observation of customers by the security cameras in a store by observing how customers pretend on seeing a product and how time do they spend on

seeing that particular product and also, they ask for the feedback from the customers. This helps us to gain more data about those products.Similar methods are used by almost everyone.

. The focus of modern grocery superstore business has beenshifted to the customer-centric organization. Customers are themost important factor for a business. Some customers can helpthe business to generate more proﬁt compared to the others.A loyalty-prone customer intends to stay with the supplierwho can provide the quality products. On the other hand, adeal-prone customer will always look for a better offer froma competitor. Customers can be classiﬁed as proﬁtable andunproﬁtable [1].Bringing a new customer to the business is cost-intensive.Because it involves various marketing strategies without hav-ing prior knowledge about the customer. However, a businesscan easily apply speciﬁc marketing strategies based on pre-vious customers information, when it approaches it’s existingcustomers. According to the study of Reichheld and Teal [2],a business can increase its proﬁt up to 95% by increasing ﬁvepercent customer retention. So, businesses are more focused onbuilding relationships with their customers. Companies mustlearn about their customers purchase preferences to build along-term relationship with them.Data accumulation is highly increased in recent years whichleads to a great interest in the ﬁeld of machine learning (ML).The computer has proven its usefulness in predicting andpattern ﬁndings by learning the data. The capabilities of ma-chine learning (ML) were extensively explored and signiﬁcantsuccess were achieved in the ﬁeld of language translation [3],[4], self-driving [5], text recognition [6], image recognition[7], [8] and voice recognition [9], [10]. Modern organizationsare combining technology with business policies to gain acompetitive advantage [11]. This has resulted in a competitionof ﬁnding relevant talent to gain the advantage in technology.People expertise in machine learning and data science arein high demand with the technology itself evolving everyday. In recent times, giant companies are spending a rigorousamount of money to gain the technological advantage overtheir competitors. Studies done in 2015 shows that companieslike Google, Facebook and Amazon had spent over USD $8.5billion to acquire companies working on Artiﬁcial Intelligence.This is an increase of 400% since 2010.Grocery superstore business has been accumulating hugedata from customers. Due to the digital revolution, most ofthe grocery superstores are using computer software to managesales and customer data. Exploring and analyzing this data canprovide new insights which can lead to proﬁt acceleration.In this paper, we will analyze the purchase behavior of acustomer using machine learning. Machine learning techniquescan be divided into supervised and unsupervised learning. Asupervised machine learning model is built based on previ-ously known purchase behavior. Once the model is built, itcan generate potentiality score for a new customer purchasepattern. A supervised model is built using labeled data. On theother hand, an unsupervised model does not have any labeleddata, rather classiﬁes customers into clusters based on similarpurchase behavior.

**Problem Identification & Objectives**

The focus of modern grocery superstore business has been

shifted to the customer-centric organization. Customers are the

most important factor for a business. Some customers can help

the business to generate more proﬁt compared to the others.

A loyalty-prone customer intends to stay with the supplier

who can provide the quality products. On the other hand, a

deal-prone customer will always look for a better offer from

a competitor. Customers can be classiﬁed as proﬁtable and

unproﬁtable [1].

Bringing a new customer to the business is cost-intensive.

Because it involves various marketing strategies without hav-

ing prior knowledge about the customer. However, a business

can easily apply speciﬁc marketing strategies based on pre-

vious customers information, when it approaches it’s existing

customers

The focus of modern grocery superstore business has been shifted to the customer-centric organization. Customers are the most important factor for a business. Some customers can help the business to generate more proﬁt compared to the others. A loyalty-prone customer intends to stay with the supplier who can provide the quality products. On the other hand, adeal-prone customer will always look for a better offer from competitor. Customers can be classiﬁed as proﬁtable insupportable [1]. Bringing a new customer to the business is cost-intensive. Because it involves various marketing strategies without having prior knowledge about the customer. However, a business can easily apply speciﬁc marketing strategies based on pre-vinous customers information, when it approaches its existing customers

# CHAPTER- 2

# literature survey

## 2.1 literature review

Studying customer behavior is age old problem which had

initially attracted by the researchers of business industries.

Every business wants to keep their customers on the long-

term basis. For last few decades, business researchers were

exploring the importance and strategies for building long-

term relations with customers [13], [14]. In an early research

by E. Gummesson [15] had discussed the importance of the

long-term relationship between business and its customer. A

close relation to customers will lower marketing cost per

customer. This does not mean that a new customer would not

be desirable, but it means the business should focus more on

how it can build strong long-term relations with its customers.

Relationship marketing concepts are introduced in 80s and few

notable studies [13], [14], [16] discussed long-term relation-

ship approaches. To build a strong relationship with your cus-

tomer it is important to understand their purchase behaviors.

Though there were many studies in 90s to understand customer

purchase pattern, it was not sufﬁcient. Approaches were way

more expensive since lots of handwritten data analysis were

involved. Due to the involvement of high cost, none of them

were feasible to apply in a practical business scenario.

In 2000, the modern technology took over the business

industries for data storing. Which brought a scope for the

researchers to explore the data in minimalistic setup. To model

a customer behavior, a range of approaches had been proposed.

Marzia et al. [17] had discussed detailed literature reviews

related to the application of predictive analytics in customer

relationship management. Xu and Walton [18] had conducted

a research on customer relationship management to understand

customer demands. They had also proposed an analytical CRM

system for customer knowledge accusation. Buckinx et al

had proposed a prediction model for the customers future

spending patterns [19]. Using a transactional database, they

analyzed customer behavior. Vanderveld et al. [20] had used

machine learning technique to describe customer lifetime value

for a real-world business. Guimei et al. [21] had explored

Alibaba sales data and proposed prediction requirements and

most important features using feature engineering and machine

learning.

As this is the ﬁrst research on this topic of purchase behavior

analysis using machine learning, all our approaches are unique.

We have used the feature engineering technique to achieve

more accurate results. Our study shows that machine learning

and feature engineering can be a legitimate tool for potential

customer prediction.

Studying customer behavior is age old problem which hadinitially attracted by the researchers of business industries.Every business wants to keep their customers on the long-term basis. For last few decades, business researchers wereexploring the importance and strategies for building long-term relations with customers [13], [14]. In an early researchby E. Gummesson [15] had discussed the importance of thelong-term relationship between business and its customer. Aclose relation to customers will lower marketing cost percustomer. This does not mean that a new customer would notbe desirable, but it means the business should focus more onhow it can build strong long-term relations with its customers.Relationship marketing concepts are introduced in 80s and fewnotable studies [13], [14], [16] discussed long-term relation-ship approaches. To build a strong relationship with your customer it is important to understand their purchase behaviors. Though there were many studies in 90s to understand customer purchase pattern, it was not sufﬁcient. Approaches were way more expensive since lots of handwritten data analysis were involved. Due to the involvement of high cost, none of them were feasible to apply in a practical business scenario. In 2000, the modern technology took over the business industries for data storing. Which brought a scope for the researchers to explore the data in minimalistic setup. To model customer behavior, a range of approaches had been proposed. Marzia et al. [17] had discussed detailed literature reviews related to the application of predictive analytics in customer relationship management. Xu and Walton [18] had conducted research on customer relationship management to understand customer demands. They had also proposed an analytical M-Systems for customer knowledge accusation. Buckinx et alhad proposed a prediction model for the customers future spending patterns [19]. Using a transactional database, they analyzed customer behavior. Vandervelde et al. [20] had used machine learning technique to describe customer lifetime value for a real-world business. Guinea et al. [21] had explored Alibaba sales data and proposed prediction requirements andmost important features using feature engineering and machine learning. As this is the ﬁrst research on this topic of purchase behavior analysis using machine learning, all our approaches are unique. We have used the feature engineering technique to achieve more accurate results. Our study shows that machine learning and feature engineering can be a legitimate tool for potential customer prediction.

# Chapter-3

# Theoretical background

## 3.1 Introduction:

## 3.2 Introduction to PYTHON

**Python Technology**

Python technology is both a programming language and a platform.

**The python Programming Language**

THE PYTHON PROGRAMMING LANGUAGE IS A HIGH-LEVEL LANGUAGE THAT CAN BE CHARACTERIZED BY ALL OF THE FOLLOWING BUZZWORDS:

* + - Simple
    - Architecture neutral
    - Object oriented
    - Portable
    - Distributed
    - High performance
    - Interpreted
    - Multithreaded
    - Robust
    - Dynamic
    - Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Python programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called Python byte codes —the platform-independent codes interpreted by the interpreter on the Python platform. The interpreter parses and runs each Python byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.

FEATURES OF MACHINE LEARNING

• It is nothing but automating the Automation.

• Getting computers to program themselves.

• Writing Software is bottleneck.

• Machine leaning models involves machines learning from data without the help of humans or any kind of human intervention.

• Machine Learning is the science of making of making the computers learn and act like humans by feeding data and information without being explicitly programmed.

• Machine Learning is totally different from traditionally programming, here data and output is given to the computer and in return it gives us the program which provides solution to the various problems. Below is the figure.

**Traditional Programming vs Machine Learning**

• Machine Learning is a combination of Algorithms, Datasets, and Programs.

• There are Many Algorithms in Machine Learning through which we will provide us the exact solution in predicting the disease of the patients.

• How Does Machine Learning Works?

• Solution to the above question is Machine learning works by taking in data, finding relationships within that data and then giving the output.

**Machine Learning Model**

• There are various applications in which machine learning is implemented such as Web search, computing biology, finance, e-commerce, space exploration, robotics, social networks, debugging and much more.

• There are 3 types of machine learning supervised, unsupervised, and reinforcement.

**BENEFITS OF PYTHON**

• Presence of Third-Party Modules

• Extensive Support Libraries

• Open Source and Community Development

• Learning Ease and Support Available

• User-friendly Data Structures

• Productivity and Speed

• Highly Extensible and Easily Readable Language.

**Python**

Python is high level language and it is also integrated version of the program. Python is an object-oriented approach and its main aim to help programmers to write the code clearly, logical code for small and large scale of project.

Pytrhon is dynamically typed and garbage collected it also support multiple programming and it is both procedure and object oriented and also functional programming. And structural programming also supported. It has many built in function it also supports filter, map and reduce function. All the machine learning algorithm and the libraries are being supported by the python programming language. Python also support list, dict, sets and other generators. Python code can be run in different platform such as anaconda, PyCharm etc.

The main goal of this programing language is as follows:

• Python is simple, object-oriented programming language.

• The language and implementation should provide support for software engineering principles such as strong type library preset for different machine learning algorithm, and all other algorithm in simple manner.

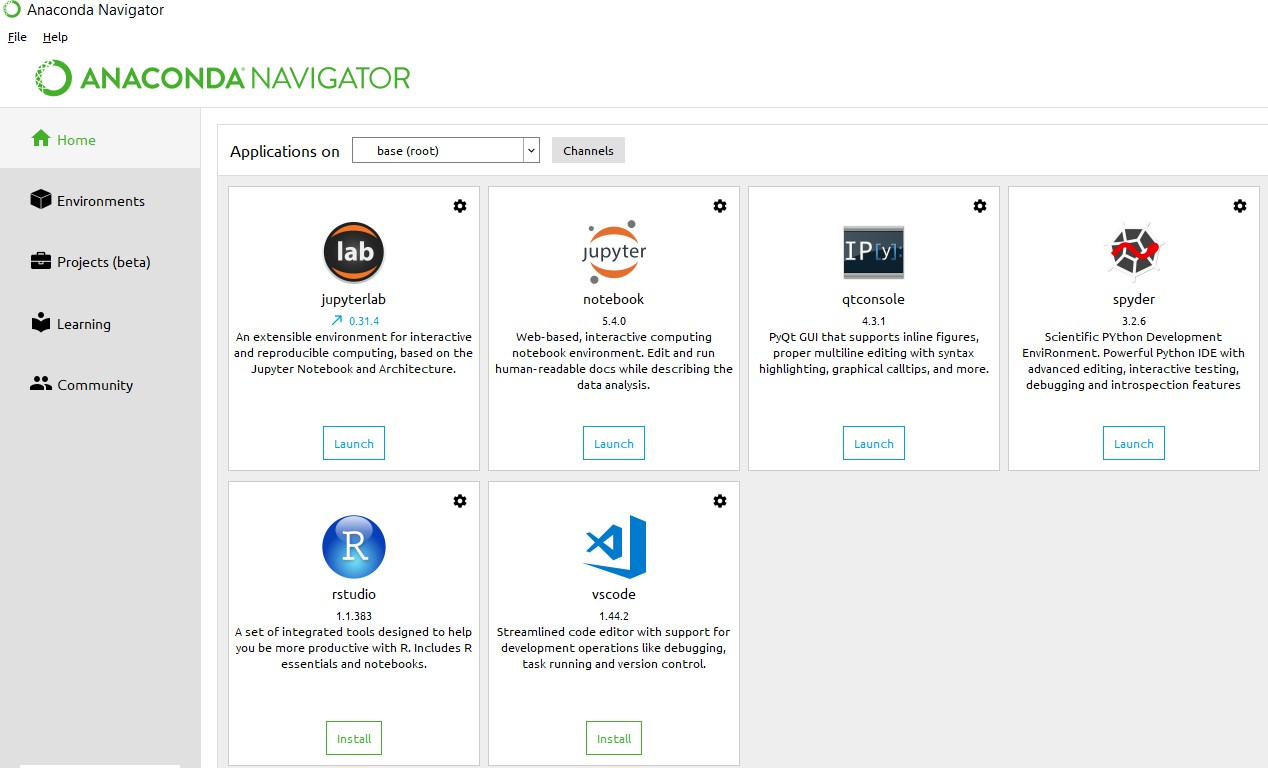
• Coding will be smooth in python and the data analysis can be easily done in python.

This is so much so to the point where we now have modules and APIs at our disposal, and you can engage in machine learning very easily without almost any knowledge at all of how it works. With the defaults from Scikit-learn, you can get 90-95% accuracy on many tasks right out of the gate. Machine learning is a lot like a car, you do not need to know much about how it works in order to get an incredible amount of utility from it.

Despite the apparent age and maturity of machine learning, I would say there's no better time than now to learn it, since you can actually use it. Machines are quite powerful, the one you are working on can probably do most of this series quickly. Data is also very plentiful lately.

**Anaconda**

Anaconda is free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine Learning applications, Large- scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. It is developed and maintained by Anaconda, Inc. The distribution incudes data-science packages suitable for Windows, Linux, and macOS. Packaged versions are required and are managed by the package management system anaconda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only conda, Python, the packages they depends on, and a small number of other packages.



**Anaconda Console**

**Jupyter notebook**

Jupiter Notebook or so called IPython Notebook is an interactive web based computational mean for starting with Jupiter Notebook documents. The term notebook itself is a huge entity to represent the integration with different entity sets. JSON is the main document form from the same for the execution which follows the brief on the schema and the input and output means. It has high integration with several language set and has various flexibilities with the choices. The extension used for the same is “.ipynb” which runs in this platform. It’s an open-source software package with interactive communication means. It has it’s open standards for the same. It’s an open community best for budding programmers . The flexibility of the same is phenomenon and splendidly done the configuration and integration of the same is simplest and easy on hold so that no prior distortion is generated and the efficiency of the same is measured through out any system of choice.

It’s the best software sets that been used across cross for designing and developing of the products and support wide help support. Not only to that, it provides scalability in the code and the deployment of the same. Various Language can be changed and the project can be undertaken on the same. The created notebook files can be shared and stored in various means for further utilization. It supports cultivated and interactive output sets. Easily crossed over for graphing, plotting and visualizing of the elements. Data Integration of the same is to it’s best. The integration of big data and it can process chunks of values in an approx. time which gives a better performance and the higher computational means. Various works on data like cleaning, cleansing, transforming modeling and visualizing can be done by the same

Machine learning is the ability that gives the computer to learn without being explicitly programmed. There are two types of machine learning:

Supervised Learning: supervised learning is the learning of the labelled data. It is the types of machine learning that maps the input and output based on the examples input-output pairs. In supervised learning each training data having pairs of input and desired outputs values. Supervised learning algorithm analyzes the training data and produces a function which can be used for mapping of new data.

Fig 2.1 Supervised Learning The output to solve the supervised learning algorithm are as:

• Determine the types of data, before doing anything else the user should understand which types of data set is to be used for training the data.

• Gathered the training data sets either in form of human experts or from measurements.

• Determine the feature of inputs from the learned data and depends on the inputs it changed into feature vector; number of features should not be large but should contains enough information to accurately predict the outputs.

• Check the learned function and the learned algorithm for example we use support vector machines or decisions tree.

• Complete the design and run the trained data sets.

• Analyzed the output and verify the data sets to get the accurate outputs.

Unsupervised Learning:

Unsupervised learning is a type of machine learning that helps in finding the previously unknown patterns in the data set without any known labels. It is known as self- organization and allows modelling probability densities of given inputs.

Fig 2.2 unsupervised Learning Some of the algorithm used in unsupervised learning are:

• Clustering

• Anomaly detection

• Neural networks

• Approach for learning latent variable models

• Non labelled data

Semi Supervised Machine Learning algorithm: It’s like the middle man which have some labeled data and some unlabeled which can be prosed by the both the structured and unsupervised learning.

The algorithms have been compared based upon the parameters: Size of the dataset and Number of technical indicators used. Accuracy and F-measure values have been computed for each algorithm. Long term model has been used to compute the accuracy and F-measure.

Reinforcement Learning: This type of learning is used to reinforce or strengthen the network based on critic information. That is, a network being trained under reinforcement learning, receives some feedback from the environment. However, the feedback is evaluative and not instructive as in the case of supervised learning. Based on this feedback, the network performs the adjustments of the weights to obtain better critic information in future.

This learning process is similar to supervised learning but we might have very less information. The following figure gives the block diagram of reinforcement learning:

**import numpy as np**

* NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.
* At the core of the NumPy package, is the ndarray object. This encapsulates n-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance. There are several important differences between NumPy arrays and the standard Python sequences:
  + NumPy arrays have a fixed size at creation, unlike Python lists (which can grow dynamically). Changing the size of an ndarray will create a new array and delete the original.
  + The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including NumPy) objects, thereby allowing for arrays of different sized elements.
  + NumPy arrays facilitate advanced mathematical and other types of operations on large numbers of data. Typically, such operations are executed more efficiently and with less code than is possible using Python’s built-in sequences.
  + A growing plethora of scientific and mathematical Python-based packages are using NumPy arrays; though these typically support Python-sequence input, they convert such input to NumPy arrays prior to processing, and they often output NumPy arrays. In other words, in order to efficiently use much (perhaps even most) of today’s scientific/mathematical Python-based software, just knowing how to use Python’s built-in sequence types is insufficient - one also needs to know how to use NumPy arrays.

**import time**

This module provides various time-related functions. For related functionality, see also the datetime and calendar modules.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

An explanation of some terminology and conventions is in order.

The epoch is the point where the time starts, and is platform dependent. For Unix, the epoch is January 1, 1970, 00:00:00 (UTC). To find out what the epoch is on a given platform, look at time.gmtime(0).

The term seconds since the epoch refers to the total number of elapsed seconds since the epoch, typically excluding leap seconds. Leap seconds are excluded from this total on all POSIX-compliant platforms.

The functions in this module may not handle dates and times before the epoch or far in the future. The cut-off point in the future is determined by the C library; for 32-bit systems, it is typically in 2038.

Function strptime() can parse 2-digit years when given %y format code. When 2-digit years are parsed, they are converted according to the POSIX and ISO C standards: values 69–99 are mapped to 1969–1999, and values 0–68 are mapped to 2000–2068.

UTC is Coordinated Universal Time (formerly known as Greenwich Mean Time, or GMT). The acronym UTC is not a mistake but a compromise between English and French.

DST is Daylight Saving Time, an adjustment of the timezone by (usually) one hour during part of the year. DST rules are magic (determined by local law) and can change from year to year. The C library has a table containing the local rules (often it is read from a system file for flexibility) and is the only source of True Wisdom in this respect.

The precision of the various real-time functions may be less than suggested by the units in which their value or argument is expressed. E.g. on most Unix systems, the clock “ticks” only 50 or 100 times a second.

On the other hand, the precision of time() and sleep() is better than their Unix equivalents: times are expressed as floating point numbers, time() returns the most accurate time available (using Unix gettimeofday() where available), and sleep() will accept a time with a nonzero fraction (Unix select() is used to implement this, where available).

The time value as returned by gmtime(), localtime(), and strptime(), and accepted by asctime(), mktime() and strftime(), is a sequence of 9 integers. The return values of gmtime(), localtime(), and strptime() also offer attribute names for individual fields.

See struct\_time for a description of these objects.

Changed in version 3.3: The struct\_time type was extended to provide the tm\_gmtoff and tm\_zone attributes when platform supports corresponding struct tm members.

Changed in version 3.6: The struct\_time attributes tm\_gmtoff and tm\_zone are now available on all platforms.

**import os**

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Notes on the availability of these functions:

The design of all built-in operating system dependent modules of Python is such that as long as the same functionality is available, it uses the same interface; for example, the function os.stat(path) returns stat information about path in the same format (which happens to have originated with the POSIX interface).

Extensions peculiar to a particular operating system are also available through the os module, but using them is of course a threat to portability.

All functions accepting path or file names accept both bytes and string objects, and result in an object of the same type, if a path or file name is returned.

On VxWorks, os.popen, os.fork, os.execv and os.spawn\*p\* are not supported.

# 

# Chapter-4

# System analysis

## 4.1 EXISTING SYSTEM:

In [6] an essay dataset is used for personality trait identification. Multi-label Naive Bayes classifier has been used to predict results. [7] uses keystroke dynamics dataset by using an application which is web-based and that runs on mobile. In this work various features are extracted on the basis of gender, age, etc. Machine learning algorithms such as Naive Bayes, kNN, Random Forest have been a over the dataset. [8] focuses on using Facebook graph API to collect data for predicting the personality of user based on Facebook profile. Big Five Inventory [8] is used for this work. Results of this work have been calculated by taking both supervised and unsupervised methods into consideration.

### 4.1.1 DISADVANTAGES OF EXISTING SYSTEM:

* Expeced accuracy levelof 70% is not attained
* Method of gray prediction is not widely accepted
* Dataset used in this study is not easy to understand and publicly not available

## 4.2 PROPOSED SYSTEM:

* To lift the revenue boundary and stay ahead of the competitors it is important to understand customer’s purchase behavior. Different business industries proposed different policies to explore the potentiality of a customer based on statistical analysis. In this paper, we rather propose a machine learning approach to identify potential customers for a retail superstore. The paper proposed an engineered approach to classify potential customer, based on previously recorded purchase behavior. Using this classiﬁcation as ground truth, we then apply machine learn-ing algorithms to ﬁnd a pattern to predict potential customers with an accuracy of 99.4%.

### 4.2.1 ADVANTAGES OF PROPOSED SYSTEM:

Multilevel predictive model proposed in this study is unique approach which was not raised before.

Grocery superstore business has been accumulating huge

data from customers. Due to the digital revolution, most of

the grocery superstores are using computer software to manage

sales and customer data. Exploring and analyzing this data can

provide new insights which can lead to proﬁt acceleration.

In this paper, we will analyze the purchase behavior of a

customer using machine learning. Machine learning techniques

can be divided into supervised and unsupervised learning. A

supervised machine learning model is built based on previ-

ously known purchase behavior

Grocery superstore business has been accumulating huge data from customers. Due to the digital revolution, most often grocery superstores are using computer software to manage sales and customer data. Exploring and analyzing this data can provide new insights which can lead to proﬁt acceleration. In this paper, we will analyze the purchase behavior of a customer using machine learning. Machine learning techniques can be divided into supervised and unsupervised learning. A supervised machine learning model is built based on previously known purchase behavior

# CHAPTER- 5

# SYSTEM design

## 5.1 introduction

System Design Introduction:

The System Design Document describes the system requirements, operating environment, system and subsystem architecture, files and database design, input formats, output layouts, human-machine interfaces, detailed design, processing logic, and external interfaces.

## 5.2 modules

### 5.2.1 Data COLLECTION:

Our dataset consists of 9259 customers sales data. Over the three months, 194439 invoice entries are given. On average a customer made 7 purchase each month. Each invoice contains minimum one and maximum 83 types of items. Items are grouped into 19 categories. We consider six glossary categories for this research. We extract all sales information related tour research from the dataset and applied machine learning approach to create the classiﬁcation model. Grocery superstore named” Tara din Super Shop” provided the sales data for this research. Tara din is located in the Sylhet district of Bangladesh. trading has been doing grocery business since 2001. They have been using customized sales software for billing. Since the database of their software was designed to meet their custom requirement, we had to apply several techniques to tailor data and extract feature as per our research requirements..

### 5.2.2 Feature Engineering:

To identify signiﬁcant features ﬁrst we have to understand their database structure. From database engineering perspective, a single invoice data of a customer was saved into multiple tables. A database table named product sale master(Figure 2) stores invoice number, item barcode, the price often item, sold quantity and measurement unit of that item. Each item of an invoice was stored into this prod-uct sale master data table. For example, an invoice with evetimes will make ﬁve rows entries in product sale master table. This table did not provide any information on ”who bought the item?” or ”what was the item?”. For our research, it was important to ﬁnd the relation between invoice, customer, measurement unit and quantity of each sold items. prod-uct sale master table only gave us data on invoice, quantity and measurement unit, but it didn’t tell us item details and customer ID. So, we had kept looking and found another table named billing details (Figure 3) contained invoice number, date of sale, time of sale, customer ID, amount of sale, vat and bill code. From this billing details, we had found the relation between an invoice number and customer ID but still, the product details were not known to us. We had to look into other .

### 5.2.3 preprocessing:

Our tailored dataset of this research contained items with

various unit scale. Using multiple unit measurement could

bias the result. As the ﬁrst step of our data preprocessing,

we had to bring all our data into a common unit scale. We

had ”Kilo Grams”, ”Grams” and ”Pieces” as measurement

units in our dataset. We had converted all weight measurement

from Kilo Grams to Grams. Generalizing measurement unit

”Pieces” with ”Grams” was challenging. We were informed

that items sold in ”Pieces” were bought in ”Kilo Grams” from

the wholesale market. From this knowledge, we had mapped

items wholesale measurement unit with the retail measurement

unit. Once all data had been formed in a single measurement

unit, we had standardized and scaled before using in machine

learning to ensure that the ﬁelds with larger numeric values

that did not bias the results.

Our tailored dataset of this research contained items with various unit scale. Using multiple unit measurement could bias the result. As the ﬁrst step of our data preprocessing, we had to bring all our data into a common unit scale. Wahad ”Kilo Grams”, ”Grams” and ”Pieces” as measurement units in our dataset. We had converted all weight measurement from Kilo Grams to Grams. Generalizing measurement unit “Pieces” with” Grams” was challenging. We were informed that items sold in” Pieces” were bought in ”Kilo Grams” from the wholesale market. From this knowledge, we had mapped items wholesale measurement unit with the retail measurement unit. Once all data had been formed in a single measurement unit, we had standardized and scaled before using in machine learning to ensure that the ﬁelds with larger numeric values that did not bias the results..

**Customer classification:**

Customers are classiﬁed into two classes. (a) High Potential Customer & (b) Low Potential Customer. For the purpose of machine understanding we present those two classes using 1and 0. In the data set High Potential Customer is represented using 1and Low Potential Customer is represented using 0.Customer is classiﬁed using Proﬁt Threshold λ.Proﬁt Threshold is calculated from the features of customer purchase historical data along with their respectiveproﬁt percentage. We have customer purchase data for three-month. Also, we have the proﬁt percentage on each item sold along these three months. All item was categorized intoic1, ic2, ic3...icn. Where nis the number of total categories. Each of those categories contribute to the total proﬁt. If theorist percentage on each category is p1, p2, p3...pn, then taproot (P)

**Algorithms**

#### Random Forest

Random forest is a tree-based algorithm which involves building several trees (decision trees), then combining their output to improve generalization ability of the model. The method of combining trees is known as an ensemble method. Ensembling is nothing but a combination of weak learners (individual trees) to produce a strong learner.

Definition: A random forest is a classifier consisting of a collection of tree structured classifiers *h*(*x,* Θ*k*)*, k* = 1*, ...* where the Θ*k* are independent identically distributed (*i.i.d*) random vectors and each tree casts a unit vote for the most popular class at input [[4](#_bookmark61)].

Random Forest Algorithm: The following are the basic steps involved in performing the random forest algorithm:

* + - * Pick N random records from the dataset.
      * Build a decision tree based on these N records.
      * Choose the number of trees you want in your algorithm and repeat steps (i) and (ii).
      * In case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

Figure [2.1](#_bookmark0) shows different trees labelling the class differently. What ensemble does is take the mode (maximum occurring class) of the output produced by n different trees to create a better model. To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.

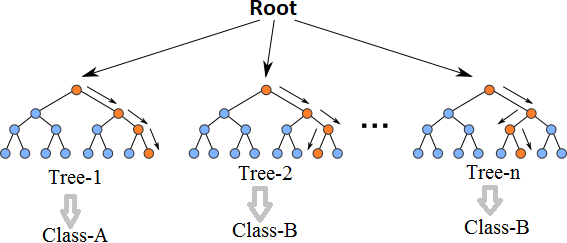


Fig. 2.1 Multiple decision trees [[12](#_bookmark69)]

Even though decision trees are pretty intuitive and easier to understand, they can be very noisy. Few changes in the data can lead to different splits and completely different models. The instability of the tree makes it unrealistic as a prediction model by itself. A single decision tree is insufficient and generally overfits the data, that is it can capture the structure of the in-sample data very well, but it tends to work poorly out-of-sample. In the context of statistics, decisions trees have low bias (as it can fit the data well) but high variances (the predictions are noisy).

Understanding the working principle of decision trees is imperative in the understanding of Random Forest Algorithm. The most popular algorithm for decision trees is ID3 algorithm. It finds the best attributes/features that best classifies the target attribute. One of the most commonly used way to figure out the best attribute is by calculating Information Gain which is, in turn, calculated using another property called Entropy.

The calculation of entropy of a system is done as follows:

*c*

*Entropy*(*S*) = ∑ *pilog*2 *pi* (2.1)

−

*i*=1

Here, c is the total number of classes or attributes and *pi* is number of examples belonging to the *ith* class. Information gain is simply the expected reduction in entropy caused by partitioning all our examples according to a given attribute. Mathematically, it is defined as:

*Gain*(*S, A*) ≡ *Entropy*(*S*) − ∑ |*Sv*| *Entropy*(*Sv*) (2.2)

*v*∈*Values*(*A*) |*S*|

S refers to the entire set of examples that we have. A is the attribute we want to partition or split. |S| is the number of examples and |*Sv*| is the number of examples for the current value of attribute A. The attribute with the highest information gain sits at the root node, and the tree is first split based on that attribute.

#### XGBoost

XGBoost is another ensemble learning method. As it is almost never sufficient to reply upon the results of just one model, it combines the predictive powers of multiple learners to reach a conclusion. The base learners are weak learners in which the bias is high, and the predictive power is just slightly better than random guessing. But each of these weak learners add some vital information for prediction, resulting in a strong learner by effectively combining these weak learners. The final strong learner brings down both the bias and the variance.

The tree ensemble model consists of a set of classification and regression trees (CART). Figure [2.2](#_bookmark1) shows a simple example of a CART that classifies whether someone will like an

app or not. The original figure from [[5](#_bookmark62)] had been modified to paint a better picture of our dataset.

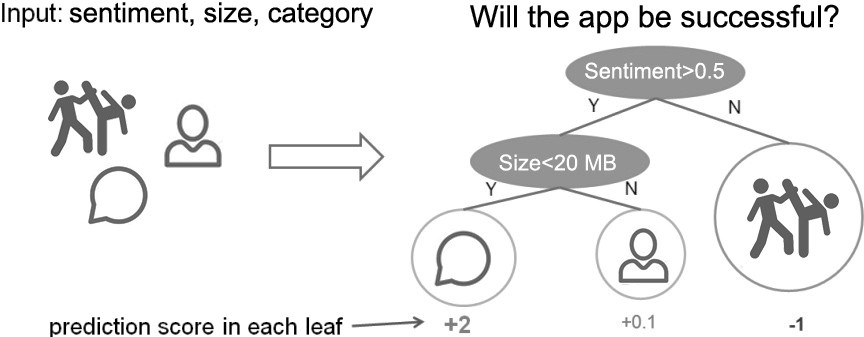


Fig. 2.2 CART Model Representation

Suppose, the many app categories available are classified into different leaves and as- signed a score on the corresponding leaf. Unlike decision trees, in which the leaf only contains decision values, in CART, a real score is associated with each of the leaves, which gives a better interpretation.

The task of training the model involves finding the best parameters *θ* that best fit the

training data *xi* and labels *yi*. This is done via the objective function which measures how well the model fits the training data. Objective functions are composed of two parts: training loss and regularization term which can be denoted by:

*ob j*(*θ* ) = *L*(*θ* ) + Ω(*θ* ) (2.3)

where *L* is the training loss function, and Ω is the regularization term. The regularization term controls the complexity of the model, helping to avoid overfitting.

While trees are built in a parallel manner in bagging, boosting builds trees sequentially such that each subsequent tree aims to reduce the errors of the previous tree. Figure [2.3](#_bookmark2) perfectly illustrates the concept. Due to each tree learning from its predecessors and updating the residual errors (difference between an observed y-value and the corresponding predicted y-value), the tree that grows next in the sequence will always learn from an updated version of the residuals. This is known as an additive strategy where what has already been learned is fixed, and a new tree is added one at a time.

The boosting process in its absolute basic can be broken down into the following steps [[22](#_bookmark79)]:

* + - * Fit a model to the data: *F*1(*x*) = *y*
      * Fit a model to the residuals: *h*1(*x*) = *y* − *F*1(*x*)

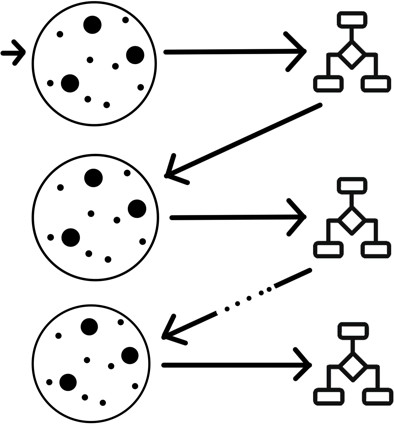


Fig. 2.3 Sequential Tree Structure

* Create a new model: *F*2(*x*) = *F*1(*x*) + *h*1(*x*)

By creating more models that correct the errors of the previous models, this can be generalized to:

*F*(*x*) = *F*1(*X* ) → *F*2(*x*) = *F*1(*x*) + *h*1(*x*)*. . . ..* → *FM*(*x*) = *FM*−1(*x*) + *hM*−1(*x*)*.* (2.4)

At each step, the residual would also need to be calculated: *hm*(*x*) = *y* − *Fm*(*x*) where *hm*(*x*) can be any model, but in our case, it is a tree-based learner. With this in mind, suppose that instead of training *h*0 on the residuals of *F*0, we train *h*0 on the gradient of the loss function, *L*(*y, F*0(*x*)) with respect to the prediction values produced by *Fm*(*x*). With samples in *hm* grouped into leaves, an average gradient can be calculated and then scaled by some factor, *γ*, such that *Fm* + *γhm* minimizes the loss function for the samples in each leaf. In practice, a different factor is chosen for each leaf. For iteration m = 1 to M:

* Calculate the gradient of L at the point *sm*−1
* “Step” in the direction of greatest descent (the negative gradient) with step size *γ*. That is, *sm* = *sm*−1 − *γL*(*sm*−1). If *γ* is small and *M* is sufficiently large, *sM* will be the location of *L* ‘s minimum value.

Most of these are true for all previous gradient boosting algorithms that came before XGBoost, but what really separates it from the others is [[22](#_bookmark79)]:

* Regularization: XGBoost can penalize complex models through both L1 and L2 regularization which helps prevent over-fitting.
  + Handling sparse data: Missing values or data processing steps like one-hot encoding can make data sparse. XGBoost incorporates a sparsity-aware split finding algorithm that can handle different types of sparsity patterns in the data.
  + Weighted quantile sketch: Most existing tree based algorithms can find the split points when the data points are of equal weights (using quantile sketch algorithm). However, they can not handle weighted data. XGBoost has a distributed weighted quantile sketch algorithm that can effectively handle weighted data.
  + Block structure for parallel learning: For faster computing, XGBoost can utilize multiple cores on the CPU. Unlike other algorithms, this enables the data layout to be reused by subsequent iterations, instead of computing it again.
  + Cache awareness: In XGBoost, non-continuous memory access is required to get the gradient statistics by row index. Hence, XGBoost has been designed to make optimal use of hardware.
  + Out-of-core computing: This feature optimizes the available disk space and maximizes its usage when handling huge datasets that do not fit into memory

#### K-Nearest Neighbours

The k-nearest neighbor (k-NN) algorithm is one of the data mining techniques considered to be among the top 10 techniques for data mining [[16](#_bookmark73)]. The k-NN method uses the well-known principle of Cicero pares cum paribus facillime congregantur (birds of a feather flock together or literally equals with equals easily associate). It tries to classify an unknown sample based on the known classification of its neighbors. Let us suppose that a set of samples with known classification is available, the so-called training set. Intuitively, each sample should be classified similarly to its surrounding samples. Therefore, if the classification of a sample is unknown, then it could be predicted by considering the classification of its nearest neighbor samples. Given an unknown sample and a training set, all the distances between the unknown sample and all the samples in the training set can be computed. The distance with the smallest value corresponds to the sample in the training set closest to the unknown sample. Therefore, the unknown sample may be classified based on the classification of this nearest neighbor.

Figure [2.4](#_bookmark3) shows the k-NN decision rule for k = 1, k = 2 and k = 3 for a set of samples divided into 2 classes X and Y. In Figure [2.4](#_bookmark3)(a), an unknown sample is classified by using only one nearest neighbour and which would label the ? as X; in Figure [2.4](#_bookmark3)(b) two known neighbours are used and since its a tie-breaker, the unknown test instance may be labeled as X or Y. In the last case in Figure [2.4](#_bookmark3)(c), the parameter k is set to 3, so that the closest three

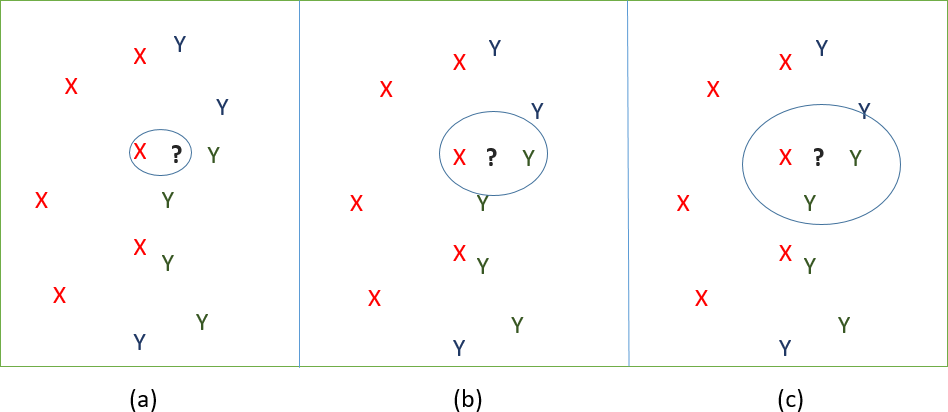
neighbours are considered for classifying the unknown one. Two of them belong to the same class, whereas only one belongs to the other class. Thus the unknown sample ? is labeled as Y.

Fig. 2.4 (a) 1-nearest neighbour; (b) 2-nearest neighbours; (c) 3-nearest neighbours The following Figure [2.5](#_bookmark4) shows an algorithm in understandable terms.

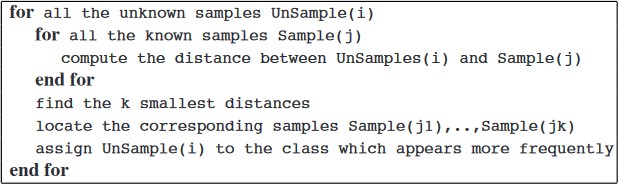


Fig. 2.5 K-NN Algorithm [[16](#_bookmark73)]

#### Support Vector Machine

A support vector machine (SVM) is a type of supervised machine learning classification algorithm which outputs an optimal hyperplane that categorizes new examples given labeled training data [[15](#_bookmark72)]. SVMs were introduced initially in 1960s and were later refined in 1990s. However, it is only now that they are becoming very popular, owing to their ability to achieve outstanding results.

Simple SVM: In case of linearly separable data in two dimensions, as shown in Figure [2.6](#_bookmark5), a typical machine learning algorithm tries to find a boundary that divides the data in such a way that the misclassification error can be minimized. If you closely look at Figure [2.6](#_bookmark5), there can be several boundaries that correctly divide the data points. The two dashed lines as well as one solid line classify the data correctly.

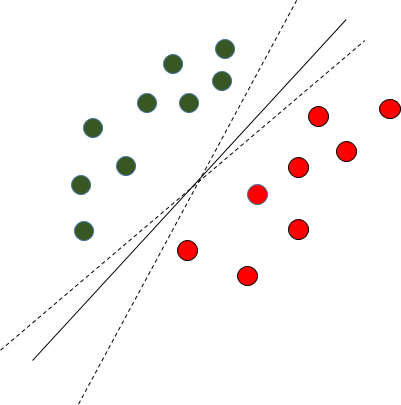


Fig. 2.6 Multiple Decision Boundaries

SVM differs from the other classification algorithms in the way that it chooses the decision boundary that maximizes the distance from the nearest data points of all the classes. An SVM doesn’t merely find a decision boundary; it finds the most optimal decision boundary. The most optimal decision boundary is the one which has maximum margin from the nearest points of all the classes. The nearest points from the decision boundary that maximize the distance between the decision boundary and the points are called support vectors as seen in Figure¬[2.7](#_bookmark6). The decision boundary in case of support vector machines is called the maximum margin classifier, or the maximum margin hyper plane.

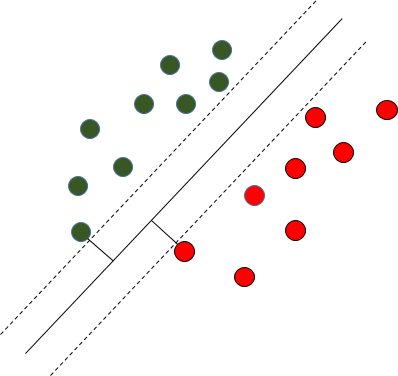


Fig. 2.7 Decision Boundary with Support Vectors

Kernel SVM: In the previous two figures Figure [2.6](#_bookmark5) and Figure [2.7](#_bookmark6) it was shown how the simple SVM algorithm can be used to find decision boundary for linearly separable data. However, in the case of non-linearly separable data, such as the one shown in Figure [2.8](#_bookmark7), a straight line cannot be used as a decision boundary.

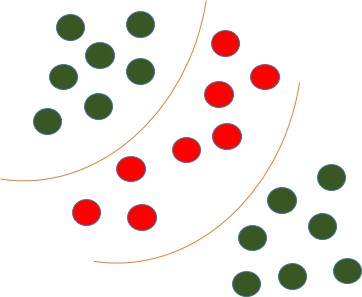


Fig. 2.8 Non-linearly Separable Data

In case of non-linearly separable data, the simple SVM algorithm cannot be used. Rather, a modified version of SVM, called Kernel SVM, is used. Basically, the kernel SVM projects the non-linearly separable data lower dimensions to linearly separable data in higher dimen- sions in such a way that data points belonging to different classes are allocated to different dimensions.

## 5.3 system architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system. Organized in a way that supports reasoning about the structures and behaviors of the system.

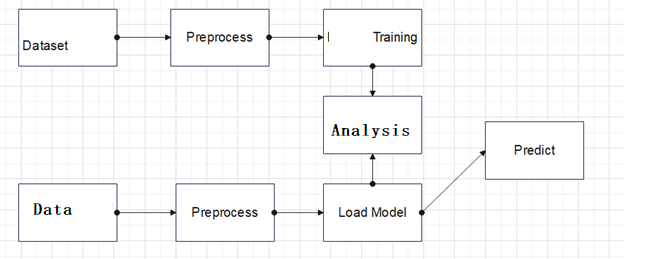
****

Figure 5. 1 System Architecture

3-Tier Architecture:

The three-tier software architecture (a three-layer architecture) emerged in the 1990s to overcome the limitations of the two-tier architecture. The third tier (middle tier server) is between the user interface (client) and the data management (server) components. This middle tier provides process management where business logic and rules are executed and can accommodate hundreds of users (as compared to only 100 users with the two tier architecture) by providing functions such as queuing, application execution, and database staging.

The three tier architecture is used when an effective distributed client/server design is needed that provides (when compared to the two tier) increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user. These characteristics have made three layer architectures a popular choice for Internet applications and net-centric information systems.

**Advantages of Three-Tier:**

* Separates functionality from presentation.
* Clear separation – better understanding.
* Changes limited to well define components.
* Can be running on WWW.
* Effective network performance.

## 5.4 UML DAIGRAMS

Global Use Case Diagrams:

Identification of actors:

Actor: Actor represents the role a user plays with respect to the system. An actor interacts with, but has no control over the use cases.

Graphical representation:



<<Actor name>>

An actor is someone or something that:

Interacts with or uses the system.

* Provides input to and receives information from the system.
* Is external to the system and has no control over the use cases.

Actors are discovered by examining:

* Who directly uses the system?
* Who is responsible for maintaining the system?
* External hardware used by the system.
* Other systems that need to interact with the system.

Questions to identify actors:

* + Who is using the system? Or, who is affected by the system? Or, which groups need help from the system to perform a task?
  + Who affects the system? Or, which user groups are needed by the system to perform its functions? These functions can be both main functions and secondary functions such as administration.
  + Which external hardware or systems (if any) use the system to perform tasks?
  + What problems does this application solve (that is, for whom)?
  + And, finally, how do users use the system (use case)? What are they doing with the system?

The actors identified in this system are:

1. System Administrator
2. Customer
3. Customer Care

Identification of use cases:

Use case: A use case can be described as a specific way of using the system from a user’s (actor’s) perspective.

Graphical representation:



A more detailed description might characterize a use case as:

* Pattern of behavior the system exhibits
* A sequence of related transactions performed by an actor and the system
* Delivering something of value to the actor

Use cases provide a means to:

* capture system requirements
* communicate with the end users and domain experts
* test the system

Use cases are best discovered by examining the actors and defining what the actor will be able to do with the system.

Guide lines for identifying use cases:

* For each actor, find the tasks and functions that the actor should be able to perform or that the system needs the actor to perform. The use case should represent a course of events that leads to clear goal
* Name the use cases.
* Describe the use cases briefly by applying terms with which the user is familiar.

This makes the description less ambiguous

Questions to identify use cases:

* What are the tasks of each actor?
* Will any actor create, store, change, remove or read information in the system?
* What use case will store, change, remove or read this information?
* Will any actor need to inform the system about sudden external changes?
* Does any actor need to inform about certain occurrences in the system?
* What usecases will support and maintains the system?

**1.2 Flow of Events**

A flow of events is a sequence of transactions (or events) performed by the system. They typically contain very detailed information, written in terms of what the system should do, not how the system accomplishes the task. Flow of events are created as separate files or documents in your favorite text editor and then attached or linked to a use case using the Files tab of a model element.

A flow of events should include:

* When and how the use case starts and ends
* Use case/actor interactions
* Data needed by the use case
* Normal sequence of events for the use case
* Alternate or exceptional flows

### 5.4.1 Construction of Use case diagrams:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

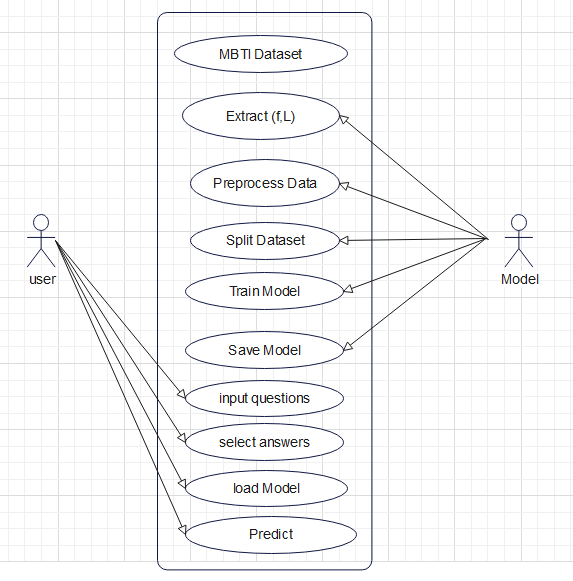
****

Figure 5. 2 Use Case Diagram

### 5.4.2 SEQUENCE DIAGRAMS:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

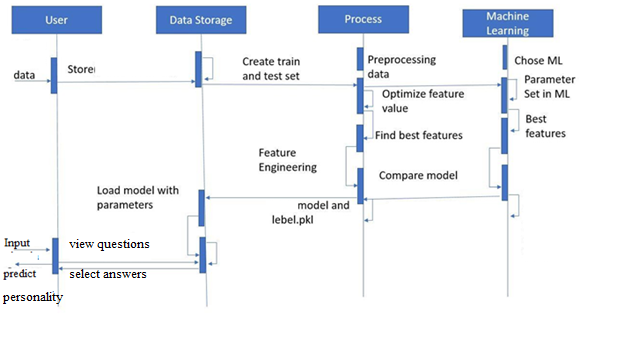
****

Figure 5. 3 Sequence diagram

### 5.4.3 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

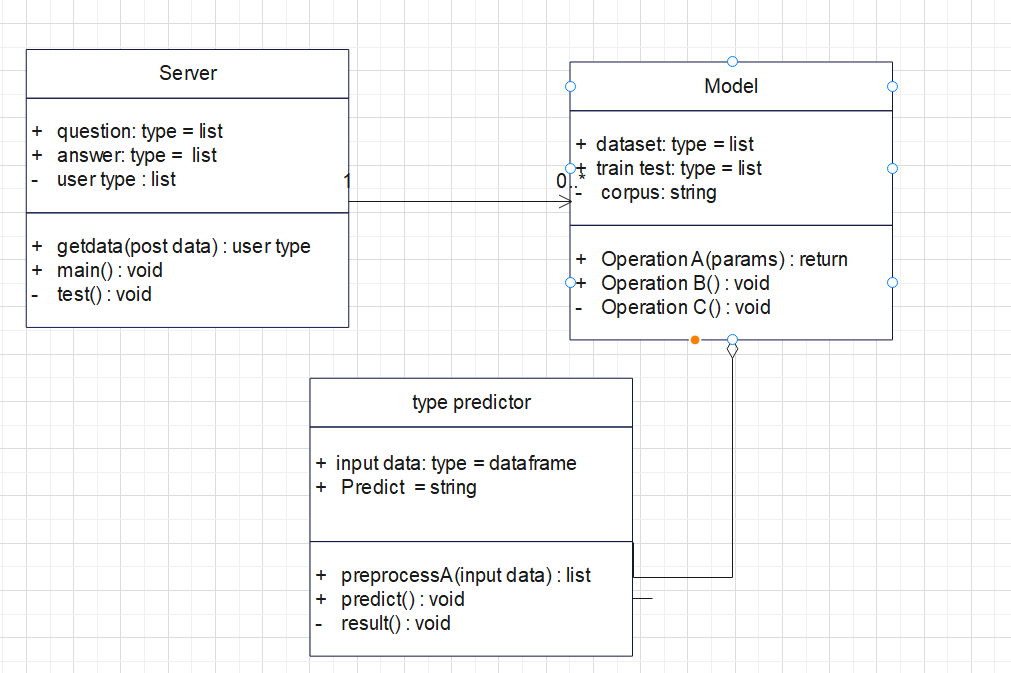
****

Figure 5. 4 Class Diagram

### 5.4.4 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

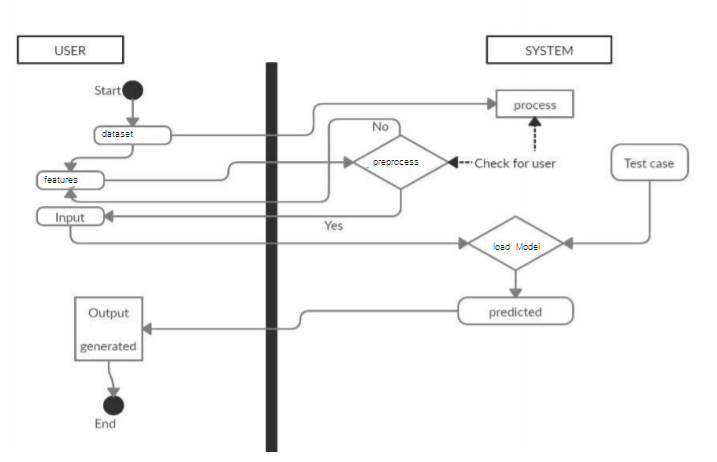
****

Figure 5. 5 Activity Diagram

# 

# CHAPTER-6

# system requirements

## 6.1 SYSTEM REQUIREMENTS

### 6.1.1 HARDWARE REQUIREMENTS:

* System : Intel(R) Core(TM) i3-7020U CPU @ 2.30GHz
* Hard Disk : 1 TB.
* Input Devices : Keyboard, Mouse
* Ram : 4 GB.

### 6.1.2 SOFTWARE REQUIREMENTS:

* Operating system : Windows XP/7/10.
* Coding Language : Python
* Tool : Anaconda
* Interface : OPENCV

# Chapter-7

# System implementation

To conduct studies and analyses of an operational and technological nature, and To promote the exchange and development of methods and tools for operational analysis as applied to defense problems.

## 7.1 input and output designs

### 7.1.1 Logical design

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modeling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems design are included. Logical design includes ER Diagrams i.e. Entity Relationship Diagrams

### 7.1.2 Physical design

The physical design relates to the actual input and output processes of the system. This is laid down in terms of how data is input into a system, how it is verified / authenticated, how it is processed, and how it is displayed as output. In Physical design, following requirements about the system are decided.

1. Input requirement,
2. Output requirements,
3. Storage requirements,
4. Processing Requirements,
5. System control and backup or recovery.

Put another way, the physical portion of systems design can generally be broken down into three sub-tasks:

1. User Interface Design
2. Data Design
3. Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the systems design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

Physical design, in this context, does not refer to the tangible physical design of an information system. To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

## 7.2 INPUT & OUTPUT REPRESENTATION

### 7.2.1 Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

### 7.2.2 Objectives

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

### Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

* 1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
  2. Select methods for presenting information.
  3. Create document, report, or other formats that contain information produced by the system.

**Code**

import webbrowser

import numpy as np

import torch

import torch.nn.functional as F

import torch.optim as optim

import torch.utils.data as data\_utils

import torchvision

from PIL import Image, ImageFile

from torch import nn

from torch import optim as optim

from torch.autograd import Variable

from torch.optim import lr\_scheduler

from torch.utils.data import DataLoader, Dataset

from torch.utils.data.sampler import SubsetRandomSampler

from torchvision import datasets, models, transforms

import cv2

train\_on\_gpu = torch.cuda.is\_available()

if not train\_on\_gpu:

**print**('CUDA is not available.  Training on CPU ...')

else:

**print**('CUDA is available!  Training on GPU ...')

ImageFile.LOAD\_TRUNCATED\_IMAGES = True

*#!pip install --upgrade wandb*

test\_transforms = transforms.Compose([transforms.Resize(255),

*#  transforms.CenterCrop(224),*

                                      transforms.ToTensor(),

                                      ])

model = models.densenet161()

model.classifier = nn.Sequential(nn.Linear(2208, 1000),

                                 nn.ReLU(),

                                 nn.Dropout(0.2),

                                 nn.Linear(1000, 2),

                                 nn.LogSoftmax(dim=1))

criterion = nn.NLLLoss()

*# Only train the classifier parameters, feature parameters are frozen*

optimizer = optim.Adam(model.parameters(), lr=0.001)

scheduler = lr\_scheduler.StepLR(optimizer, step\_size=7, gamma=0.1)

model = model.cuda()

model.load\_state\_dict(torch.load('tensorboardexp.pt'))

classes = ["accident", "noaccident"]

*# model.load\_state\_dict(torch.load('tensorboardexp.pt'))*

count = 0

counts = 1

videopath = '12.mp4'

vid = cv2.VideoCapture(videopath)

ret = True

while ret:

    if ret == True:

        ret, frame = vid.read()

        try:

            img = Image.fromarray(frame)

        except ValueError:

            break

        except AttributeError:

            break

        img = test\_transforms(img)

        img = img.unsqueeze(dim=0)

        img = img.cuda()

        model.eval()

        with torch.no\_grad():

            output = model(img)

            \_, predicted = torch.max(output, 1)

            index = int(predicted.item())

            if index == 0:

                cv2.imwrite(r"C:\xampp\htdocs\img\frame%d.png" % count, frame)

                count += 1

                if counts == 1:

                    webbrowser.open('127.0.0.1', new=2)

                    counts += 1

            labels = 'status: ' + classes[index]

        cv2.putText(frame, labels, (10, 100),

                    cv2.FONT\_HERSHEY\_DUPLEX, 2, (0, 0, 255), 5, cv2.LINE\_AA)

        cv2.imshow('Frame', frame)

        if cv2.waitKey(1) & 0xFF == **ord**('q'):

            break

vid.release()

cv2.destroyAllWindows()

import numpy as np

import torch

import torch.nn.functional as F

import torch.optim as optim

import torch.utils.data as data\_utils

import torchvision

from PIL import Image, ImageFile

from torch import nn

from torch import optim as optim

from torch.autograd import Variable

from torch.optim import lr\_scheduler

from torch.utils.data import DataLoader, Dataset

from torch.utils.data.sampler import SubsetRandomSampler

from torchvision import datasets, models, transforms

from torchsummary import summary

train\_on\_gpu = torch.cuda.is\_available()

if not train\_on\_gpu:

**print**('CUDA is not available.  Training on CPU ...')

else:

**print**('CUDA is available!  Training on GPU ...')

ImageFile.LOAD\_TRUNCATED\_IMAGES = True

data\_dir = 'accidents'

valid\_size = 0.2

train\_transforms = transforms.Compose([transforms.RandomRotation(30),

                                       transforms.RandomResizedCrop(224),

                                       transforms.RandomVerticalFlip(),

                                       transforms.RandomHorizontalFlip(),

                                       transforms.ToTensor(),

                                       transforms.Normalize([0.485, 0.456, 0.406],

                                                            [0.229, 0.224, 0.225])])

test\_transforms = transforms.Compose([transforms.Resize(255),

*#  transforms.CenterCrop(224),*

                                      transforms.ToTensor(),

                                      ])

valid\_transforms = transforms.Compose([transforms.RandomRotation(30),

                                       transforms.RandomResizedCrop(224),

                                       transforms.RandomVerticalFlip(),

                                       transforms.RandomHorizontalFlip(),

                                       transforms.ToTensor(),

                                       transforms.Normalize([0.485, 0.456, 0.406],

                                                            [0.229, 0.224, 0.225])])

*# define samplers for obtaining training and validation batches*

*# Pass transforms in here, then run the next cell to see how the transforms look*

train\_data = datasets.ImageFolder(data\_dir, transform=train\_transforms)

**print**(**len**(train\_data))

train\_data, test\_data, valid\_data = torch.utils.data.random\_split(train\_data, [

                                                                  27298, 5850, 5850])

trainloader = torch.utils.data.DataLoader(

    train\_data, batch\_size=8, num\_workers=1, pin\_memory=True)

testloader = torch.utils.data.DataLoader(

    test\_data, batch\_size=8, num\_workers=1, pin\_memory=True)

validloader = torch.utils.data.DataLoader(

    valid\_data, batch\_size=8, num\_workers=1, pin\_memory=True)

n\_classes = 2

def **get\_num\_correct**(preds, labels):

    return preds.argmax(dim=1).eq(labels).sum().item()

model = models.densenet161()

model.classifier = nn.Sequential(nn.Linear(2208, 1000),

                                 nn.ReLU(),

                                 nn.Dropout(0.2),

                                 nn.Linear(1000, 2),

                                 nn.LogSoftmax(dim=1))

criterion = nn.NLLLoss()

*# Only train the classifier parameters, feature parameters are frozen*

optimizer = optim.Adam(model.parameters(), lr=0.001)

scheduler = lr\_scheduler.StepLR(optimizer, step\_size=7, gamma=0.1)

model = model.cuda()

total\_params = **sum**(p.numel() for p in model.parameters())

**print**(f'{total\_params:,} total parameters.')

total\_trainable\_params = **sum**(

    p.numel() for p in model.parameters() if p.requires\_grad)

**print**(f'{total\_trainable\_params:,} training parameters.')

total\_correct = 0.0

epoches = 100

valid\_loss\_min = np.Inf

for epoch in **range**(1, epoches+1):

    torch.cuda.empty\_cache()

*# keep track of training and validation loss*

    train\_loss = 0.0

    valid\_loss = 0.0

*###################*

*# train the model #*

*###################*

    model.train()

    for data, target in trainloader:

*# move tensors to GPU if CUDA is available*

        if train\_on\_gpu:

            data, target = data.cuda(), target.cuda()

*# clear the gradients of all optimized variables*

        optimizer.zero\_grad()

*# forward pass: compute predicted outputs by passing inputs to the model*

        output = model(data)

*# calculate the batch loss*

        loss = criterion(output, target)

*# backward pass: compute gradient of the loss with respect to model parameters*

        loss.backward()

*# perform a single optimization step (parameter update)*

        optimizer.step()

*# update training loss*

        train\_loss += loss.item()\*data.size(0)

        scheduler.step()

*# torch.cuda.empty\_cache()*

        total\_correct += get\_num\_correct(output, target)

*######################*

*# validate the model #*

*######################*

    model.eval()

    for data, target in validloader:

        torch.cuda.empty\_cache()

*# move tensors to GPU if CUDA is available*

        if train\_on\_gpu:

            data, target = data.cuda(), target.cuda()

*# forward pass: compute predicted outputs by passing inputs to the model*

        output = model(data)

*# calculate the batch loss*

        loss = criterion(output, target)

*# update average validation loss*

        valid\_loss += loss.item()\*data.size(0)

*# torch.cuda.empty\_cache()*

*# calculate average losses*

    train\_loss = train\_loss/**len**(trainloader.sampler)

    valid\_loss = valid\_loss/**len**(validloader.sampler)

**print**('Epoch: {} \tTraining Loss: {:.6f} \tValidation Loss: {:.6f}'.format(

        epoch, train\_loss, valid\_loss))

*# save model if validation loss has decreased*

    if valid\_loss <= valid\_loss\_min:

**print**('Validation loss decreased ({:.6f} --> {:.6f}).  Saving model ...'.format(

            valid\_loss\_min,

            valid\_loss))

        torch.save(model.state\_dict(), 'sed.pt')

        valid\_loss\_min = valid\_loss

# Chapter-8

# System testing

## 8.1 INTRODUCTION:

Testing is the debugging program is one of the most critical aspects of the computer programming triggers, without programming that works, the system would never produce an output of which it was designed. Testing is best performed when user development is asked to assist in identifying all errors and bugs. The sample data are used for testing. It is not quantity but quality of the data used the matters of testing. Testing is aimed at ensuring that the system was accurately an efficiently before live operation commands.

Testing objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, testing is a process of executing a program with intent of finding an error.

1. A successful test is one that uncovers an as yet undiscovered error.
2. A good test case is one that has probability of finding an error, if it exists.
3. The test is inadequate to detect possibly present errors.
4. The software more or less confirms to the quality and reliable standards.

## 8.2 Levels of Testing

Code testing:

This examines the logic of the program. For example, the logic for updating various sample data and with the sample files and directories were tested and verified.

Specification Testing:

Executing this specification starting what the program should do and how it should performed under various conditions. Test cases for various situation and combination of conditions in all the modules are tested.

Unit testing:

In the unit testing we test each module individually and integrate with the overall system. Unit testing focuses verification efforts on the smallest unit of software design in the module. This is also known as module testing. The module of the system is tested separately. This testing is carried out during programming stage itself. In the testing step each module is found to work satisfactorily as regard to expected output from the module. There are some validation checks for fields also. For example the validation check is done for varying the user input given by the user which validity of the data entered. It is very easy to find error debut the system.

Each Module can be tested using the following two Strategies:

1. Black Box Testing
2. White Box Testing

### 8.2.1 BLACK BOX TESTING

What is Black Box Testing?

Black box testing is a software testing techniques in which functionality of the software under test (SUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software. This type of testing is based entirely on the software requirements and specifications.

In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.



The above Black Box can be any software system you want to test. For example : an operating system like Windows, a website like Google ,a database like Oracle or even your own custom application. Under Black Box Testing , you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation.

Black box testing - Steps

Here are the generic steps followed to carry out any type of Black Box Testing.

* Initially requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

Types of Black Box Testing

There are many types of Black Box Testing but following are the prominent ones -

* Functional testing – This black box testing type is related to functional requirements of a system; it is done by software testers.
* Non-functional testing – This type of black box testing is not related to testing of a specific functionality, but non-functional requirements  such as performance, scalability, usability.
* Regression testing – Regression testing is done  after code fixes , upgrades or any other system maintenance to check the new code has not affected the existing code.

### 8.2.2 WHITE BOX TESTING

White Box Testing is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.White box testing is also known as clear, open, structural, and glass box testing.

It is one of two parts of the "box testing" approach of software testing. Its counter-part, blackbox testing, involves testing from an external or end-user type perspective. On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing. The term "whitebox" was used because of the see-through box concept. The clear box or whitebox name symbolizes the ability to see through the software's outer shell (or "box") into its inner workings. Likewise, the "black box" in "black box testing" symbolizes not being able to see the inner workings of the software so that only the end-user experience can be tested

WHAT DO YOU VERIFY IN WHITE BOX TESTING?

White box testing involves the testing of the software code for the following:

* Internal security holes
* Broken or poorly structured paths in the coding processes
* The flow of specific inputs through the code
* Expected output
* The functionality of conditional loops
* Testing of each statement, object and function on an individual basis

The testing can be done at system, integration and unit levels of software development. One of the basic goals of whitebox testing is to verify a working flow for an application. It involves testing a series of predefined inputs against expected or desired outputs so that when a specific input does not result in the expected output, you have encountered a bug.

HOW DO YOU PERFORM WHITE BOX TESTING?

  To give you a simplified explanation of white box testing, we have divided it into **two basic steps**. This is what testers do when testing an application using the white box testing technique:

**STEP 1) UNDERSTAND THE SOURCE CODE**

The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testing of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices. Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

**Step 2) CREATE TEST CASES AND EXECUTE**

The second basic step to white box testing involves testing the application’s source code for proper flow and structure. One way is by writing more code to test the application’s source code. The tester will develop little tests for each process or series of processes in the application. This  method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools as we will explain further on in this article.

Unit testing:

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­1 |
| Name of Test: ­ | Load dataset |
| Items being tested: ­ | Dataset features and labels are displayed or not |
| Sample Input: ­ | Dataset csv file |
| Expected output: ­ | All features and labels should be displayed |
| Actual output: ­ | Total data is displayed |
| **Remarks: ­** | **Pass.** |

|  |  |
| --- | --- |
| Sl # Test Case : ­ | UTC­2 |
| Name of Test: ­ | Split data |
| Items being tested: ­ | Data is divided in to train and test set |
| Sample Input: ­ | Test and train size |
| Expected output: ­ | Dataset is divided in to 2 parts |
| Actual output: ­ | Based on given test size data is divided and stored in train and test sets |
| Remarks: ­ | pass |

**Integration Testing:**

Integration testing is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Integration testing is defined as the testing of combined parts of an application to determine if they function correctly. It occurs after unit testing and before validation testing. Integration testing can be done in two ways: Bottom­up integration testing and Top­down integration testing.

* + 1. **Bottom­up Integration**

This testing begins with unit testing, followed by tests of progressively higher­level combinations of units called modules or builds.

* + 1. **Top­down Integration**

In this testing, the highest­level modules are tested first and progressively, lower­level modules are tested thereafter.

In a comprehensive software development environment, bottom­up testing is usually done first, followed by top­down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic actual situations. Table 6.5 shows the test cases for integration testing and their results

|  |  |
| --- | --- |
| Sl # Test Case : ­ | ITC­1 |
| Name of Test: ­ | Train Model |
| Item being tested: ­ | Model fit is performed |
| Sample Input: ­ | Train x and train y |
| Expected output: ­ | Fit is performed |
| Actual output: ­ | Training is done and accuracy is displayed |
| Remarks: ­ | Pass. |

|  |  |
| --- | --- |
| Sl # Test Case : ­ | ITC­2 |
| Name of Test: ­ | Accuracy calculation |
| Item being tested: ­ | If accuracy of each algorithm is calculated |
| Sample Input: ­ | Test x and test y |
| Expected output: ­ | Accuracy of each algorithm |
| Actual output: ­ | Accuracy of each model |
| Remarks: ­ | Pass. |

**System testing**:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black­box testing, and as such, should require no knowledge of the inner design of the code or logic. System testing is important because of the following reasons:

System testing is the first step in the Software Development Life Cycle, where the application is tested as a whole.

The application is tested thoroughly to verify that it meets the functional and technical specifications.

The application is tested in an environment that is very close to the production environment where the application will be deployed.

System testing enables us to test, verify, and validate both the business requirements as well as the application architecture.

System Testing is shown in below tables

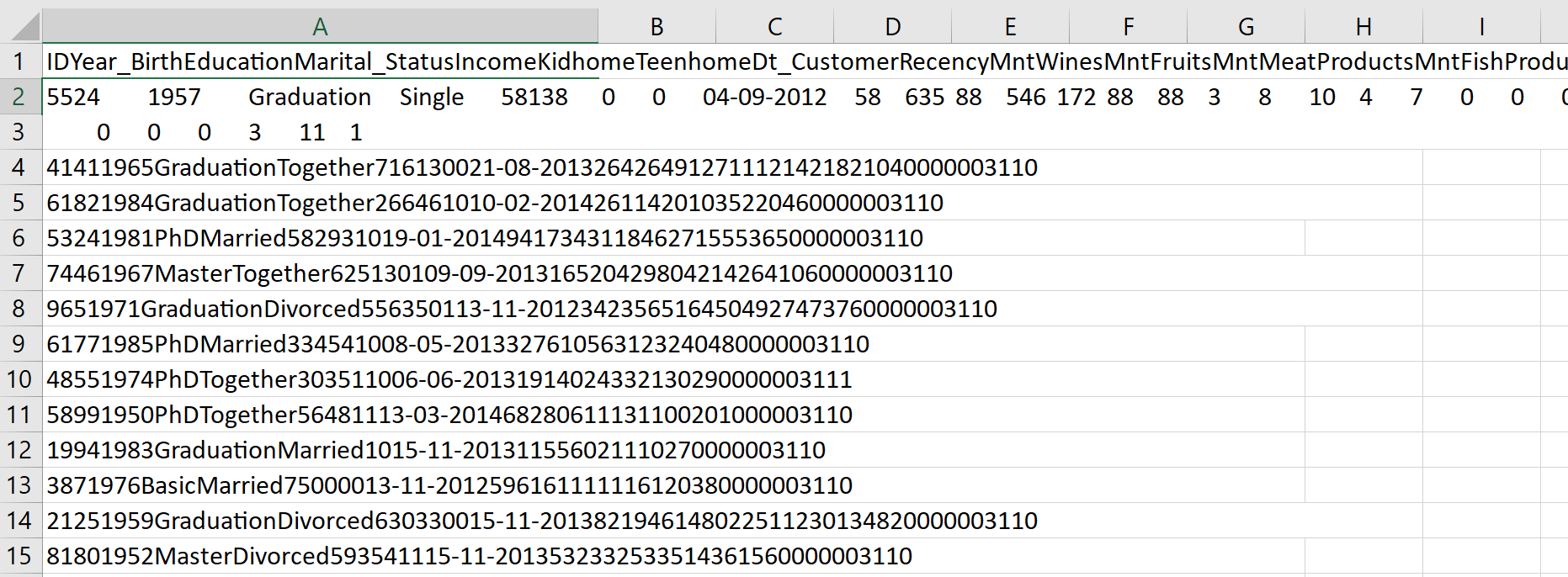
|  |  |
| --- | --- |
| Sl # Test Case : ­ | STC­1 |
| Name of Test: ­ | System testing in various versions of OS |
| Item being tested: ­ | OS compatibility. |
| Sample Input: ­ | Execute the program in windows XP/ Windows­7/8 |
| Expected output: ­ | Performance is better in windows­7 |
| Actual output: ­ | Same as expected output, performance is better in windows­7 |
| Remarks: ­ | Pass |

# CHAPTER-9

# Output Screens

## 9.1 Dataset SCREEN

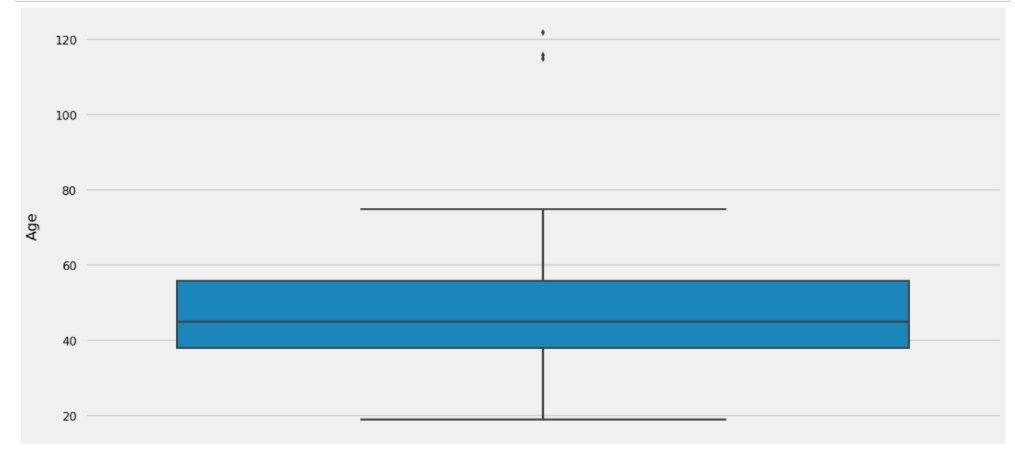
**DATASET**

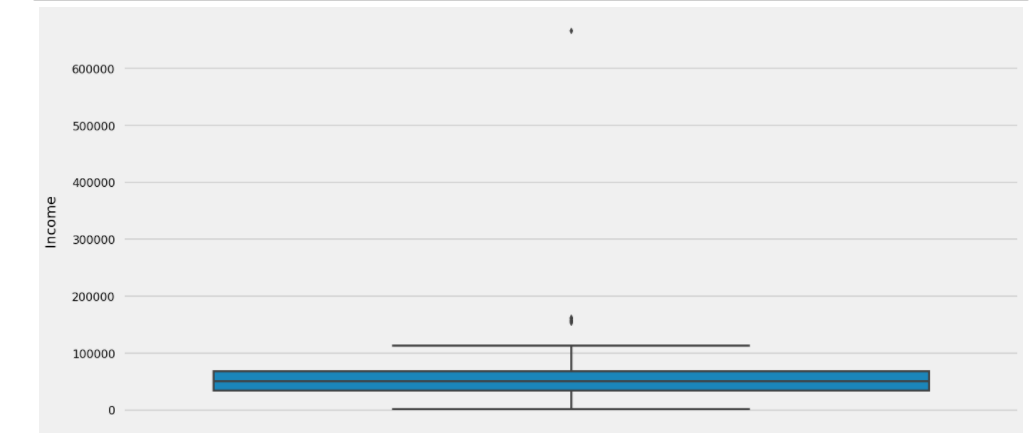
****

**FEATURES:**

ID Year Birth Education Marital Status Income Kid home Teen home Dt\_Customer Recency MN twines Mn Fruits MntMeatProducts MntFishProducts MntSweetProducts MntGoldProds NumDealsPurchases NumWebPurchases NumCatalogPurchases NumStorePurchases NumWebVisitsMonth AcceptedCmp3 AcceptedCmp4 AcceptedCmp5 AcceptedCmp1 AcceptedCmp2 Complain Z\_CostContact Z\_Revenue Response

**Removing Outliers:**

****

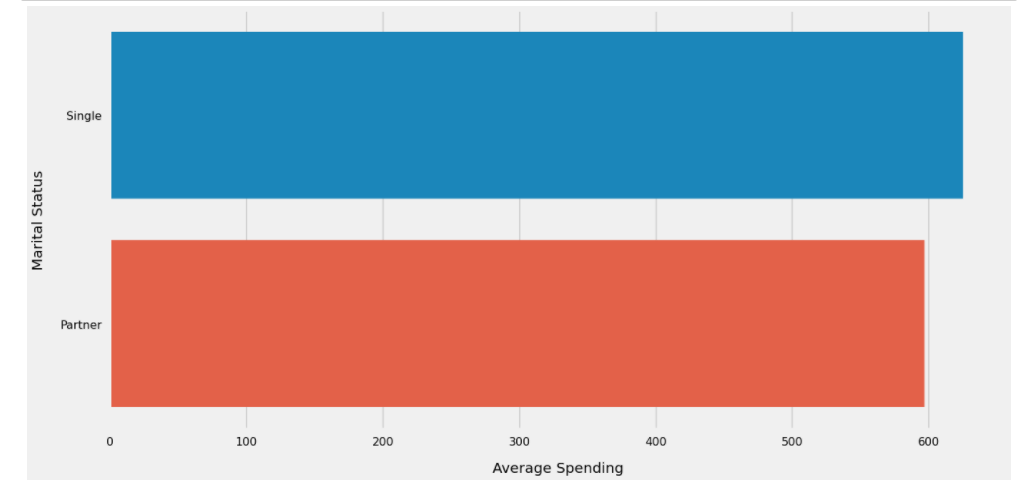
****

1. There are some customers aged above 100. This is unlikely to happen. Let's drop those customers from data

2. There are some customers who are earning more than 120,000 and some of them even more than 600,000. They are clearly the outliers in the data, so we will leave them out

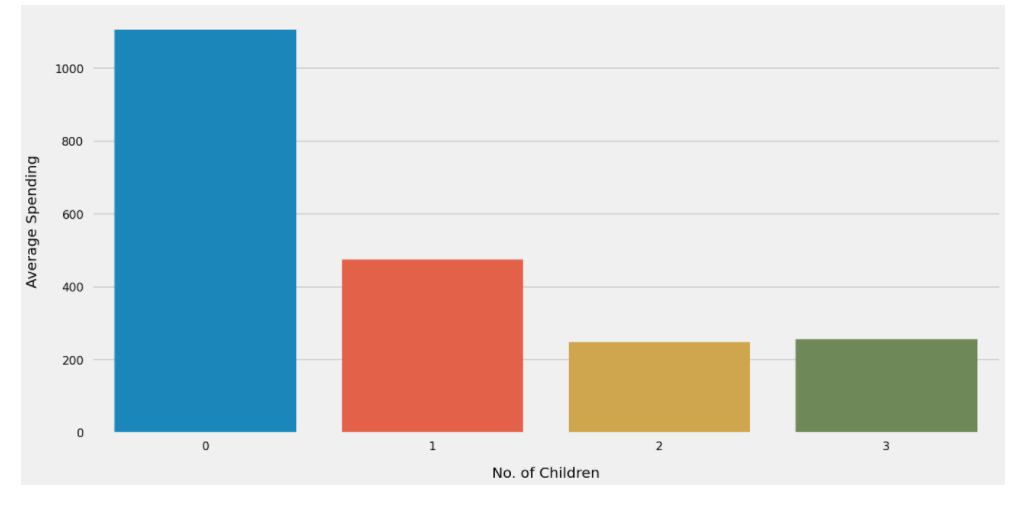
**DATA ANALYSIS:**

**Average spending:**

****

Despite being the minority, the Singles spent more money on the average as compared to the customers having partners

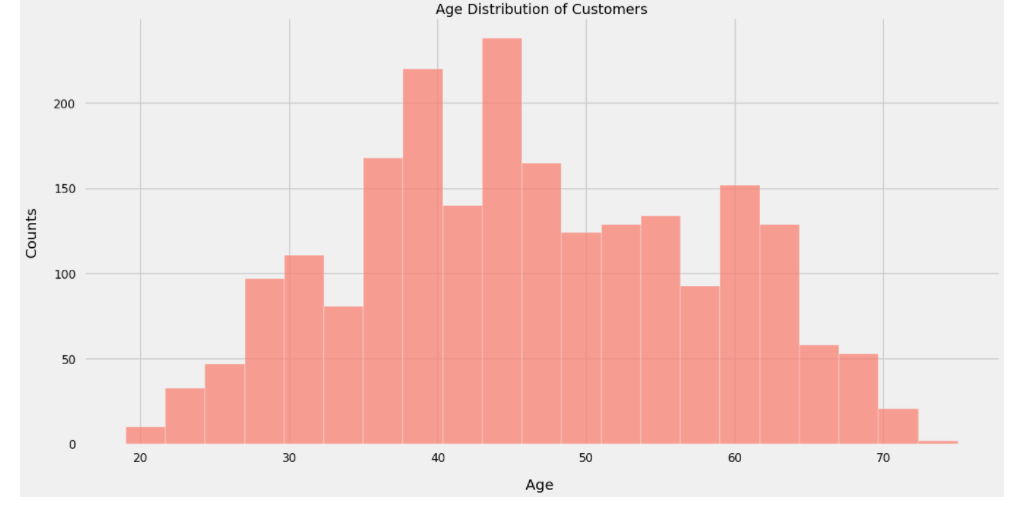
**Average Spendings:**

****

1. Customers who don't have any children at home spent higher than the customers having 1 children

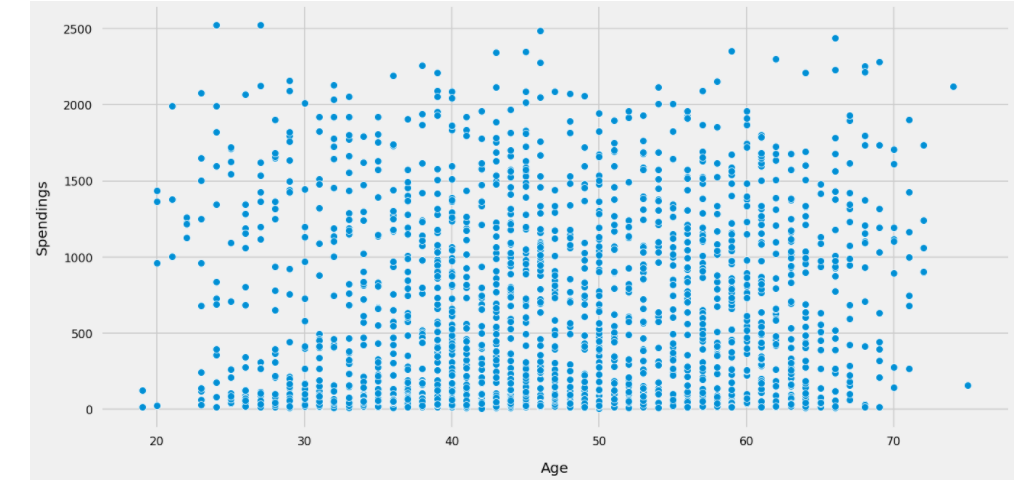
2. The customers having 1 children are spending higher than the customers havin 2 and 3 children

**Age Distribution of Customers:**

****

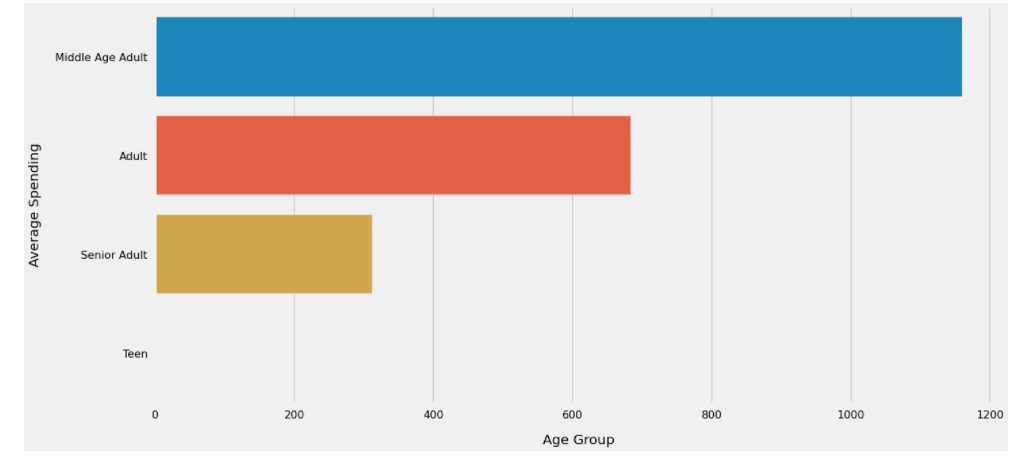
Age of the customers is nearly normally distributed, with most of the customers aged between 40 and 60.

**Relation ship age vs spending:**

****

There doesn't seem to be any clear relationship between age of customers and their spending habits

**Average Spendings:**

****

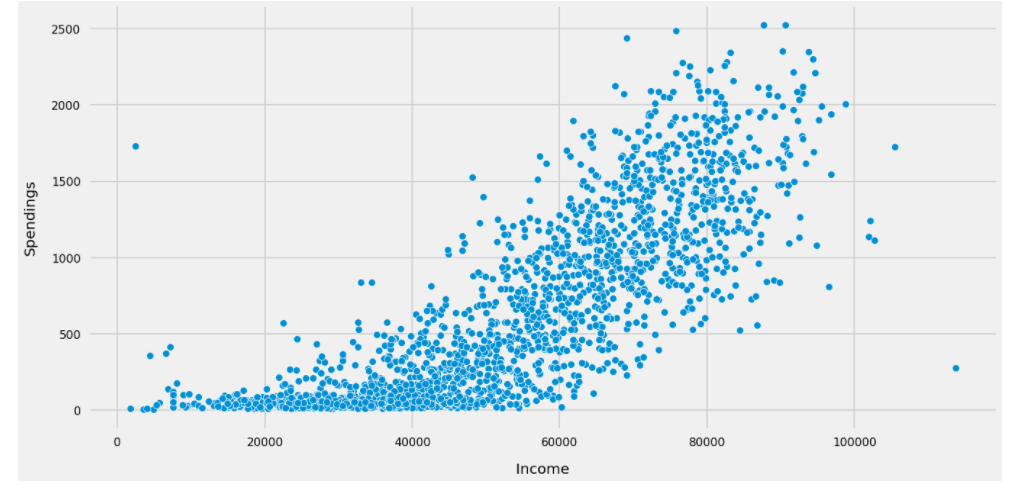
Middle age adults spent much more than the other age groups

**Income Distribution of Customers:**

****

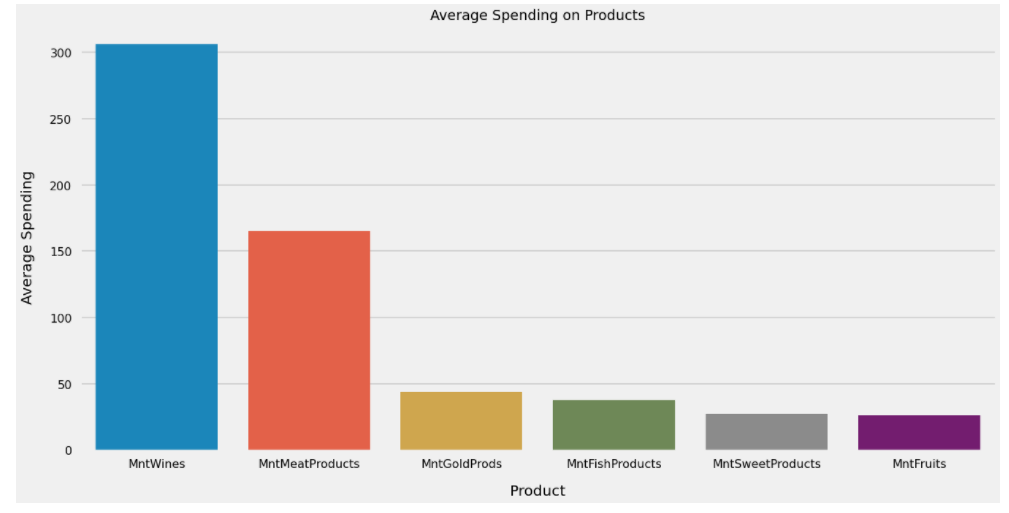
The salaries of the customers have normal distribution with most of the customers earning between 25000 and 85000

**Relationship: Income vs Spendings:**

****

The relationship is linear. Customers having higher salaries are spending more

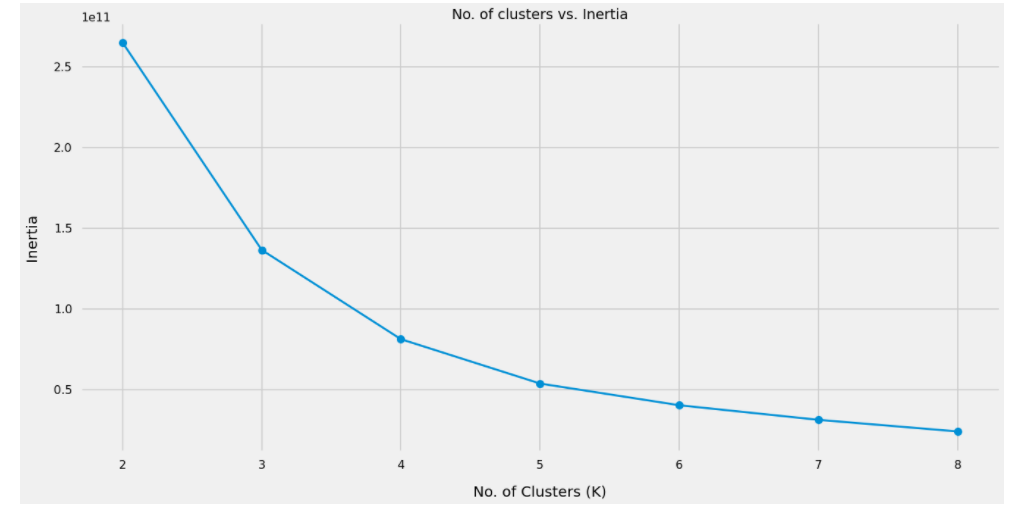
**Most Bought Products:**



1. Wine and Meats products are the most famous products among the customers

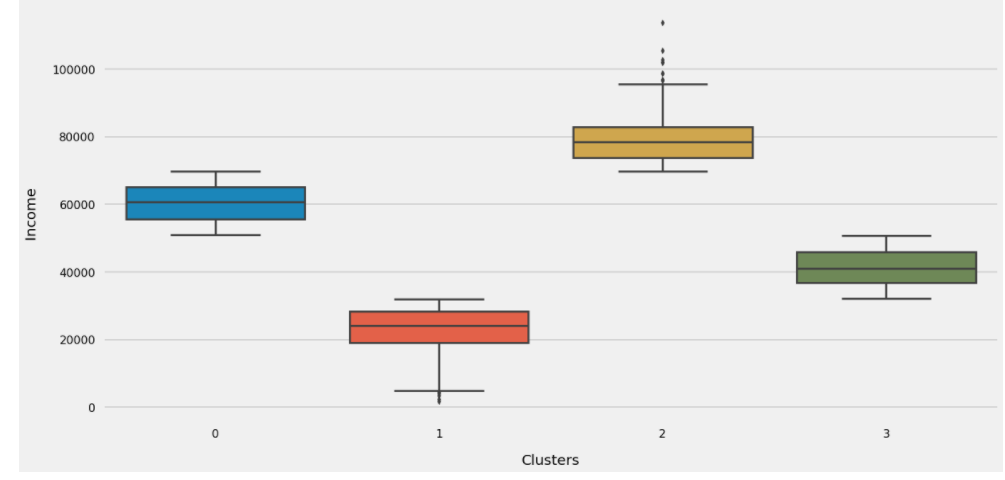
2. Sweets and Fruits are not being purchased often

**CLUSTURING METHODS:**

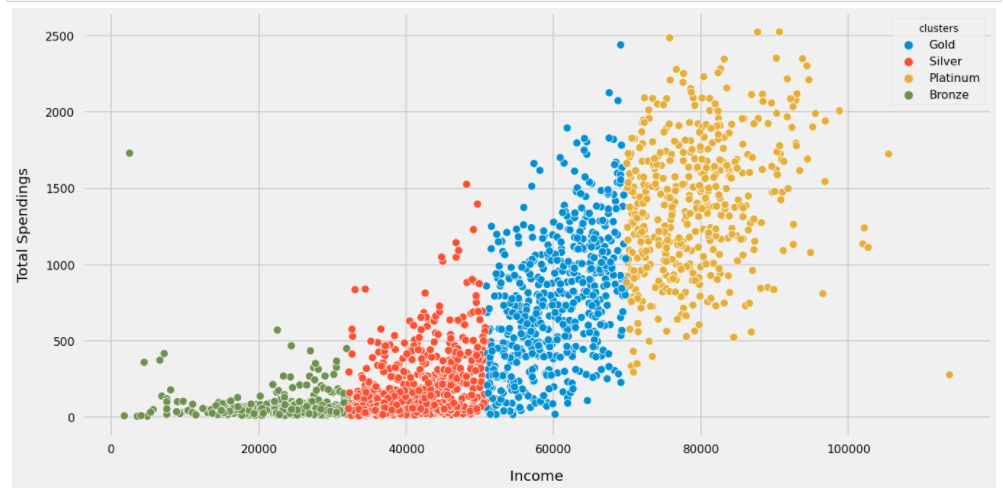


Based on the above plot we will segment the customers into 4 clusters, as the inertia value donot decrase much after 4 clusters

**Clusters Identification:**

****

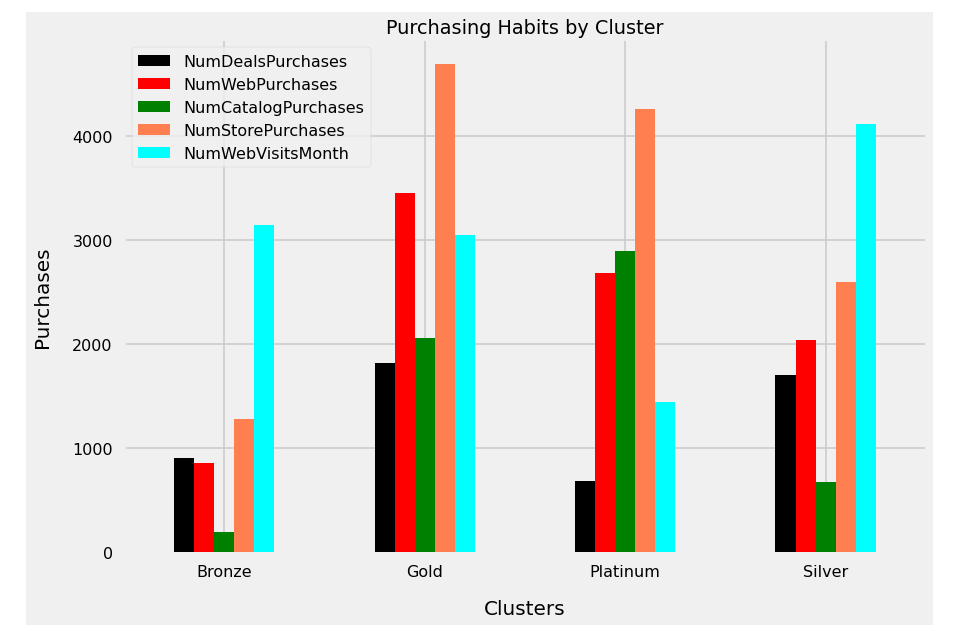
**INCOMMING VS SPENDING CLUSTES:**

****

1. The 4 clusters can easily be identified from the above plot

2. Those earning more are also spending more

**PURCHASE HABBITS BY CLUSTERS:**



1. Platinum and Gold Customers mostly likely to do store purchasing

2. Most of the web and catalog purchases are also done by the customers from Platinum and Gold segments

3. Silver and Gold categoriesnalso like to buy from the stores

4. Deal purchases are common among the Gold and Silver customers

5. Silver category customers made the most number of web visits while customers from Platinum segment have least web visits

# CONCLUSION and Future work

.

Although there are many approaches proposed for identify-ing potential customers, a machine learning approach is rather rare. Our research is ﬁrst of its kind where machine learnings used to study customer’s purchase behavior for a retail superstore. The experiment of potential customer classiﬁcationachieved prediction accuracy up to 99.4% with recall 98.9%and precision 99.7%. We engineered features to capture the relationship between categories, items, quantity, measurement unit, and sales. The difference in result between native and engineered feature is 42.6%. A business can be highly beneﬁted by identifying their potential customer correctly. The potential customer can be approached with a customized marketing plan which can increase the sale of a business. As the research is based on the data acquired from the grocery superstore Taradin as a case study, the area of the research only limits to the grocery superstore itself and the industry. However, this results possibly generalize enough to adhere to the problem deﬁnition for the whole grocery superstore system in Bangladesh. Tara dins existing customers data is considered for this research. As a result, if there Isan unexplored customer segment which is not in this dataset, will remain out of this research scope. In future, machine learning can be used to understand customers behavior, a product of interest, buying frequency which will help to plan more appropriate marketing plan and efﬁcient supply chain management.

.

# 

# REFERENCES

[1] Berry, Leonard L. ”Relationship marketing of servicesgrowing interest,

emerging perspectives.” Journal of the Academy of marketing science

23, no. 4, pp. 236-245, (1995).

[2] Reichheld, Frederick F., and Jr WE Sasser. ”Zero defections: Quality

comes to services.” Harvard business review 68, no. 5, pp. 105-111,

(1990).

[3] Collobert, Ronan, and Jason Weston. ”A uniﬁed architecture for natural

language processing: Deep neural networks with multitask learning.” In

Proceedings of the 25th international conference on Machine learning,

pp. 160-167. ACM, 2008.

[4] Goldberg, Yoav. ”A primer on neural network models for natural

language processing.” Journal of Artiﬁcial Intelligence Research 57, pp.

345-420, (2016).

[5] Ohn-Bar, Eshed, and Mohan Manubhai Trivedi. ”Looking at humans in

the age of self-driving and highly automated vehicles.” IEEE Transac-

tions on Intelligent Vehicles 1, no. 1, pp. 90-104, (2016).

[6] Zhang, Xiang, Junbo Zhao, and Yann LeCun. ”Character-level con-

volutional networks for text classiﬁcation.” In Advances in neural

information processing systems, pp. 649-657. 2015.

[7] Rosten, Edward, and Tom Drummond. ”Machine learning for high-speed

corner detection.” In European conference on computer vision, pp. 430-

443. Springer, Berlin, Heidelberg, 2006.

[8] Chapelle, Olivier, Patrick Haffner, and Vladimir N. Vapnik. ”Support

vector machines for histogram-based image classiﬁcation.” IEEE trans-

actions on Neural Networks 10, no. 5, pp. 1055-1064, (1999).

[9] Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. ”Speech

recognition with deep recurrent neural networks.” In Acoustics, speech

and signal processing (icassp), 2013 ieee international conference on,

pp. 6645-6649. IEEE, 2013.

[1] Berry, Leonard L.” Relationship marketing of servicesgrowing interest,emerging perspectives.” Journal of the Academy of marketing science23, no. 4, pp. 236-245, (1995).

[2] Reichheld, Frederick F., and Jr WE Sasser. ”Zero defections: Qualitycomes to services.” Harvard business review 68, no. 5, pp. 105-111,(1990).

[3] Collobert, Ronan, and Jason Weston. ”A uniﬁed architecture for naturallanguage processing: Deep neural networks with multitask learning.” InProceedings of the 25th international conference on Machine learning,pp. 160-167. ACM, 2008.

[4] Goldberg, Yoav. ”A primer on neural network models for naturallanguage processing.” Journal of Artiﬁcial Intelligence Research 57, pp.345-420, (2016).

[5] Ohn-Bar, Eshed, and Mohan Manubhai Trivedi. ”Looking at humans inthe age of self-driving and highly automated vehicles.” IEEE Transac-tions on Intelligent Vehicles 1, no. 1, pp. 90-104, (2016).

[6] Zhang, Xiang, Junbo Zhao, and Yann LeCun. ”Character-level con-volutional networks for text classiﬁcation.” In Advances in neuralinformation processing systems, pp. 649-657. 2015.

[7] Rosten, Edward, and Tom Drummond. ”Machine learning for high-speedcorner detection.” In European conference on computer vision, pp. 430-443. Springer, Berlin, Heidelberg, 2006.

[8] Chapelle, Olivier, Patrick Haffner, and Vladimir N. Vapnik. ”Supportvector machines for histogram-based image classiﬁcation.” IEEE trans-actions on Neural Networks 10, no. 5, pp. 1055-1064, (1999).

[9] Graves, Alex, Abdel-rahman Mohamed, and Geoffrey Hinton. ”Speechrecognition with deep recurrent neural networks.” In Acoustics, speechand signal processing (icassp), 2013 ieee international conference on,pp. 6645-6649. IEEE, 2013.