APPENDIX 1

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

ENABLING MACHINES TO PREDICT CUSTOMER BEHAVIOR USING VOTING CLASSIFIER TECHNIQUE

A Report for Project 2

Submitted by

ABHISHEK SINGH

(1613112003)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING WITH SPECIALIZATION OF DATA ANALYTICS

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

Under the Supervision of

Dr. SATYAJEE SRIVASTAVA, Ph.D.,

Associate Professor

APRIL / MAY- 2020

APPENDIX 2

SCHOOL OF COMPUTING AND SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

Certified that this project report **"ENABLING MACHINES TO PREDICT CUSTOMER BEHAVIOR USING VOTING CLASSIFIER TECHNIQUE"** is the bonafide work of "**ABHISHEK SINGH (1613112003)**" who carried out the project work under my supervision.

SIGNATURE OF HEAD

Dr. MUNISH SHABARWAL,
Ph.D (Management), Ph.D (CS)
Professor & Dean,
School of Computing Science & Engineering

SIGNATURE OF SUPERVISOR

Dr. SATYAJEE SRIVASTAVA,
Ph.D(CS)
Associate Professor,
School of Computer Science & Engineering

ABSTRACT

In the retail industry, understanding existing customers' behavior based on their buying behavior is one thing that can drive business from bottom to top. Customer on an individual basis as well as on a collective basis shows his/her interest towards particular business products, interest being personal or towards the best product of that business. At an initial level, some businesses succeeds in understanding their customer needs and behavior but when businesses grow up and ready to expand, they need some powerful technologies that can do work for them for such a huge amount of customer data. In this project, we have discussed a technique to segment customers in different categories based on their past behavior and then built a voting classifier model to predict what items a particular customer will buy in future transactions.

APPENDIX 3

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

B2C	Business-to-Customer
CSB	Customer Segmentation and Behavior
EDA	Exploratory Data Analysis
KNN	K-Nearest-Neighbors
NER	Named Entity Recognition
NLP	Natural Language Processing
PCA	Principal Component Analysis
POS	Parts-of-Speech
RFM	Recency, Frequency and Monetory
STP	Segmentation-Targeting-Positioning
SVC	Support-Vector-Classifier

CHAPTER 1

INTRODUCTION

1.1 Overall Description

Retail industry is one of the world's largest industries that play a crucial role in setting up of economic condition for any country. This industry has a broad customer base that makes it this much large [7]. Retail industry basically fulfills an individual's day-to-day needs. Every customer has different needs and preferences and that can be observed with the help of his/her buying behavior itself. Observing customer's behavior towards your business products like what type of products are mostly searched by a particular customer and what products are actually converted to transaction and how frequently a customer comes to business for the fulfillment of needs etc. All these come under an umbrella of simple process i.e. *Customer Segmentation*. As the name describes itself, customer segmentation is a process of dividing the customer base of the business into several (2 or more) subgroups o the basis of specific characteristics so as to increase the sale of each category of products. This also leds to less marketing expenditure. In brief, the customer base of any business has two different categories i.e. Existing Customers and Potential Customers. Customized marketing strategy is highly adopted by businesses these days. Businesses follow STP approach which consists of three stages: Segmentation – Targeting – Positioning [8]. It helps a business to its product sales with lesser marketing strategy. Voting is considered to be one of the efficient and

simplest methods of combining the predicted results of multiple algorithms of machine learning. The main idea behind voting classifier technique is to create parallel dedicated models followed by finding the model accuracy for each of the individual models followed by creating a single robust model which is trained by those individual models and it then predicts final output based on merged majority of voting for each and every output class. Identifying segments of customers and their behavioral patterns over different time intervals, is an important application for businesses, especially in case of the last tier of the online retail chain which is concerned with "electronic Business-to-Customer relationship" (B2C) . This is particularly important in dynamic and ever-changing markets, where customers are driven by ever changing market competition and demands. This could lead to the prediction of 'churn', or which customers are leaving the company's loyalty. Also, the provision of customized service to the customers is vital for a company to establish long lasting and pleasant relationship with consumers [14]. It has also been observed that keeping old customers generates more profit than attracting new ones. So, customer retention is a big factor too. So, there is always a trade-off between customer benefits and transaction costs, which has to be optimized by the managers.

1.2 Purpose

The business problem that we are trying to solve in this project is to find the efficient way to categorize the consumer base in different segments based on their buying behavior. Following this, we have also been able to predict

what kind of products a customer will probably be buying in future that will be based on their segments they belong and the purchase they had already made.

1.3 The need for Customer Behavior Prediction

The emergence of the business-to-customer (B2C) markets has resulted in various studies on developing and improving customer retention and profit enhancement. This is mainly due to the retail business becoming increasingly competitive with costs being driven down by new and existing competitors. In general, consumer markets have several characteristics such as repeat buying over the relevant time interval, a large number of customers, and a wealth of information detailing past customer purchases. In those markets, the goal of CSB is to identify a customer, understand and predict the customer-buying pattern, identify an appropriate offer, and deliver it in a personalized format directly to the customer [14]. One typical example of the CSB model corresponds to the case of an online retail shop which sells various products through internet and performs transaction directly with customers through the internet. An online retail shop defines a customer as a person who has already bought products or performed a transaction with the shop. The exponential growth of the Internet has led to a hoard of customer and market data to the market managers. The increased availability of individual consumer data presents the possibility of direct targeting of individual customers. That is, the abundance of customer information enables marketers to take advantage of individual - level

purchase models for direct marketing and targeting decisions. But, such an enormous amount of data can be a huge clutter, and it can become cumbersome to draw meaningful conclusions from such raw data. This is where the utility of customer behavior prediction using Data mining techniques comes in. The major customer values or characteristics that are used to measure purchase behavior of customers include Recency, Frequency, and Monetary values (RFM). RFM measures provide information on what customers do. Recency tells how long it has been since each customer made the last purchase. Frequency tells how many times each customer has purchased an item during certain intervals of time. Monetary tells how much each customer has spent in total. Monetary measures the total expenditure of the customer for a number of transactions over a period of time. These characteristics may be the most important in determining the likely profitability of a particular product or an individual customer, so they are used to segregate the list of customers into groups having different characteristics based on the RFM values.

1.4 Relevance of Data Mining towards CSB

Data mining techniques are the processes designed to identify and interpret data for the purpose of understanding and deducing actionable trends and designing strategies based on those trends. Data mining techniques extract the raw data, and then transform them to get the transformed data, and then get meaningful patterns among the transformed data. As businesses evaluate their investments on marketing activities, they tend to focus on their data mining techniques and capability. How to learn more about customers and

their inclination towards particular products, use that information to make appropriate choices to customers, and understand which marketing strategies can succeed in long term customer satisfaction and retention. Managers can understand their customer by evaluating customer behavior, customer segregation, customer profiles, loyalty (how long have they been associated with the company) and profitability (which products can be targeted to the particular customer so as to extract maximum profits). Data Mining helps managers to identify valuable patterns contained in raw data and their relations so as to help the major decisions. The basic structure of CSB model lifecycle is shown in fig.1. The model can have two initiating pts. Firstly, the customer does some purchase and then the data is measured and evaluated. Afterwards, the company mines the evaluated data and then they can have an understanding of the patterns that the customer shows while purchasing. With the help of that data, the organization can formulate its steps to maximize or optimize its business plans. Secondly, the organization takes some action for improving the customer's satisfaction by making a good informative offer, and then studies the actions taken by the customer.

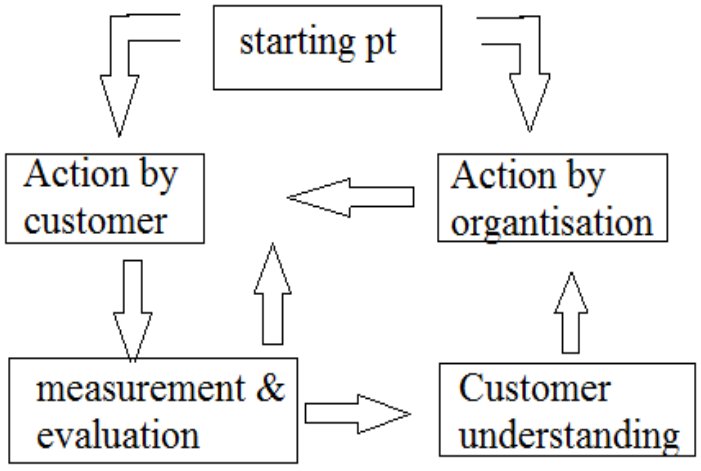

Fig. 1 The basic CSB Cycle

1.5 Related Work

Customer behavior analysis is based on consumer buying behavior, with the customer playing three distinct roles of a user, payer and buyer. Relationship marketing is an important 12 aspect for customer purchase analysis as it has an importance in the research of the marketing through the re-affirmation of the importance of the customer or buyer. A greater importance is also placed on existing consumer retention, customer relationship management, personalization, customization and one-to-one marketing. Customer understanding is the heart of CSB. It is the basis for optimizing customer lifetime value, which in turn engulfs customer segmentation and actions to maximize customer conversion, retention, loyalty and profitability. Proper customer understanding and action ability lead to increased customer lifetime value. Improper customer understanding can lead to catastrophic actions. A customer can be a user, purchaser, influence maker etc. Therefore the transaction data query may be have different types of inquiries, which include, suggestions, queries, requisitions, and reclamations. In the "Examination of Customer's Inquiry" step, we can scrutinize which type of query has been placed by the customer and where it will be forwarded.

There are varieties of Algorithmic techniques available to perform customer segmentation based on their buying behavior and some other characteristics also.

In [1], Abdullah Al-Mudimigh, Farrukh Saleem and Zahid Ullah evaluate and analyze the customer buying pattern by using rule induction process on clustered data from the customer's database with reference to the customer

query.

In [2], Euiho Suh , Seungjae Lim , Hyunseok Hwang and Suyeon Kim lay their focus on studying anonymous customers and then they sequentially mine the data through data preprocessing and then extracting association rules from them.

In [3] authors have classified the data for customer segmentation in two categories i.e. Internal data and External data. They had categorized the customer profile and transaction history data as internal data and data like cookies, server log and survey data were categorized in external data. They have also categorized methods like Magento, Business Rule, Quantile membership, Customer Profiling, Supervised clustering etc. as Simple technique, Target technique, RFM technique and unsupervised technique.

In [4] had used credit card transaction data for model building and prepared predictive models at segment-level utilizing pattern based clustering approach. They devised two matrices i.e. Fluctuate-rate matrix and monetary matrix and performed clustering on both of the matrices to discover various customer characteristics. Further, they used those characteristics to build consumption based consumer segmentation model.

In [5] proposed hybrid classifier technique using Decision tree and KNN for customer behavior analysis which outperformed the performance of previously accepted Naïve Bayes model by many researchers. Hybrid classifier has got accuracy of 90.75% that had a significant difference from

the accuracy of Naïve Bayes classifier that was only 74.11%.

In [6] compared the performance of 3 individual classification algorithms i.e. Random Forest, Support Vector Machine and Logistic Regression with that of Majority Voting algorithm and proved successful in achieving better Precision, Recall, F-Measure and Accuracy as well.

1.6 Problem Model Formulation

A typical online retail mart has thousands of transactions stored in its database. It handles hundreds, maybe thousands of customers per day. All these data transactions also referred to as 'orders' need to be clustered according to some chosen parameters, and then a meaningful pattern or rules are to be inferred. For example, if a customer buys a certain product, is it necessary that he will buy another product related to that purchase, i.e., how to infer a rule 13 among 2 or more purchases of the customer. To incorporate a pattern to recognize a group of customers having similar purchase behavior.

This study investigates the efficient way of finding the optimal number of customer segments to categorize the consumer base accordingly. Also, building a voting classifier with the best possible combination of classification algorithms among Linear SVC, Logistic Regression, Random Forest, Decision Tree, KNN, and XGBoost.

Chapter 2

EXISTING APPROACH

2.1 The Customer Segmentation Approach

Customer segmentation is one of the most important area of knowledge based marketing. In case of online retail stores, it is really a challenging task, as data bases are large and multidimensional. In previous approach, they considered a clustering algorithm, which is based on a Vector Quantization based algorithm, and can be effectively used to automatically assign existing or new arriving customers into the respective clusters.

2.1.1 A background of the Vector Quantization based clustering algorithm

It is an efficient algorithm designed by Linde, Buzo and Gray for the design of good block or vector quantizes with quite general distortion measurements is developed for use on either known probabilistic source descriptions or on a long training sequence of data, incorporated herein by reference. The algorithm involves no differentiation; hence it works well even when the distribution has discrete components, as is the case when a sample distribution obtained from a training sequence is used. As with the common variation techniques, the algorithm produces a quantized meeting necessary but not sufficient condition for optimality [10]. Usually, however, at least local optimality is ensured in both approaches.

An N-level k-dimensional ‘quantizer’ is a mapping, q; that assigns to each input vector, x = (x_0, -, X_{k-1}), a reproduction vector, x^ = q(x), drawn from a finite reproduction alphabet, A = bi; i = 1, -, N} [1]. The level N describes the number of times the division of the codebook occurs. The quantizer is completely described by the reproduction alphabet (or codebook) A together with the partition, S = {Si; i = 1, …., N), of the input vector space into the sets Si = {x: q(x) =vi) of input vectors mapping into the i^{th} reproduction vector (or codeword), Such quantizers are also called block quantizers, vector quantizers, and block source codes. Here the input vectors x can be any kind of customer RFM values. Here it is assumed that the distortion caused by reproducing an input vector x by a reproduction vector i is giveri by a nonnegative distortion measure d(x, x^). Many such distortion measures have been proposed in the literature [10]. The most common for reasons of mathematical convenience is the squared error distortion, which has been used in the implementation of the algorithm.

$$d(x, x\hat{}) = \sum_{i=0}^{k-1} |x_i - x\hat{}_i| \qquad \text{(eq. 1)}$$

An N-level quantizer will be said to be optimal (or globally optimal) if it minimizes the expected distortion, that is, is optimal if for all other quantizers 4 having N reproduction vectors D (q*) < D (q) [10]. A quantizer is said to be locally optimum if D (q) is only a local minimum, that is, slight changes in q cause an increase in distortion. The goal of block quantizer design is to obtain an optimal quantizer if possible and, if not, to obtain a locally optimal and hopefully “good” quantizer. Several such algorithms have been proposed in the literature for the computer-aided design of locally

optimal quantizers.

2.2 The Association rules based approach for customer purchase predictions

Association rules are if-then statements that help to show the probability of relationships between data items within large data sets in various types of databases. Association rule mining has a number of applications and is widely used to help discover sales correlations in transactional data.

2.2.1`A background of the Association rules based data mining approach

Association rules are like classification criteria. There are generally the left-hand side of the rule, known as the antecedent and the right hand side of the rule, known as the inference part. Association rules were initially applied to analyze the relationships of product items purchased by customers at retail stores. In data mining, association rules are descriptive patterns of the form X=>Y, where X and Y are statements regarding the values of attributes of an instance in a database. X is termed the left-hand-side (LHS), and is the conditional part of an association rule. Meanwhile, Y is called the right hand side (RHS), and is the consequent part. The most typical application of association rules is market basket analysis, in which the market basket comprises the set of items (namely item set) purchased by a customer during a single store visit [12]. It is also known as the Shopping cart analysis of the customer purchases. The process of Association rule mining approach usually finds out a huge number of rules. These rules are then pruned down

on the basis of their "Coverage" value, which is defined as the number of instances from the whole set, where the rule predicts correctly, and their accuracy (the same number expressed as the proportion of instances to which the rule correctly applies). Nowadays, what we call coverage is often called as "support" and what we call 21 accuracy is also called "confidence". We are only interested in association rules with high coverage values. The distinction between LHS and RHS of the association rules seek combinations of attribute – value pairs that have predefines minimum coverage. These are called item-sets. An attribute value pair is called an item. It typically comes from the process of "Market basket analysis" where the store manager analyzes the different items purchased by the customer in a single purchase, and tries to find out association rules among them.

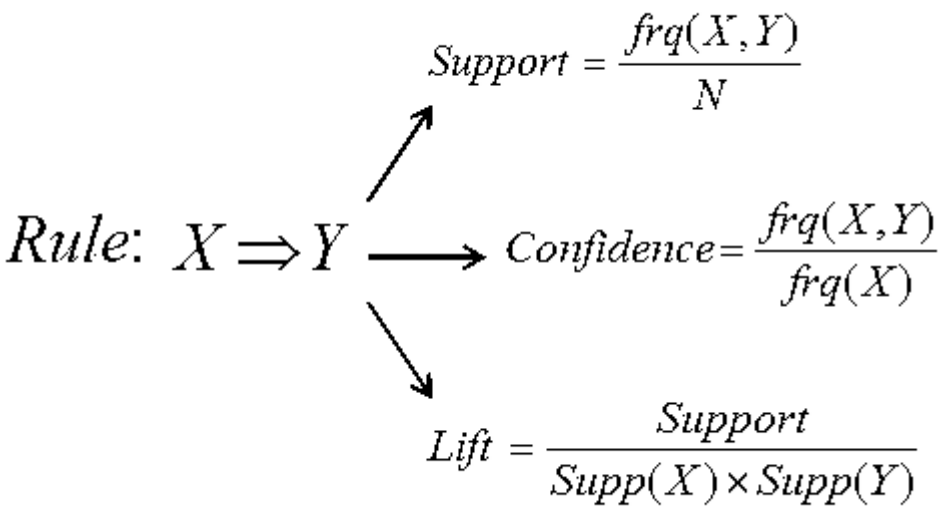

Fig. 2 Formula Structure of Association Rule Mining

2.2.2 A Description of the Association rules mining model proposed at that time

That study involves the Apriori algorithm, used to detect rules and prune them according to their coverage. ‘Apriori’ is the most basic algorithm for learning association rules. Apriori is designed to operate on databases containing various kinds of transactions (for example, collections of items bought by customers, or details of a website surfing). As is common in association rule mining, given a set of item-sets (for instance, sets of retail transactions, each listing individual items purchased, in this study), the algorithm attempts to find subsets which are common to at least a minimum number K of the item sets [14]. Apriori uses a "bottom up" approach for its execution, where the required subsets are extended one item at a time (a step known as candidate generation), and such candidates are tested against the data. The algorithm terminates when no further successful items can be added to the existing item sets. ‘Apriori’, while very basic and historically important, suffers from a number of shortcomings or trade-offs. Candidate generation step generates large numbers of subsets (the algorithm attempts to load up the candidate set with as many as possible before each scan of the database). And with increase in the number of item sets, the computational over head also increases.

It has been considered here, the application of the Apriori algorithm for incremental stages, firstly for a 1-itemset, then 2-itemset and finally for a 3-itemset, each of whom have shown varying results. The 1 item set approach applies the formulation of rules based on the criteria as to “if item X purchased”. That is it tries to find out the occurrence of individual items

from a number of orders. Then the results are stored, and a pool of rules is obtained which are denoted by “occurrence” here.

For the 2-item set analysis, the market basket is again scrutinized and rules of the form “if item X purchased then item Y purchased” are found out. These are again stored. Now, the coverage of the rules is found out as the ratio of the number of instances where the entire is true for the occurrences and the number of instances where the antecedent is true. Based on this criterion, the rules are pruned again and the qualified rules are stored for further perusal.

For the 3-item set analysis, the market basket analysis is done and rules of the form “if item X and Y purchased then item Z purchased” are found out. These are again stored. Now, the coverage of the rules is found out same as that in 2-item set, as the ratio of the number of instances where the entire is true for the occurrences and the number of instances where the antecedent is true. Based on this criterion, the rules are pruned again and the qualified rules are stored for further perusal.

Chapter 3

PROPOSED APPROACH

3.1 Architectural Diagram

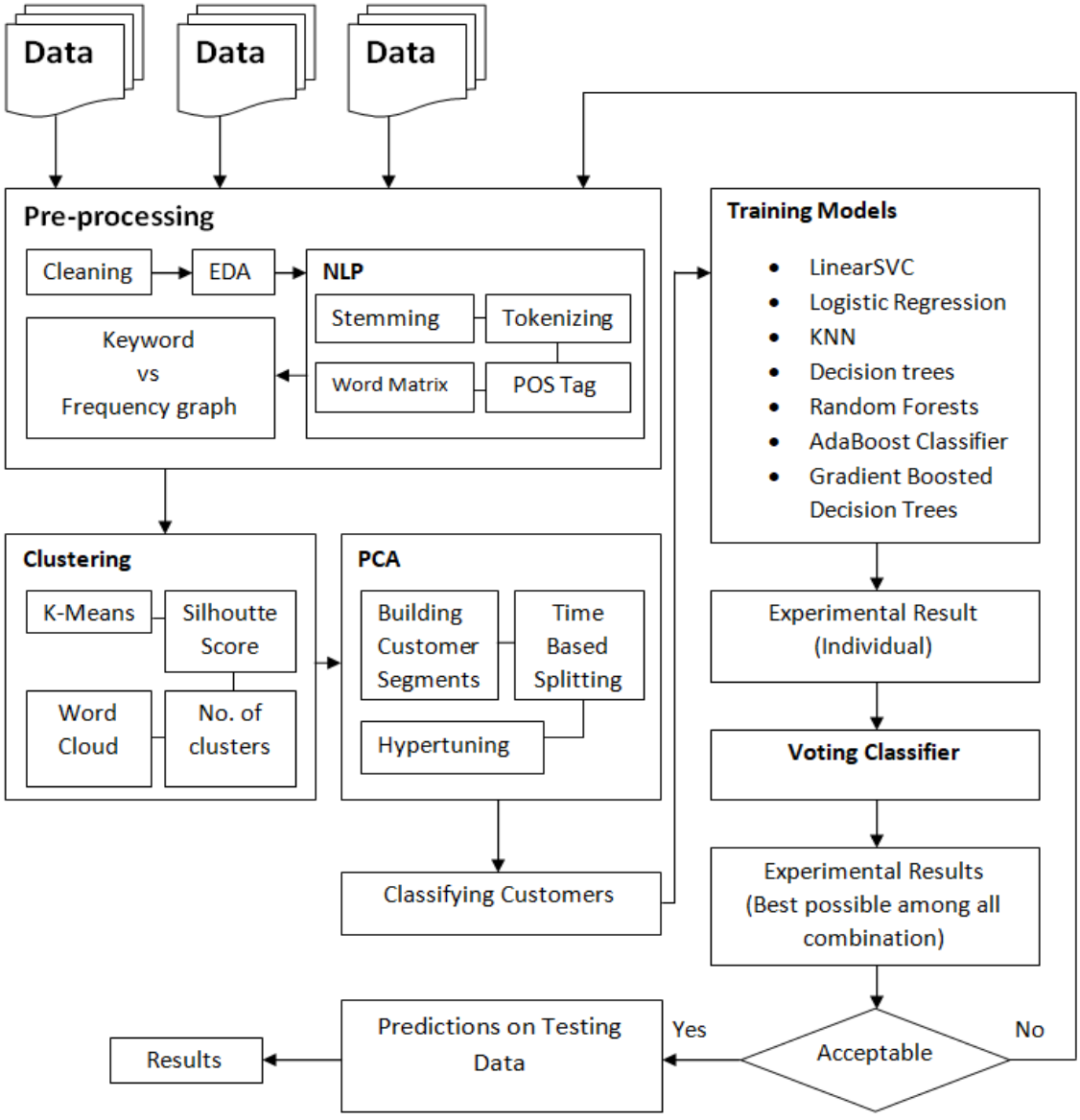

Fig. 3 End-to-End Workflow

Let’s understand each and every component of our Project workflow one-by-one.

3.2 Dataset

3.2.1 Data Collection

The data for our study is collected from UCI Machine Learning repository. This is a transnational data set which contains all the transactions occurring between 01/12/2010 and 09/12/2011 for a UK-based and registered non-store online retail. The company mainly sells unique all-occasion gifts. Many customers of the company are wholesalers.

3.2.2 Attribute Information

S.no	Attribute Name
1.	Invoice_No
2.	Stock_Code
3.	Description
4.	Quantity
5.	Invoice_Date
6.	Unit_Price
7.	Customer_ID
8.	Country

Table 1 – Dataset Attributes

1. **Invoice_No:** Invoice number. Nominal, a 6-digit integral number uniquely assigned to each transaction. If this code starts with letter 'c', it indicates a cancellation.
2. **Stock_Code:** Product (item) code. Nominal, a 5-digit integral number uniquely assigned to each distinct product.
3. **Description:** Product (item) name. Nominal.
4. **Quantity:** The quantities of each product (item) per transaction. Numeric.
5. **Invoice_Date:** Invoice Date and time. Numeric, the day and time when each transaction was generated.
6. **Unit_Price:** Unit price. Numeric, Product price per unit in sterling.
7. **Customer_ID:** Customer number. Nominal, a 5-digit integral number uniquely assigned to each customer.
8. **Country:** Country name. Nominal, the name of the country where each customer resides.

3.3 Preprocessing

3.3.1 Data Cleaning

Data cleaning is one of the important parts of machine learning. It plays a significant part in building a model. Data Cleaning is one of those things that everyone does but no one really talks about. It surely isn't the fanciest part of machine learning and at the same time, there aren't any hidden tricks or secrets to uncover. However, proper data cleaning can make or break your project. Professional data scientists usually spend a very large portion of their time on this step. Because of the belief that, **"Better data beats fancier algorithms"**.

If we have a well-cleaned dataset, we can get desired results even with a very simple algorithm, which can prove very beneficial at times.

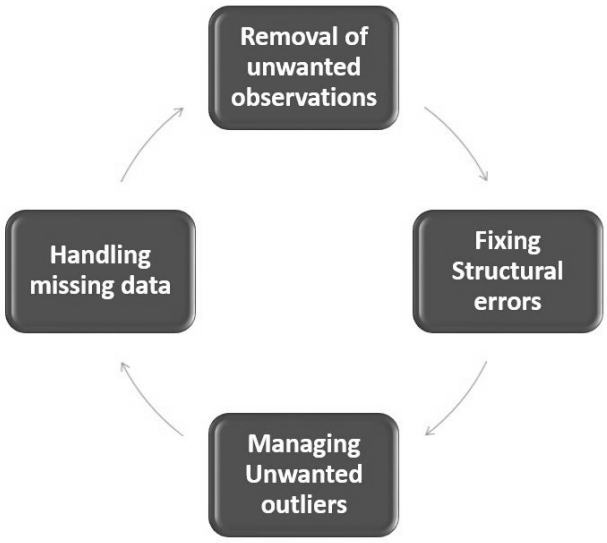

Fig. 4 Steps involved in Data Cleaning

3.3.2 Exploratory Data Analysis

In data mining, Exploratory Data Analysis (EDA) is an approach to analyzing datasets to summarize their main characteristics, often with visual methods. EDA is used for seeing what the data can tell us before the modeling task. It is not easy to look at a column of numbers or a whole spreadsheet and determine important characteristics of the data. It may be tedious, boring, and/or overwhelming to derive insights by looking at plain numbers. Exploratory data analysis techniques have been devised as an aid in this situation.

3.3.2.1 Univariate Analysis

Univariate analysis is the simplest form of data analysis, where the

data being analyzed consists of only one variable.

i) Box Plots

ii) Histogram etc.

3.3.2.2 Multivariate Analysis

Multivariate data analysis refers to any statistical technique used to analyze data that arises from more than one variable.

i) Scatter Plot

ii) Bar Chart etc.

3.4 Natural Language Processing

Natural Language Processing, usually shortened as NLP, is a branch of artificial intelligence that deals with the interaction between computers and humans using the natural language.

The ultimate objective of NLP is to read, decipher, understand, and make sense of the human languages in a manner that is valuable. Most NLP techniques rely on machine learning to derive meaning from human languages.

3.4.1 How does NLP Works?

NLP entails applying algorithms to identify and extract the natural language rules such that the unstructured language data is converted into a form that computers can understand.

When the text has been provided, the computer will utilize algorithms to extract meaning associated with every sentence and collect the essential data from them. Sometimes, the computer may fail to understand the meaning of a sentence well, leading to obscure results. For example, a humorous incident occurred in the 1950s during the translation of some words between the English and the Russian languages [9].

Here is the biblical sentence that required translation:

"The spirit is willing, but the flesh is weak."

Here is the result when the sentence was translated to Russian and back to English:

"The vodka is good, but the meat is rotten."

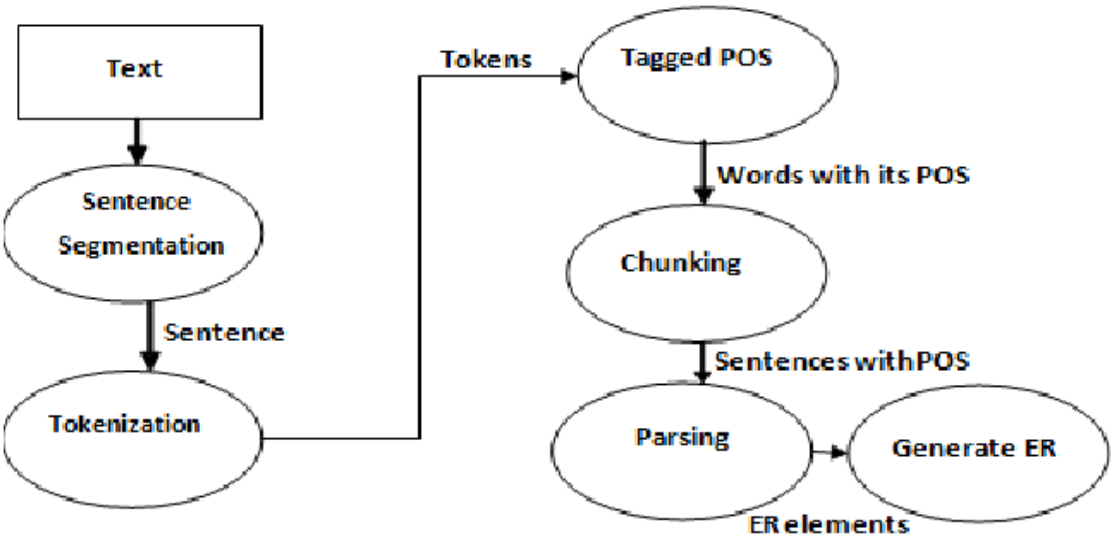

Fig.5 Generating ER from NLP

3.4.2 Techniques used in NLP

Syntactic analysis and semantic analysis are the main techniques used to complete Natural Language Processing tasks. Here is a description on how they can be used.

3.4.2.1 Syntax

Syntax refers to the arrangement of words in a sentence such that they make grammatical sense.

In NLP, syntactic analysis is used to assess how the natural language aligns with the grammatical rules. Computer algorithms are used to apply grammatical rules to a group of words and derive meaning from them [9]. Here are some syntax techniques that we have used:

- **Lemmatization**: It entails reducing the various inflected forms of a word into a single form for easy analysis.

- **Morphological segmentation**: It involves dividing words into individual units called morphemes.

- **Word segmentation**: It involves dividing a large piece of continuous text into distinct units.

- **Part-of-speech tagging**: It involves identifying the part of speech for every word.

- **Parsing**: It involves undertaking grammatical analysis for the provided sentence.

- **Sentence breaking**: It involves placing sentence boundaries on a large piece of text.

- **Stemming**: It involves cutting the inflected words to their root form.

3.4.2.2 Semantics

Semantics refers to the meaning that is conveyed by a text. Semantic analysis is one of the difficult aspects of Natural Language Processing that has not been fully resolved yet [9].

It involves applying computer algorithms to understand the meaning and interpretation of words and how sentences are structured.

Here are some techniques in semantic analysis:

- **Named entity recognition (NER):** It involves determining the parts of a text that can be identified and categorized into preset groups. Examples of such groups include names of people and names of places.
- **Word sense disambiguation:** It involves giving meaning to a word based on the context.
- **Natural language generation**: It involves using databases to derive semantic intentions and convert them into human language.

3.5 Cluster Analysis

In this section, the research study that was conducted during this project is presented providing a basic knowledge of cluster analysis and customer segmentation.

3.5.1 Customer Segmentation

An important marketing strategy that is widely used by businesses is customer segmentation. As previously stated, the point of customer segmentation is to split the user-base into smaller groups that can be targeted with specialized content and offers. The produced customer groups are drawn from user behavior data which gives the business a deeper

understanding of the types of users that exists in the system. The benefit of customer segmentation is twofold. Firstly, a better knowledge about the types of users in a system can lead to better business and marketing strategies. Secondly, a user is likely to use an application more often if he/she always receives relevant content. Another essential point is that if a customer is pleased, he/she is more likely to recommend the application to other people which helps in the expansion of a company [11]. This type of marketing technique is a subset of a company's Business Intelligence explained. To be able to create a set of similar customer groups, an extensive analysis of the available data combined with research and evaluation of clustering algorithms is needed.

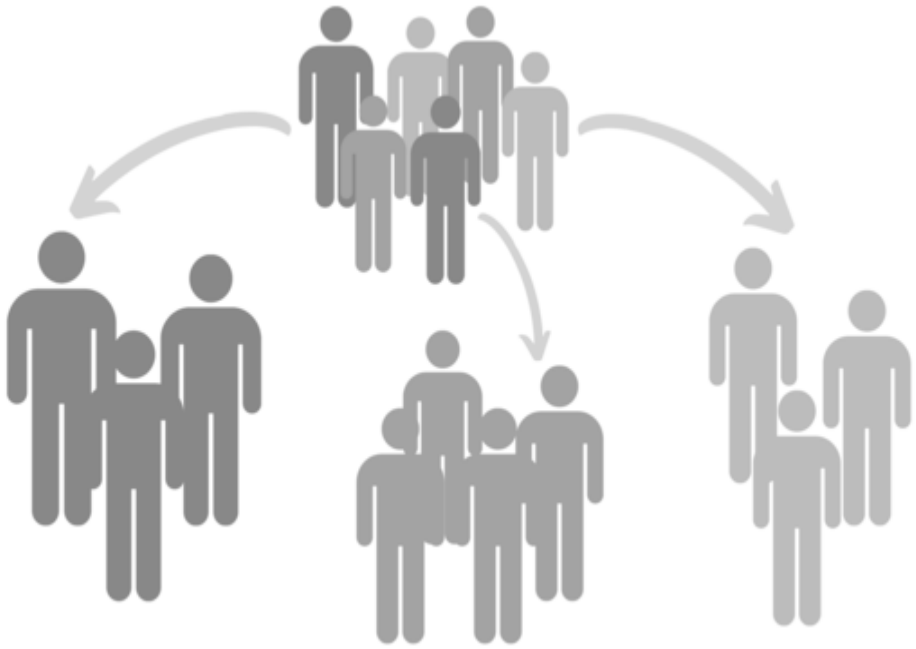

Fig. 6 Customer Segmentation

3.5.2 Clustering Algorithm

Clustering algorithms are used to assign users into groups so that users belonging to the same group are more similar than users in another group. The goal of this division is to find meaningful underlying patterns within the data space. User similarity is determined by a distance measure. This section will introduce the most common similarity measures and clustering algorithms.

3.5.2.1 Similarity Measures

Clustering is highly dependent on defining a relevant similarity or distance measure. The simplest and most common distance measure is Euclidian distance.

$$d(x, y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2} \quad \text{(eq. 2)}$$

where n is the number of features in the data objects x and y, and x_k and y_k are the k^{th} attribute of the feature data objects x and y respectively.

Cosine correlation is widely used in the field of recommender systems, mainly in collaborative filtering. The main idea behind cosine correlation is to compute the cosine value of the angle that two n-dimensional feature vectors form [14]. This is possible using the following equation, where n is the number of features in the data

objects x and y, · indicates the vector dot product and || x || is the norm of vector x:

$$cos(x, y) = \frac{(x \cdot y)}{\| x \| \| y \|} \quad \text{(eq. 3)}$$

The another distance measure that will be covered in this report is Pearson correlation. This distance measure is also widely used in recommender systems. Pearson correlation computes the linear relationship between two feature vectors, in other words two feature vectors are similar if a best fitting straight line is close to all data points in both vectors. It is computed using the following function:

$$Pearson(x, y) = \frac{\Sigma(x, y)}{\sigma x \cdot \sigma y} \quad \text{(eq. 4)}$$

where x and y are two feature vectors, ∑ is the covariance of the data points x and y and is the standard deviation of a feature vector. The result is a value between -1 and 1 where a value close to 1 or -1 means that all values are located on the best fitting line, and values closer to 0 shows that there is little correlation between the given feature vectors.

3.5.2.2 K-means

K-means clustering is widely used in the field of cluster analysis and customer segmentation. K-means is an algorithm designed to group a

set of items into K subgroup or clusters. The algorithm is dependent on a manually set value for K. The K centroids are initialized to random observations in the dataset. K-means is then tasked with iteratively moving these centroids to minimize the cluster variance using two steps

• for each centroid c identifies the subset of items that are closer to c than any other centroid using some similarity measure.

• calculate a new centroid each cluster after every iteration which is equal to the mean vector of all the vectors in the cluster.

This two-step process is repeated until convergence is reached. The standard implementation of K-means uses Euclidian distance measure described in the section above to find the subset of items that corresponds to each cluster. This is done by calculating mean squared error, which in this case is equivalent with the Euclidian distance, of each item's feature vector with the K centroid and choosing the closest result [15]. However, other distance measures can be used instead of Euclidian distance. Aggarwal et al. claim that for high dimensional data, the choice of distance measure used in clustering is vital for its success.

3.5.2.3 Silhouette Score

This is a better measure to decide the number of clusters to be formulated from the data. It is calculated for each instance and the

formula goes like this:

Silhouette Coefficient = (x - y) / max(x, y*) (eq. 5)

where, **y** is the mean intra cluster distance: mean distance to the other instances in the same cluster. **x** depicts mean nearest cluster distance i.e. mean distance to the instances of the next closest cluster. The coefficient varies between -1 and 1. A value close to 1 implies that the instance is close to its cluster is a part of the right cluster. Whereas, a value close to -1 means that the value is assigned to the wrong cluster [16].

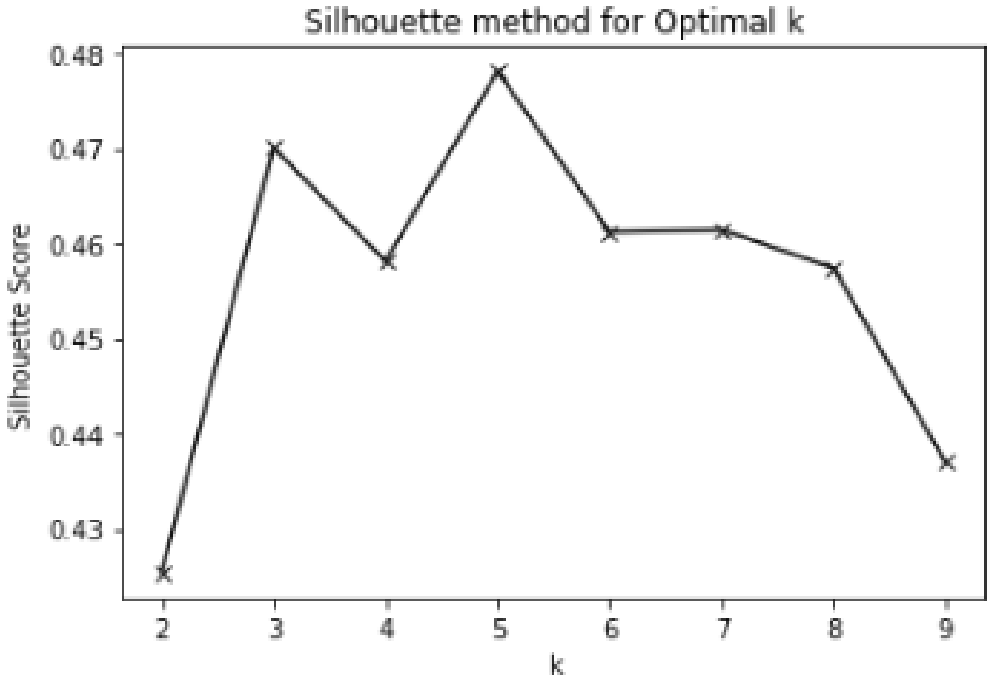

Fig. 7 Silhouette Method

As per this method k=3 was a local optima, whereas k=5 should be chosen for the number of clusters. This method is better as it makes the decision regarding the optimal number of clusters more meaningful and clear. But this metric is computation expensive as the coefficient is calculated for every instance [16]. Therefore, decision regarding the optimal metric to be chosen for the number of cluster decision is to be made according to the needs of the product.

3.6 Dimensionality Reduction

In machine learning classification problems, there are often too many factors on the basis of which the final classification is done. These factors are basically variables called features. The higher the number of features, the harder it gets to visualize the training set and then work on it. Sometimes, most of these features are correlated, and hence redundant. This is where dimensionality reduction algorithms come into play. Dimensionality reduction is the process of reducing the number of random variables under consideration, by obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

3.6.1 Principal Component Analysis

PCA is a widely used method in statistical analysis. The algorithm is often used whenever high dimensional data is involved, such as facial recognition applications, speech/music segmentation and customer segmentation. PCA is

used to find underlying structures (Principal Components) in the data by linearly combining the features with each other. PCA is tasked with finding the best principal components (characteristics that differ the most in the dataset) among all possible linear combinations of the available data. The result is an ordered list of principal components that account for the largest amount of variance in the data. A principal component is a summary of multiple features in the original data combined into one. In order to use PCA further as a dimension reduction algorithm, the least contributing principal components can be discarded. Furthermore, the number of principal components is always less than or equal to the number of original features. PCA is defined as an orthogonal linear transformation of the original data. An orthogonal linear transformation is a simple linear transformation which preserves the inner product of the vectors. In other words, the lengths of the vectors and the angles between the vectors are preserved. Orthogonal transformations are therefore only rotations, reflections or a combination of rotations and reflections of the original feature space. This transformation forms a new coordinate system where the values in each axis are based on the computed principal components.

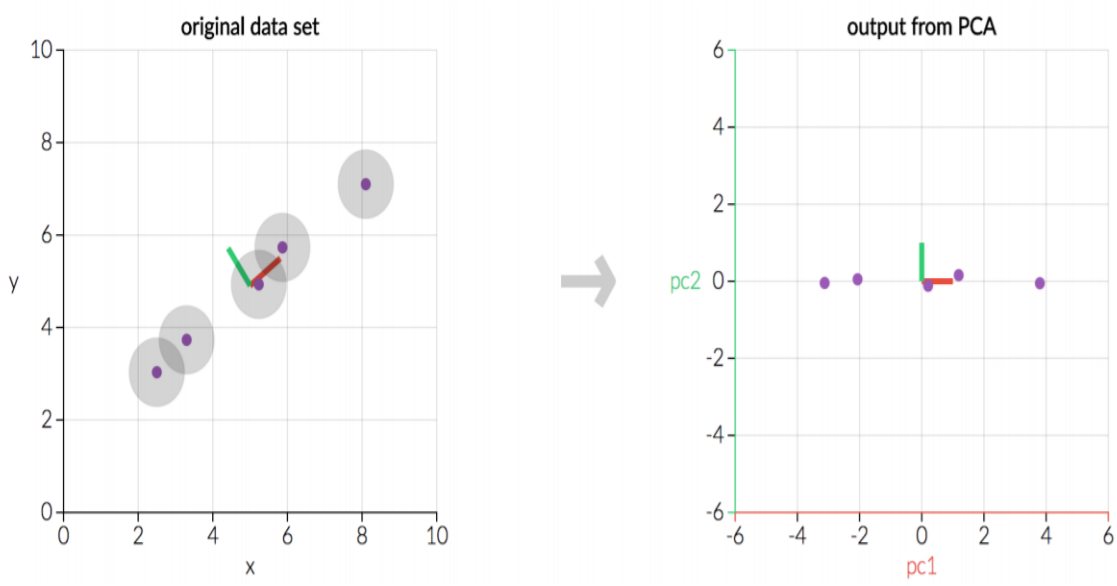

Fig. 8 Visual example of Dimensionality Induction

Given the data in the leftmost plot in the figure above, PCA is tasked with finding correlations in the data and create a new coordinate system that explains the data with as little information as possible. In this simple two dimensional example, a correlation between x and y can be seen. More precisely, points with low values for x also have low values for y and points with high values for x have high values for y. In this plot three groups of similar points can be identified with Euclidian similarity. The first group, consisting of two points, is located between x = 2 and x = 4, the second group also consists of two points located between x = 4 and x = 6 and finally the last point at x = 8 which is well separated from the other groups. Given this correlation, PCA rotates the plane, creating two new axis Principle Component 1 and Principle Component 2. In the rightmost plot above, the

same conclusions about the group of points can be drawn only by using PC1 meaning that PCA has successfully reduced the dimensionality of the data from two dimensions to one. For this simple example PCA is not really useful since a two dimensional dataset can be plotted and analyzed rather easily, however PCA provides the possible to reduce the dimensionality of high dimensional data which is almost always a necessity in data mining applications.

3.6.2 Mathematical Details

Given a data matrix X which n rows and m columns, where each row represents an observation in the dataset and each column represents a particular kind of observation. In the case of customer segmentation in an E-marketplace, a row is a user and a column is an item, a brand or any similar type of data available in the marketplace [13]. The transformation explained above is defined by a set of m-dimensional weight vectors $w_k = (w_1,..., w_m)_k$. Each row vector xi of X is then mapped to a new vector of PCA scores $t_i = (t_1,..., t_m)_i$, such that:

$t_{ki} = x_i \cdot w_k$, where i = 1, ..., n and k = 1, ..., m. The resulting set of the t vectors are ordered such that the individual variables of t in the whole dataset inherits the maximum amount of variance of x. Without diving into the mathematical formulae, the optimization of the first weight vector can be recognized with a Rayleigh quotient and thus the optimization problem of PCA has its root in this quotient [17].

Mathematically speaking, PCA is done in five different steps. These steps are briefly discussed below:

1. **Standardizing:** Standardizing is important in PCA since the variance in the data is maximized to produce linear separability. This means that the data needs to be in the same scale. Standardized scores are derived by subtracting the sample mean from an individual score and then dividing the difference by the sample standard deviation.

2. **Calculate the covariance matrix:** Calculate the covariance matrix for the standardized input matrix. The covariance between two variables or features can be seen as a description of the similarities between the variance of the variables. In other words, how do the two variables relate to each other.

3. **Calculate the eigenvectors and eigenvalues of the covariance matrix:** As previously stated, the principal components are used to create the axes of new coordinate system. In order to rotate the data into the new coordinate system, all data points are multiplied with the eigenvectors which indicate the direction of the new axes deduced from the principal components [17]. The eigenvalues are used to determine the magnitude of the new feature space. The eigenvectors and eigenvalues can be deduced from the covariance matrix calculated in the step above.

4. **Choosing principal components and deriving the new features**: Finally the new dataset is derived by choosing the desired principal components. The amount of principal components chosen depends on the dimension of the original data and the percentage of variance explained by each of the selected components, i.e. only contributing components should be chosen.

   As stated this is a fairly brief overview of the mathematics behind PCA since this is not within the scope of this project. However, In Joliffe provide a more in depth explanation of these steps and the mathematical formulas behind them. As any other algorithm, PCA has some limitations and some assumptions that need to hold in order to get meaningful principal components. Mainly the issue is that the original data set needs to have some underlying structure that can be linearly separated since PCA finds “hidden” linear correlations in high dimensional data. Also, during the training phase, the training data needs to be chosen such that all underlying patterns in the data are caught. Therefore choosing a good training set is the most important prerequisite of PCA.

3.7 Voting Classifier

Voting is considered to be one of the efficient and simplest methods of combining the predicted results of multiple algorithms of machine learning. The main idea behind this is to create parallel dedicated models followed by finding the model accuracy for each of the individual models followed by creating a single robust model which is trained by those individual models

and it then predicts final output based on merged majority of voting for each and every output class [6].

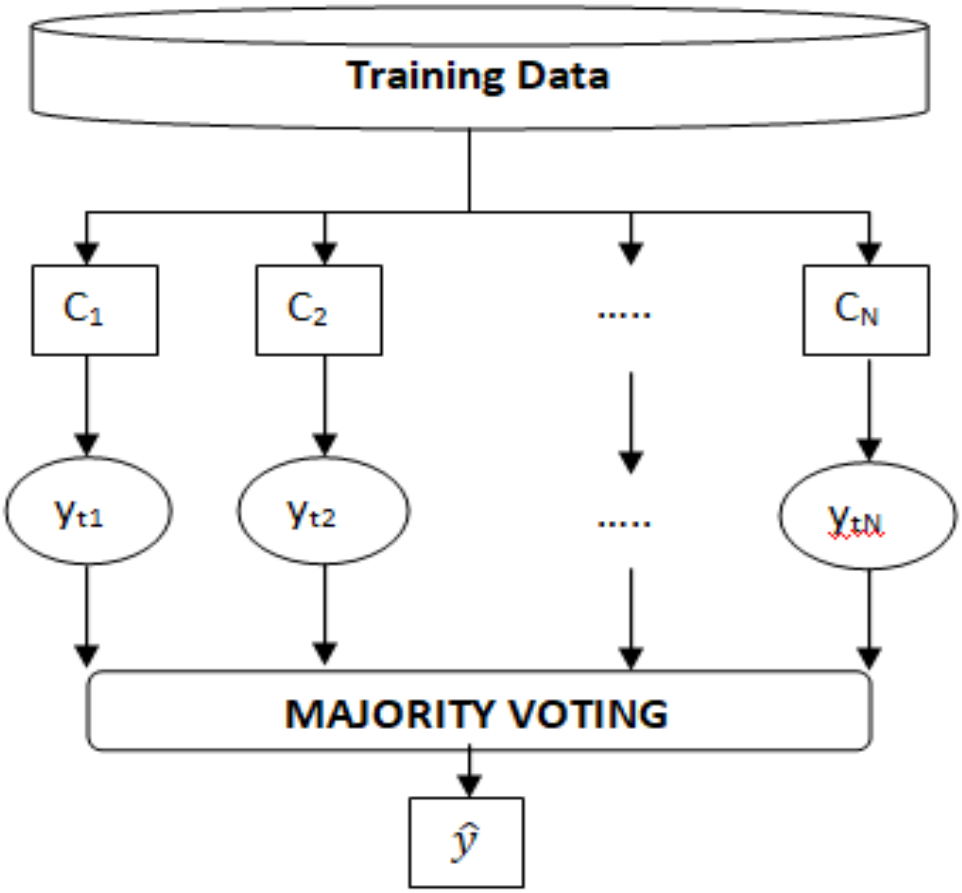

Fig. 9 Majority Voting Classifier Model

Voting classifier has basically two types:

Hard Voting **–** Here, the final predicted class $\hat{y}$ is the class having majority of votes being highest in number $N_c(y_t)$. It means the class whose probability of being finally predicted by each classifier is highest.

$$\hat{y} = \boldsymbol{argmax}(\boldsymbol{N_c}(\boldsymbol{y_t^1}),\ \boldsymbol{N_c}(\boldsymbol{y_t^2}), \ldots\ldots\ldots, \boldsymbol{N_c}(\boldsymbol{y_t^N})) \qquad \text{(eq. 6)}$$

Soft Voting – Here, the final predicted class $\hat{y}$ is based on the overall average of the probability (P_N) been given to that particular class.

$$\hat{y} = \boldsymbol{argmax}\left(\frac{\mathbf{1}}{\boldsymbol{N_{classifiers}}}\right)\sum_{\boldsymbol{classifiers}}(\boldsymbol{P_1}, \boldsymbol{P_2}, \ldots., \boldsymbol{P_N}) \qquad \text{(eq. 7)}$$

We have used Hard voting classifier in our study.

Chapter 4

IMPLEMENTATION

4.1 Choice of Programming Language and Environment

- Language Used – Python (Version 3.7)
- Environment – Jupyter Notebook
- Operating System – Windows
- RAM – 8 GB
- Processor – Intel Core i5 (7^{th} Generation)

4.2 Libraries Used

- Numpy
- Pandas
- Matplotlib
- Seaborn
- Scipy
- Itertools
- NLTK
- Wordcloud
- Sklearn
- Random
- Datetime

4.3 Raw Dataset Snippet

```
data = pd.read_excel('Online Retail.xlsx', dtype={'StockCode':str})
data.head(3)
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom

Fig. 10 Importing Dataset

```
data.shape
```

```
(541909, 8)
```

4.4 Data Preprocessing

```
# Checking for null values.
info = pd.DataFrame(data=data.isnull().sum()).T.rename(index={0:'Null values'})
info = info.append(pd.DataFrame(data=data.isnull().sum()/data.shape[0] * 100).T.rename(ind
ex={0:'% Null values'}))
info
```

```
# Removing null values
data.dropna(axis=0, subset = ['CustomerID'], inplace=True)
info = pd.DataFrame(data=data.isnull().sum()).T.rename(index={0:'Null values'})
info = info.append(pd.DataFrame(data=data.isnull().sum()/data.shape[0] * 100).T.rename(ind
ex={0:'% Null values'}))
info
```

```
# Removing duplicate entries :
data.drop_duplicates(inplace=True)
data.duplicated().sum()
```

4.5 Exploratory Data Analysis

```
plt.figure(figsize=(14,6))
plt.plot(data.groupby(['Country']).groups.keys(), data.groupby(['Country'])['CustomerID'].
count())
plt.xticks(rotation = 90, fontsize = 14)
plt.title("Number of transanctions done for each country")
plt.ylabel("No. of trans.")
plt.xlabel("Country")
plt.show()
```

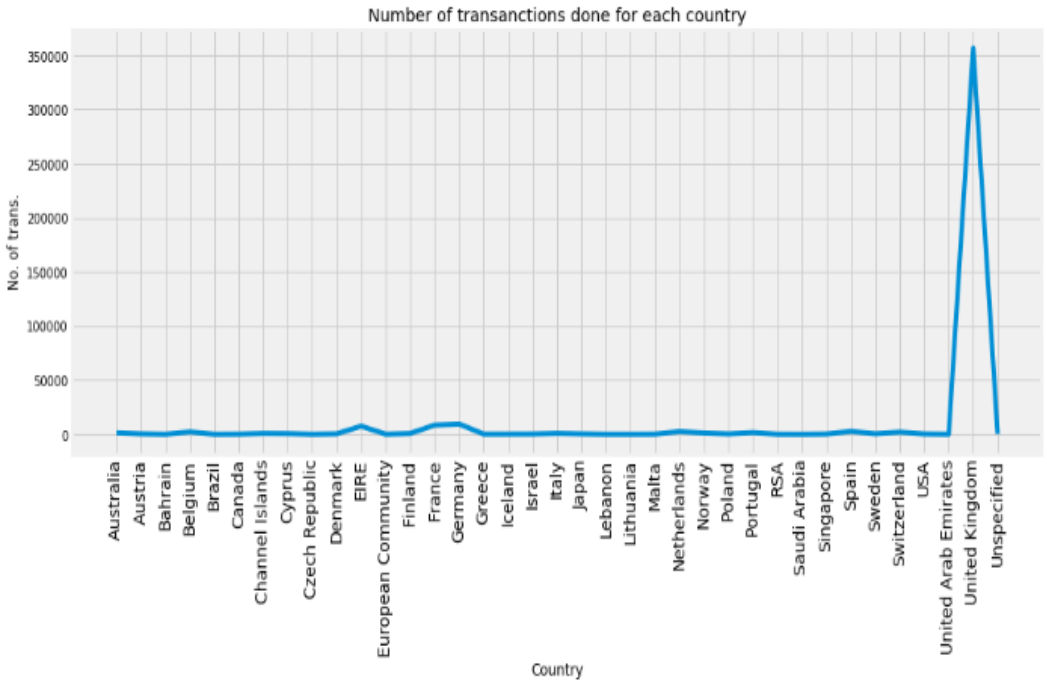

Fig. 11 EDA – I

```
# StockCode Feature ->
# We will see how many different products were sold in the year data was collected.
print(len(data['StockCode'].value_counts()))
```

```
3684
```

```
# Transanction feature
# We will see how many different transanctions were done.
print(len(data['InvoiceNo'].value_counts()))
```

```
22190
```

```
# Transanction feature
# We will see how many different Customers are there.
print(len(data['CustomerID'].value_counts()))
```

```
4372
```

```
pd.DataFrame({'products':len(data['StockCode'].value_counts()),
              'transanctions':len(data['InvoiceNo'].value_counts()),
              'Customers':len(data['CustomerID'].value_counts())},
             index = ['Quantity'])
```

	Customers	products	transanctions
Quantity	4372	3684	22190

Checking the number of items bought in a single transanctions :

```
df = data.groupby(['CustomerID', 'InvoiceNo'], as_index=False)['InvoiceDate'].count()
df = df.rename(columns = {'InvoiceDate':'Number of products'})
df[:10].sort_values('CustomerID')
```

	CustomerID	InvoiceNo	Number of products
0	12346.0	541431	1
1	12346.0	C541433	1
2	12347.0	537626	31
3	12347.0	542237	29
4	12347.0	549222	24
5	12347.0	556201	18
6	12347.0	562032	22
7	12347.0	573511	47
8	12347.0	581180	11
9	12348.0	539318	17

Counting number of cancelled transanctions :

```
df['orders cancelled'] = df['InvoiceNo'].apply(lambda x: int('C' in str(x)))
df.head()
```

	CustomerID	InvoiceNo	Number of products	orders cancelled
0	12346.0	541431	1	0
1	12346.0	C541433	1	1
2	12347.0	537626	31	0
3	12347.0	542237	29	0
4	12347.0	549222	24	0

Removing cancelled orders :

```
df_cleaned = data.copy(deep=True)
df_cleaned['QuatityCancelled'] = 0
entry_to_remove = []; doubtfull_entry = []

for index, col in data.iterrows():
    if(col['Quantity'] > 0)or(col['Description']=='Discount'):continue
    df_test = data[(data['CustomerID']==col['CustomerID'])&(data['StockCode']==col['StockC
ode'])&
                   (data['InvoiceDate']<col['InvoiceDate'])&(data['Quantity']>0)].copy()

    # Order cancelled without counterpart, these are doubtful as they maybe errors or mayb
e orders were placed before data given
    if(df_test.shape[0] == 0):
        doubtfull_entry.append(index)

    # Cancellation with single counterpart
    elif(df_test.shape[0] == 1):
        index_order = df_test.index[0]
        df_cleaned.loc[index_order, 'QuantityCancelled'] = -col['Quantity']
        entry_to_remove.append(index)

    # Various counterpart exists for orders
    elif(df_test.shape[0] > 1):
        df_test.sort_index(axis = 0, ascending=False, inplace=True)
        for ind, val in df_test.iterrows():
            if val['Quantity'] < -col['Quantity']:continue
            df_cleaned.loc[ind, 'QuantityCancelled'] = -col['Quantity']
            entry_to_remove.append(index)
            break
```

```
print("Entry to remove {}".format(len(entry_to_remove)))
print("Doubtfull Entry {}".format(len(doubtfull_entry)))
```

```
Entry to remove 7521
Doubtfull Entry 1226
```

```
# Deleting these entries :
df_cleaned.drop(entry_to_remove, axis=0, inplace=True)
df_cleaned.drop(doubtfull_entry, axis=0, inplace=True)
```

Plotting the purchases made :

```
price_range = [0, 50, 100, 200, 500, 1000, 5000, 50000]
count_price = []
for i,price in enumerate(price_range):
    if i==0:continue
    val = basket_price[(basket_price['Basket Price'] < price)&
                       (basket_price['Basket Price'] > price_range[i-1])]['Basket Price'].
count()
    count_price.append(val)

plt.rc('font', weight='bold')
f, ax = plt.subplots(figsize=(11, 6))
colors = ['yellowgreen', 'gold', 'wheat', 'c', 'violet', 'royalblue', 'firebrick']
labels = ["{}<.<{}".format(price_range[i-1], s) for i,s in enumerate(price_range) if i !=
0]
sizes = count_price
explode = [0.0 if sizes[i] < 100 else 0.0 for i in range(len(sizes))]
ax.pie(sizes, explode = explode, labels = labels, colors = colors,
       autopct = lambda x:'{:1.0f}%'.format(x) if x > 1 else '',
       shadow = False, startangle = 0)
ax.axis('equal')
f.text(0.5, 1.01, "Distribution of order amounts", ha = 'center', fontsize = 18)
plt.show()
```

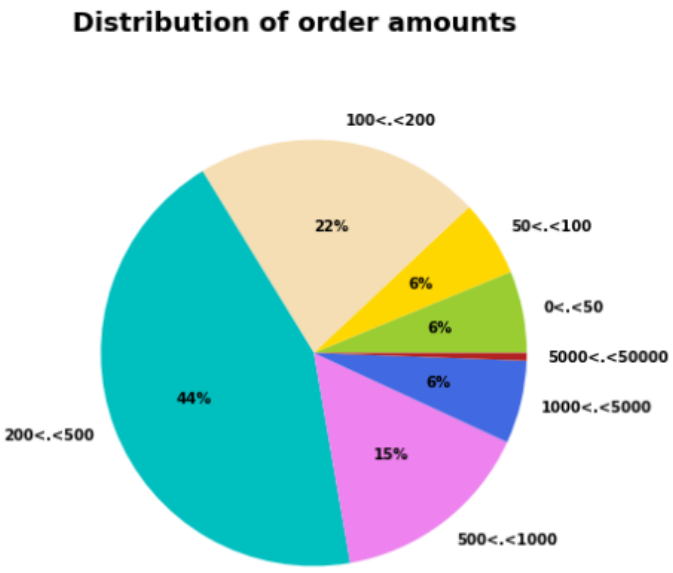

Fig. 12 EDA – II

```
# Plotting keywords vs frequency graph :
list_products = []
for k, v in count_keywords.items():
    word = keywords_select[k]
    list_products.append([word, v])
```

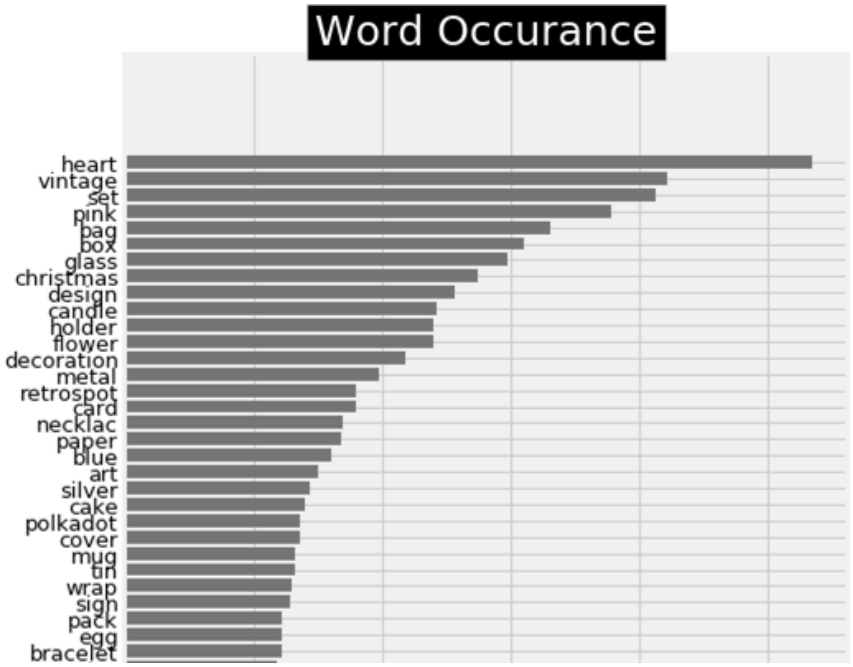

Fig. 13 EDA – III

4.6 Clustering

```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
```

```
matrix = X.as_matrix()
```

```
# Using optimal number of clusters using hyperparameter tuning:
for n_clusters in range(3, 10):
    kmeans = KMeans(init='k-means++', n_clusters = n_clusters, n_init = 30)
    kmeans.fit(matrix)
    clusters = kmeans.predict(matrix)
    sil_avg = silhouette_score(matrix, clusters)
    print("For n_clusters : ", n_clusters, "The average silhouette_score is : ", sil_avg)
```

```
For n_clusters :  3 The average silhouette_score is :  0.10158702596
For n_clusters :  4 The average silhouette_score is :  0.126800458839
For n_clusters :  5 The average silhouette_score is :  0.126032697286
For n_clusters :  6 The average silhouette_score is :  0.151785830934
For n_clusters :  7 The average silhouette_score is :  0.146845533401
For n_clusters :  8 The average silhouette_score is :  0.152603858325
For n_clusters :  9 The average silhouette_score is :  0.160329064505
```

```
# Choosing number of clusters as 5:
# Trying Improving the silhouette_score :
n_clusters = 5
sil_avg = -1
while sil_avg < 0.145:
    kmeans = KMeans(init = 'k-means++', n_clusters = n_clusters, n_init = 30)
    kmeans.fit(matrix)
    clusters = kmeans.predict(matrix)
    sil_avg = silhouette_score(matrix, clusters)
    print("For n_clusters : ", n_clusters, "The average silhouette_score is : ", sil_avg)
```

```
For n_clusters :  5 The average silhouette_score is :  0.147087004595
```

```
# Printing number of elements in each cluster :
pd.Series(clusters).value_counts()
```

```
# Plotting the intra cluster silhouette distances.
from sklearn.metrics import silhouette_samples
sample_silhouette_values = silhouette_samples(matrix, clusters)
graph_component_silhouette(n_clusters, [-0.07, 0.33], len(X), sample_silhouette_values, cl
usters)
```

Wordcloud

```
# Code for printing word cloud.
from random import randint
import random
def random_color_func(word=None, font_size=None, position=None,orientation=None, font_path
=None, random_state=None):
    h = int(360.0 * tone / 255.0)
    s = int(100.0 * 255.0 / 255.0)
    l = int(100.0 * float(random_state.randint(70, 120)) / 255.0)
    return "hsl({}, {}%, {}%)".format(h, s, l)
```

4.7 Dimensionality Reduction – PCA

```
pca = PCA()
pca.fit(matrix)
pca_samples = pca.transform(matrix)
```

```
# Checking the amount of variance explained :
fig, ax = plt.subplots(figsize=(14, 5))
sns.set(font_scale=1)
plt.step(range(matrix.shape[1]), pca.explained_variance_ratio_.cumsum(), where = 'mid', la
bel = 'Cummulative Variance Explained')
sns.barplot(np.arange(1, matrix.shape[1] + 1), pca.explained_variance_ratio_, alpha = 0.5,
color = 'g',
            label = 'Individual Variance Explained')
plt.xlim(0, 100)
plt.xticks(rotation = 45, fontsize = 14)
ax.set_xticklabels([s if int(s.get_text())%2 == 0 else '' for s in ax.get_xticklabels()])

plt.ylabel("Explained Variance", fontsize = 14)
plt.xlabel("Principal Components", fontsize = 14)
plt.legend(loc = 'upper left', fontsize = 13)
plt.show()
```

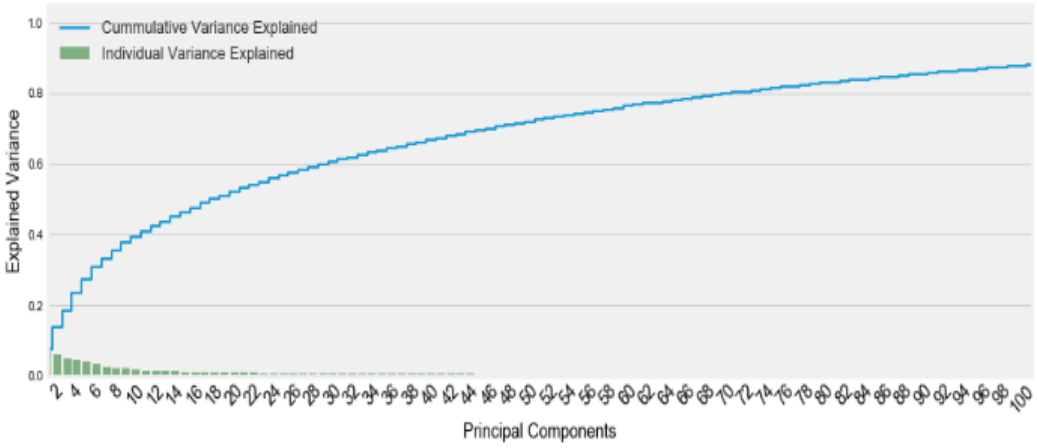

Fig. 14 Scree - Plot

Generating Customer Segments/Categories

```
corresp = dict()
for key, val in zip(liste_produits, clusters):
    corresp[key] = val

df_cleaned['categ_product'] = df_cleaned.loc[:, 'Description'].map(corresp)
df_cleaned[['InvoiceNo', 'Description', 'categ_product']][:10]
```

	InvoiceNo	Description	categ_product
0	536365	WHITE HANGING HEART T-LIGHT HOLDER	3
1	536365	WHITE METAL LANTERN	1
2	536365	CREAM CUPID HEARTS COAT HANGER	1
3	536365	KNITTED UNION FLAG HOT WATER BOTTLE	1
4	536365	RED WOOLLY HOTTIE WHITE HEART.	1
5	536365	SET 7 BABUSHKA NESTING BOXES	4
6	536365	GLASS STAR FROSTED T-LIGHT HOLDER	1
7	536366	HAND WARMER UNION JACK	3
8	536366	HAND WARMER RED POLKA DOT	2
9	536367	ASSORTED COLOUR BIRD ORNAMENT	2

```
# Using optimal number of clusters using hyperparameter tuning:
for n_clusters in range(3, 21):
    kmeans = KMeans(init='k-means++', n_clusters = n_clusters, n_init = 30)
    kmeans.fit(scaled_matrix)
    clusters = kmeans.predict(scaled_matrix)
    sil_avg = silhouette_score(scaled_matrix, clusters)
    print("For n_clusters : ", n_clusters, "The average silhouette_score is : ", sil_avg)
```

```
For n_clusters :  3 The average silhouette_score is :  0.162844157296
For n_clusters :  4 The average silhouette_score is :  0.172789757934
For n_clusters :  5 The average silhouette_score is :  0.178618183922
For n_clusters :  6 The average silhouette_score is :  0.178340736992
For n_clusters :  7 The average silhouette_score is :  0.187064683525
For n_clusters :  8 The average silhouette_score is :  0.196915969104
For n_clusters :  9 The average silhouette_score is :  0.199877098815
For n_clusters :  10 The average silhouette_score is :  0.208299939338
For n_clusters :  11 The average silhouette_score is :  0.211951519795
For n_clusters :  12 The average silhouette_score is :  0.185416904372
For n_clusters :  13 The average silhouette_score is :  0.190849210173
For n_clusters :  14 The average silhouette_score is :  0.189508603103
For n_clusters :  15 The average silhouette_score is :  0.186869105246
For n_clusters :  16 The average silhouette_score is :  0.192349473152
For n_clusters :  17 The average silhouette_score is :  0.182436572554
For n_clusters :  18 The average silhouette_score is :  0.18541150664
For n_clusters :  19 The average silhouette_score is :  0.183150761094
For n_clusters :  20 The average silhouette_score is :  0.185291371346
```

```
# Choosing number of clusters as 10:
# Trying Improving the silhouette_score :
n_clusters = 10
sil_avg = -1
while sil_avg < 0.208:
    kmeans = KMeans(init = 'k-means++', n_clusters = n_clusters, n_init = 30)
    kmeans.fit(scaled_matrix)
    clusters = kmeans.predict(scaled_matrix)
    sil_avg = silhouette_score(scaled_matrix, clusters)
    print("For n_clusters : ", n_clusters, "The average silhouette_score is : ", sil_avg)
```

```
For n_clusters :  10 The average silhouette_score is :  0.207478318028
For n_clusters :  10 The average silhouette_score is :  0.206803335274
For n_clusters :  10 The average silhouette_score is :  0.208112226061
```

```
# Reorganizing the content of the dataframe.
liste_index = []
for i in range(5):
    column = 'categ_{}'.format(i)
    liste_index.append(merged_df[merged_df[column] > 45].index.values[0])

liste_index_reordered = liste_index
liste_index_reordered += [s for s in merged_df.index if s not in liste_index]

merged_df = merged_df.reindex(index = liste_index_reordered)
merged_df = merged_df.reset_index(drop = False)
merged_df.head()
```

	cluster	count	min	max	mean	sum	categ_0	categ_1	categ_2
0	8.0	2.158798	194.749227	318.017811	247.880760	575.944936	57.381072	8.076618	18.095180
1	4.0	2.444211	210.756611	374.099432	274.922690	784.221158	5.399340	58.032710	11.574298
2	0.0	2.438017	219.674752	336.033742	274.490598	701.306552	13.135762	11.451039	56.475878
3	7.0	3.006557	208.386590	412.569574	300.998205	1046.705443	7.090669	9.696536	15.241867
4	2.0	2.609418	193.528812	320.373573	249.371102	702.617537	5.295918	19.501249	11.796693

```
selected_customers.to_csv("selected_customers.csv")
```

```
merged_df.to_csv("merged_df.csv")
```

4.8 Classifying the Customers

Defining Helper Functions :

```
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
class Class_Fit(object):
    def __init__(self, clf, params = None):
        if params:
            self.clf = clf(**params)
        else:
            self.clf = clf()

    def train(self, x_train, y_train):
        self.clf.fit(x_train, y_train)

    def predict(self, x):
        return self.clf.predict(x)

    def grid_search(self, parameters, Kfold):
        self.grid = GridSearchCV(estimator = self.clf, param_grid = parameters, cv = Kfol
d)

    def grid_fit(self, X, Y):
        self.grid.fit(X, Y)

    def grid_predict(self, X, Y):
        self.predictions = self.grid.predict(X)
        print("Precision: {:.2f} %".format(100 * accuracy_score(Y, self.predictions)))
```

Train, Test Splitting :

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, train_size = 0.8)
```

Training Models :

```
from sklearn.svm import LinearSVC
```

```
svc = Class_Fit(clf=LinearSVC)
svc.grid_search(parameters = [{'C':np.logspace(-2,2,10)}], Kfold = 5)
```

```
svc.grid_fit(X=X_train, Y=Y_train)
```

```
svc.grid_predict(X_test, Y_test)
```

```
class_names = [i for i in range(1,11)]
cnf = confusion_matrix(Y_test, svc.predictions)
np.set_printoptions(precision=2)
plt.figure(figsize=(8,8))
plot_confusion_matrix(cnf, class_names)
```

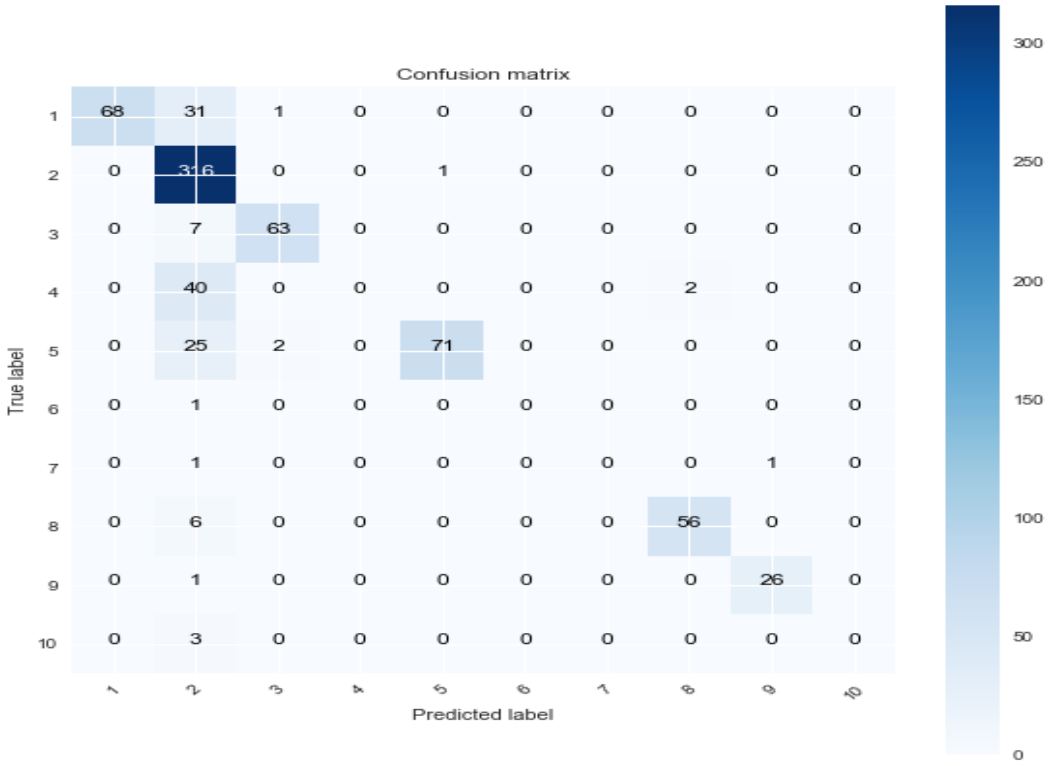

Fig. 15 Confusion Matrix

```
g = plot_learning_curve(svc.grid.best_estimator_, "SVC Learning Curve", X_train, Y_train,
ylim=[1.01, 0.6], cv = 5,
                        train_sizes = [0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9,
1])
```

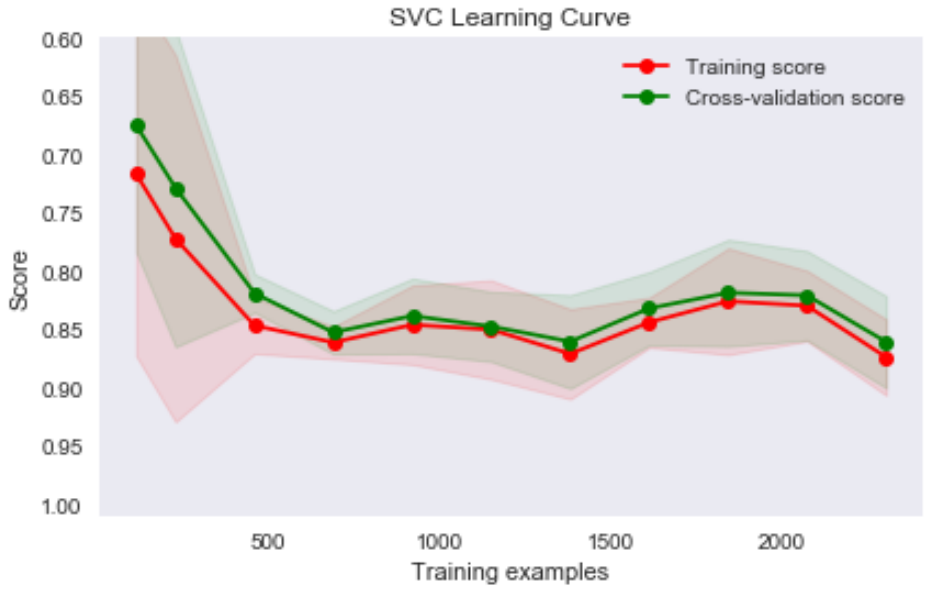

Fig. 16 Learning Curve

Similarly, we have revised this process for different algorithms like – ***Logistic Regression, K-Nearest neighbours, Decision Trees, Random Forests, AdaBoost Classifier, XGBoost*** and tried to measure their performance in terms of Precision Score. Finally, we built a Voting Classifier with all possible combinations of above different algorithms and finally reached to best possible combination.

Logistics Regression :

```
from sklearn.linear_model import LogisticRegression
```

```
lr = Class_Fit(clf = LogisticRegression)
lr.grid_search(parameters = [{'C':np.logspace(-1,2,10)}], Kfold = 5)
lr.grid_fit(X_train, Y_train)
lr.grid_predict(X_test, Y_test)
```

K-Nearest Neighbours :

```
from sklearn.neighbors import KNeighborsClassifier
```

```
knn = Class_Fit(clf = KNeighborsClassifier)
knn.grid_search(parameters = [{'n_neighbors':np.arange(1,50,1)}], Kfold = 5)
knn.grid_fit(X_train, Y_train)
knn.grid_predict(X_test, Y_test)
```

Decision Trees :

```
from sklearn.tree import DecisionTreeClassifier
```

```
tr = Class_Fit(clf = DecisionTreeClassifier)
tr.grid_search(parameters = [{'criterion':['entropy', 'gini'], 'max_features':['sqrt', 'lo
g2']}], Kfold = 5)
tr.grid_fit(X_train, Y_train)
tr.grid_predict(X_test, Y_test)
```

Random Forests:

```
from sklearn.ensemble import RandomForestClassifier
```

```
rf = Class_Fit(clf = RandomForestClassifier)
rf.grid_search(parameters = [{'criterion':['entropy', 'gini'],
                              'max_features':['sqrt', 'log2'], 'n_estimators':[20, 40, 60,
80, 100]}], Kfold = 5)
rf.grid_fit(X_train, Y_train)
rf.grid_predict(X_test, Y_test)
```

AdaBoost Classifier:

```
from sklearn.ensemble import AdaBoostClassifier
```

```
ada = Class_Fit(clf = AdaBoostClassifier)
ada.grid_search(parameters = [{'n_estimators':[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]}],
Kfold = 5)
ada.grid_fit(X_train, Y_train)
ada.grid_predict(X_test, Y_test)
```

Gradient Boosted Decision Trees :

```
import xgboost
```

```
gbdt = Class_Fit(clf = xgboost.XGBClassifier)
gbdt.grid_search(parameters = [{'n_estimators':[10, 20, 30, 40, 50, 60, 70, 80, 90, 10
0]}], Kfold = 5)
gbdt.grid_fit(X_train, Y_train)
gbdt.grid_predict(X_test, Y_test)
```

Voting Classifier :

```
rf_best = RandomForestClassifier(**rf.grid.best_params_)
gbdt_best = xgboost.XGBClassifier(**gbdt.grid.best_params_)
svc_best = LinearSVC(**svc.grid.best_params_)
tr_best = DecisionTreeClassifier(**tr.grid.best_params_)
knn_best = KNeighborsClassifier(**knn.grid.best_params_)
lr_best = LogisticRegression(**lr.grid.best_params_)
```

```
from sklearn.ensemble import VotingClassifier
```

```
votingC = VotingClassifier(estimators=[('rf', rf_best), ('gb', gbdt_best), ('knn', knn_bes
t), ('lr', lr_best)])
```

```
votingC = votingC.fit(X_train, Y_train)
```

```
predictions = votingC.predict(X_test)
```

Testing the model :

```
basket_price = set_test.copy(deep=True)
```

```
transanctions_per_user = basket_price.groupby(by=['CustomerID'])['Basket Price'].agg(['cou
nt', 'min', 'max', 'mean', 'sum'])

for i in range(5):
    col = 'categ_{}'.format(i)
    transanctions_per_user.loc[:, col] = basket_price.groupby(by=['CustomerID'])[col].sum
() / transanctions_per_user['sum'] * 100

transanctions_per_user.reset_index(drop = False, inplace = True)
basket_price.groupby(by=['CustomerID'])['categ_0'].sum()

transanctions_per_user['count'] = 5 * transanctions_per_user['count']
transanctions_per_user['sum'] = transanctions_per_user['count'] * transanctions_per_user
['mean']

transanctions_per_user.sort_values('CustomerID', ascending = True)[:5]
```

```
list_cols = ['count', 'min', 'max', 'mean', 'categ_0', 'categ_1', 'categ_2', 'categ_3', 'c
ateg_4']
matrix_test = transanctions_per_user[list_cols].as_matrix()
scaled_test_matrix = scaler.transform(matrix_test)
```

```
Y = kmeans.predict(scaled_test_matrix)
columns = ['mean', 'categ_0', 'categ_1', 'categ_2', 'categ_3', 'categ_4' ]
X = transanctions_per_user[columns]
predictions = votingC.predict(X)
```

Chapter 5

RESULTS

After performing K-Means clustering on our data and getting the Silhouette score of different number of clusters, we optimized the Silhouette score to 0.147408 with optimal number of clusters as 5 whose distributions are as follows.

Cluster No.	Population
0	606
1	1009
2	626
3	964
4	673

Table 2 No. of Customers in Different Clusters

And, the Intra Cluster Silhouette distances are shown in Figure below.

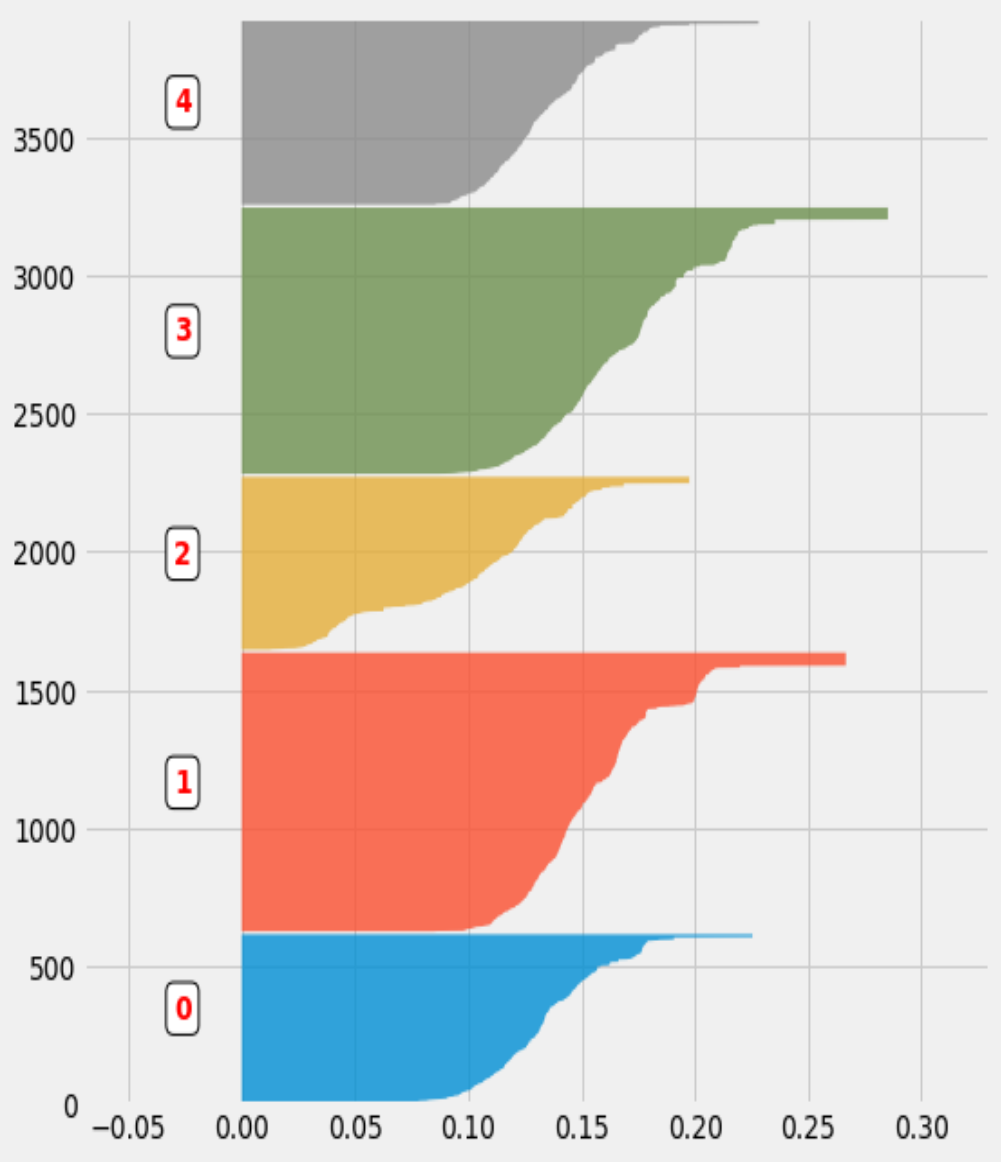

Fig. 17 Intra-Cluster Silhouette Distances

NLP techniques have helped us to identify category of customers more reliably with the help of Wordcloud.

Fig. 18 WorldCloud

Analysis –

1. Cluster number one contains almost all the items that come in the category of gifts and decorations.

2. Cluster number two consists of luxury items.

5.1 Classification Results

Results of both individual classifiers as well as our Voting Classifier model are compared after performing Cross validation in terms of Precision. The performances of all individual classifiers as well as our own built Voting classifiers based on Majority Voting are shown in Fig. 19 below.

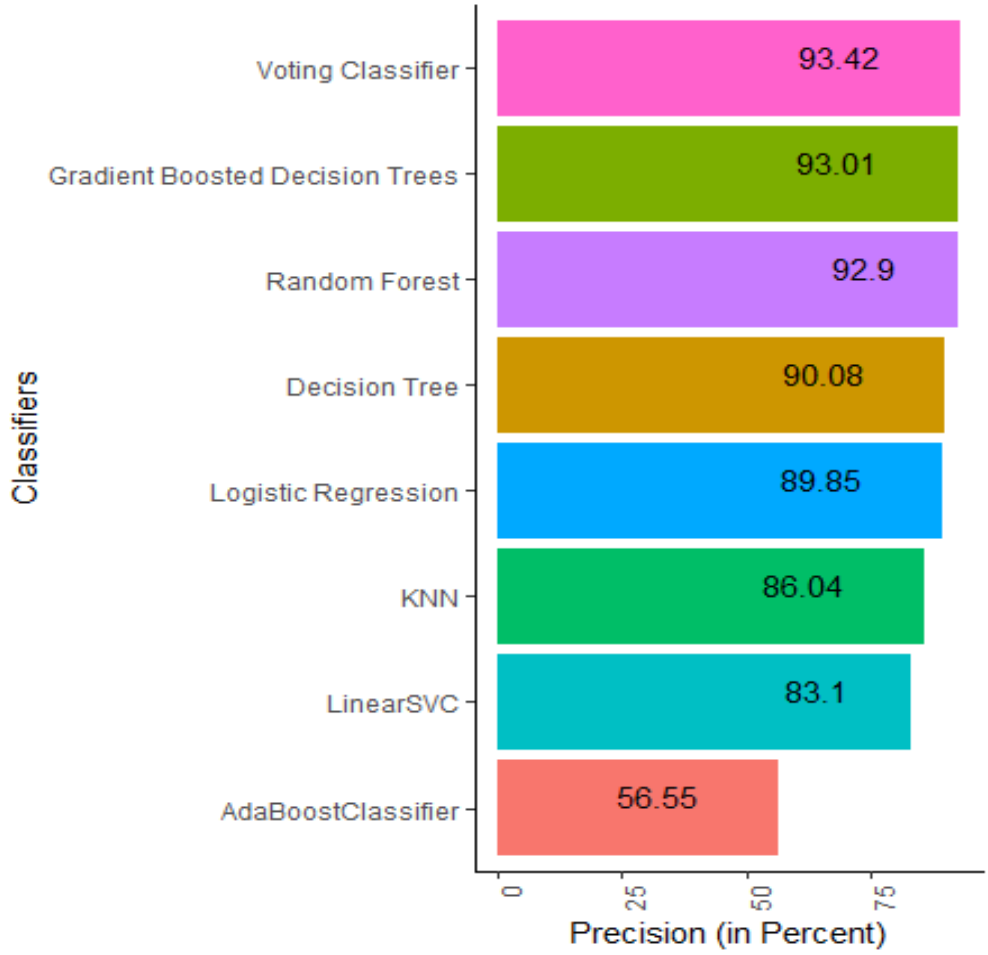

Fig. 19 Performance on Training Data

Although we could make our voting classifier by combining all the base classifiers but this is not considered a good approach. So, we tried all possible combinations of base classifiers and finally achieved best Precision

score by combining Random Forest, Gradient Boosted Decision Tree, KNN and Logistic Regression only.

From Fig. 19, as we achieved best precision score by our voting classifier model that we further used for final predictions. The result that we achieved has an accuracy of 91.23% that has outperformed the hybrid model [5] used earlier by researchers in their study.

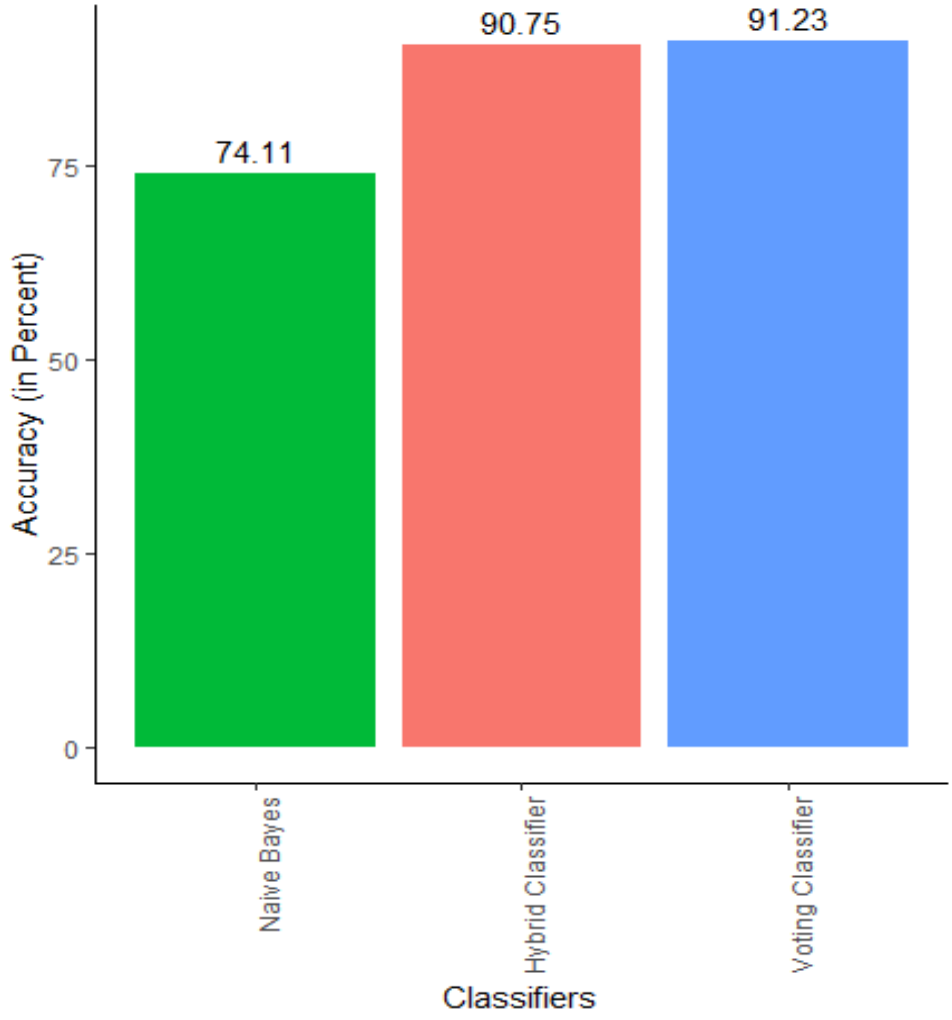

Fig. 20 Final Classifier Results

Chapter 6

CONCLUSION AND DISCUSSION

6.1 Analysis of Results

This chapter concludes the results and implications emerging from the results of this study. Furthermore, limitations and future research suggestions are discussed. Firstly, results indicates that our idea of study i.e. Implementing Voting classifier for predicting customer future purchases has outperformed LinearSVC, Random Forest, KNN, Logistic Regression, Decision Tree, AdaBoost and even Gradient Boosted Decision Tree. Our Model has also got accuracy higher than previous studies performed.

6.2 Limitations and Future Research

There are multiple considerations regarding the limitations faced in this project and future research on this field and continuing the work of this project. First, the data and so information in this project had to be cut down to third because only one row per customer was used. The retail company required this limitation, but it also kept the scope of this project on a higher level. Limiting the scope allowed to get a better overview of the current field of CSB and concentrate more on the comparison of different ML models and current ways of working in the retail company compared to ML methods. However, in future research, it could make sense to include the time dimension of a customer to ML models to get more information regarding a

customer. Taking the time dimension into account could also relieve the imbalance problem since customers would be seen in a different light by the model.

Furthermore, future research could be conducted by using more data, other features, or datasets from the current retail company. Especially other features could make sense because the number of features was cut down by more than half by the feature selection algorithm. Additionally, multiple studies have been able to get high accuracy on CSB by using behavioral data, which would also be interesting.

Secondly, this study did not take comprehensibility into account, which has been in interest on the CSB field, as was presented in the literature review of this project. It makes sense for companies to also get the information out of the model why their customers are purchasing a particular kind of product more often, not just that they are purchasing. From some of the models used in this study, it would be possible to extract the feature importance of different features, but it isn't included in the scope.

Thirdly, future research could include other, more complex models to predict the churn. Some examples could include weighted random forests, hybrid models that could be used on unstructured data. This way, it would be possible to mine out features that could be used in future CSB research in the retail company. Other ways the models could be improved would be using hybrid models that were achieving significant performance gains, as explained in the literature review.

Lastly, extensive grid searches and other optimizations on a large dataset are

not feasible on a home computer because they require multiple iterations to find optimal settings, which could affect the performance of the models significantly. Hence, in future research, either more performing computers are suggested to be used or for example, cloud computing.

REFERENCES

[1] Abdullah Al-Mudimigh, Farrukh Saleem, Zahid Ullah Department of Information System: Efficient implementation of data mining: improve customer's behavior, 2009 IEEE ,(2009),pp.7-10.

[2] Euiho Suh, Seungjae Lim, Hyunseok Hwang, Suyeon Kim : A prediction model for the purchase probability of anonymous customers to support real time web marketing: a case study, Expert Systems with Applications 27 ,(2004), pp. 245-250.

[3] Juni Nurma Sari, Lukito Nugroho, Ridi Ferdiana, Paulus Insap Santosa: Review on Customer Segmentation Technique on Ecommerce, Journal of Computational and Theoretical Nanoscience, 22(10):3018-3022, Oct-2016

[4] Jing Wu, Zheng Lin: Research on customer segmentation model by clustering, ICEC'05, Proceedings of the 7th International conference on Electronic commerce, Page 316-318, Aug-2005, DOI:10.1145/1089551.1089610

[5] Kareena, Raj Kumar: A Consumer Behavior Prediction Method for E-Commerce Application, International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878, Volume - 8, Issue – 2S6, Jul-2019

[6] A.Salini, U.Jeyapriya: A Majority Vote Based Ensemble Classifier for Predicting Students Academic Performance, International Journal of Pure and Applied Mathematics, Volume 118 No. 24, ISSN: 1314-3395, Mar-2018

[7] B.B. Goyal, Meghna Aggarwal: Organized retailing in India - An empirical study of appropriate formats and expected trends, Global Journal of Business Research, Volume 3, Number 2, 2009

[8] Market Segmentation, Targeting and Positioning: Travel Marketing, Tourism Economics and the Airline Product: An Introduction to Theory and Practice, Edition: 1, Chapter: 4, Springer, Dec-2017, DOI: 10.1007/978-3-319-49849-2_4

[9] Dr. Michael J. Garbade, A Simple Introduction to Natural Language Processing: Becoming Human: Artificial Intelligence Magazine, Oct- 2018

[10] Yoseph Linde, Andres Buzo, Robert M. Gray: An Algorithm for Vector Quantizer Design, *IEEE Transactions on communications, vol. com-*28, no. 1, (january 1980), pp. 84-86.

[11] Sung Ho Ha, Sang Chan Park, Sung Min Bae : Customer's time-variant purchase behavior and corresponding marketing strategies: an online retailer's case, *Computers & Industrial Engineering 43 (2002) 801–820,* (2002),pp.801-806.

[12] Mu-Chen Chen, Hsu-Hwa Chang, Ai-Lun Chiu : Mining changes in customer behavior in retail marketing, *Expert Systems with Applications 28* ,(2005), pp. 773-776.

[13] Ian H. Witten & Eibe Frank : Data Mining : Practical machime learning tools and techniques, *San Francisco, Morgan Kaufmann publishers,*

(2005), pp. 112-118,136-139.

[14] Ilung Pranata and Geoff Skinner. “Segmenting and targeting customers through clusters selection & analysis”. In: Advanced Computer Science and Information Systems (ICACSIS), 2015 International Conference on. IEEE. 2015, pp. 303–308.

[15] Trupti M Kodinariya and Prashant R Makwana. “Review on determining number of Cluster in K-Means Clustering”. In: International Journal 1.6 (2013), pp. 90–95.

[16] Peter J Rousseeuw. “Silhouettes: a graphical aid to the interpretation and validation of cluster analysis”. In: Journal of computational and applied mathematics 20 (1987), pp. 53–65.

[17] Volodymyr Kuleshov. “Fast algorithms for sparse principal component analysis based on Rayleigh quotient iteration”. In: International Conference on Machine Learning. 2013, pp. 1418–1425