Predicting Customer Purchasing Intention

Akash Deoraj

Supervised By

Yuhua Li

Abstract

Vast number of people visiting ecommerce website may not have the intention to make a purchase. This could be due to various reasons. However, based on a user's activity within the ecommerce website can give us a indication if the user is likely has an intent to purchase or not. In this project I will be exploring the possibility of using google analytics data of user activity withing an ecommerce website to predict customer purchase intention. Using the application of Machine Learning algorithms to derive highly accurate prediction models. Ability to predict customer purchase intention can be invaluably helpful to ecommerce businesses as it allows them to understand the digital retail space better.

Acknowledgements

I would like to thank my supervisor Dr. Yuhua Li for all the guidance I received throughout the project. I would also like to thank my Friends and Family for their support.

Table of Contents

1.Introduction

Motivation

Retail shopping is continuing to shift to E-commerce shopping and as a result the dynamics of shopping is changing around the world. E-commerce has already become a major form of retail market. Online customers often browse pages of e-commerce sites before they place orders or abandon their browsing without purchase. This information can help businesses to better cater to customer preferences and help both the business and customers mutually by recommending products specific to each customer and therefore increasing sales for the businesses. However, most of the time customers visiting these online websites may not make any purchase at all. This could be for various reasons i.e., Price of product or window shopping. It is important to predict customers' purchasing intention so that retention measures like e.g., recommending suitable products can be taken to convert potential customers into purchasers. Currently, the closest existing solution to this problem has been the recommendation system. Where previous purchase information from a customer is processed to predict the types of products a customer would be interested in. There is evidence [12] to suggest that retention measures such as an apt recommendation system plays an extremely important part in converting sales. Here the prediction of customer purchase intention can help strategize different marketing strategies and could be added to the mechanism of the recommendation system[13] of an ecommerce retailer. An example could be that if the ML solution predicts a strong customer purchase intention, then maybe the recommend system could recommend a higher quality or a more expensive product as it can be inferred that the user would be willing to consider a better or more expensive product if their intention to purchase a type of product is very strong. If the solution predicts a low purchase intention, then recommendation system could recommend products that are on discount or products with special offers i.e. “Buy one get one free”. Later this historical data of how customer intention changes with such recommendation can also be studied and be applied to improve the recommendation system itself further. **However**, this report focuses only to the extent of predicting customer purchase intention.

Aims and objective of the project

This project aims to use the information customers may leave in the form of the trace of browsing history data or user information when they visit an online shopping site. With the help of this information, the project aims to predict online shoppers' purchasing intention by using clickstream and session information data. The project aims to create a machine learning model based on this information to predict customer’s purchasing intension. The Objective of the project is to build a Machine Learning that can predict customer purchase intention as accurately as possible.

Scope of the project

The scope of this project is only limited to predicting customer purchase intention and evaluating and measuring the accuracy of these predictions.

2.Background

This section will be covering some of the technical knowledge researched to develop the final solution. It will explain the concepts, dataset used, software, plugins that are used or covered to address the problem being solved. It will also cover the wider context of the project, constraints of the approach and stakeholders within the problem area.

Background technical research.

Machine Learning.

As mentioned in the introduction, the problem is classified as a Machine Learning problem. However, here I will be going in depth into that is Machine Learning and why Machine learning is a suitable approach to predict customer purchase intention. Machine learning is a type of computer algorithms that improve automatically through interaction and collecting new data.[14] It is a subset of artificial intelligence. Machine learning algorithms build a model based using a sample dataset this is also called training data. It then builds the model on the sample dataset to make predictions or decisions without being explicitly programmed to do so.[15] We can see from how Machine learning algorithm function and know that problems that have too many rules to identify a solution will generally be a Machine Learning problem. As there are too many rules and edge cases, traditional programming can be nearly impossible to build a good solution. For example, creating a program to identify if an image is of a cat cannot be traditionally programmed due to numerous challenges in defining the rules, addressing the edge cases and many such limitation. It is more effective to let the machine develop its own algorithm, rather than having human programmers specify every needed step[16]. Using machine learning this problem can be addressed as the Machine Learning algorithm builds models that use various statistical and mathematical reasoning to approximate the image it is looking at to the target image. The more varied images of a cat it is familiar with, the better it will get at identifying an image of a cat in the future. This is in a way similar to how human brains functions; humans use their memory or data from past experience to make sense of information in the present.

In the case of this report, I am trying to predict customer purchase intension which cannot be solved easily with traditional programming due to limitations discussed above. So, I will use the Machine Learning algorithms to build a model or an algorithm that can help predict the outcome of customer purchase intention with as high a degree of accuracy as possible.

There are two main Machine Learning problems. These are Supervised and Unsupervised. Here I have researching on what these two different types of Machines Learning model are trying to solve and to identify the most suitable Machine Learning model to the problem I am trying to solve in this report.

Supervised Learning

Supervised Learning is when the machine learning task of learning a function that maps an input to an output based on example input-output pairs.[1] It infers a function from labelled training data consisting of a set of training examples.[2] When training a supervised learning algorithm, the training data will consist of inputs paired with the correct outputs. During training, the algorithm will search for patterns in the data that correlate with the desired outputs. After training, a supervised learning algorithm will take in new unseen inputs and will determine which label the new inputs will be classified as based on prior training data. The objective of a supervised learning model is to predict the correct label for newly presented input data. At its most basic form, a supervised learning algorithm can be written simply as:

$$y = f(x)$$

Where Y is the predicted output that is determined by a mapping function that assigns a class to an input value x. The function used to connect input features to a predicted output is created by the machine learning model during training. Supervised learning can be split into two subcategories [3]:

Classification: During training, a classification algorithm will be given data points with an assigned category. The job of a classification algorithm is to then take an input value and assign it a class, or category, that it fits into based on the training data provided. The model will find correlation between features within the data and class to create the mapping function mentioned earlier: *Y=f(x).*

Regression: Regression is a predictive statistical process where the model attempts to find the important relationship between dependent and independent variables. The goal of a regression algorithm is to predict by identifying the factors that create an impact in the outcome of the predict. In a regression model we have dependent variables and independent variables. Dependent variable are the main factors that have an impact on the outcome whereas the independent variables are variable that we suspect to have an impact on the dependent variables. The equation for basic linear regression can be written as so:

$$y = w[0] * x[0] + w[1] * x[1] + \cdots + w[i] * x[i] + b$$

Where *x[i]* is the features for the data and where *w[i]* and *b* are parameters which are developed during training. For simple linear regression models with only one feature in the data, the formula looks like this [3]:

$$y = mx + c$$

Unsupervised learning

Unsupervised learning, also known as unsupervised machine learning, uses machine learning algorithms to analyse and cluster unlabelled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information make it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, and image recognition [4]. Two of the main methods used in unsupervised learning are principal component and cluster analysis. Cluster analysis is used in unsupervised learning to group, or segment, datasets with shared attributes in order to extrapolate algorithmic relationships.[5] In other words, clustering deals with finding a *structure* in a collection of unlabelled data by finding distinct groups in the dataset. For clustering, we need to define a proximity measure for two data points. Proximity here means how similar/dissimilar the samples are with respect to each other. A "good" proximity measure is VERY application dependent. The clusters should be invariant under the transformations "natural" to the problem [6]. Clustering algorithms can be classified as follow [7]

Exclusive Clustering: In exclusive clustering data are grouped in an exclusive way, so that a certain datum belongs to only one definite cluster. K-means clustering is one example of the exclusive clustering algorithms

Overlapping Clustering: The overlapping clustering uses fuzzy sets to cluster data, so that each point may belong to two or more clusters with different degrees of membership.

Hierarchical Clustering: Hierarchical clustering algorithm has two versions: agglomerative clustering and divisive clustering Agglomerative clustering is based on the union between the two nearest clusters. The beginning condition is realized by setting every datum as a cluster. After a few iterations it reaches the final clusters wanted. Basically, this is a bottom-up version Divisive clustering starts from one cluster containing all data items. At each step, clusters are successively split into smaller clusters according to some dissimilarity. Basically, this is a top-down version.

Probabilistic Clustering: Probabilistic clustering, e.g., Mixture of Gaussian, uses a completely probabilistic approach.

Some of the common clustering algorithms are K-means, Fuzzy K-means, Mixture of Gaussians

Selecting between Supervised Learning and Unsupervised Learning

Here it can be seen that the problem is based on predicting the "Revenue" class label using labelled features and so we can classify the problems as a Supervised Learning problem. This helps us understand that it is more suitable and appropriate to implement a Supervised Learning ML algorithm instead of an Unsupervised learning ML algorithm because in the case of an unsupervised learning problem the data is unlabelled, and the purpose is to find structure in the data. As mentioned earlier the supervised learning can be split into two subcategories. These are **Classification** and **Regression.** I have researched algorithms from both the subcategories- classification and regression below.

List of Supervised Machine Learning algorithms researched for this project.

Logistics Regression: the logistic regression model is used to calculate probabilities of a class label or event existing such as pass/fail, win/lose, veg/non-veg/vegan, rating between 1 to 10 . Logistic regression uses the logistics function to model the dependant variable or the class field i.e pass/win. However more complex dependant variable which have more than two classifications i.e vegan/veg/non-veg can also be modelled. These multiple classification logistic models are usually done using Multinomial logistic regression. As mentioned earlier, there are than one way type of logistic regression model.

Binary Logistic Regression: This type of logistic model is used to model the binary dependent variable. Binary Logistic regression is best used when the categorical response or outcome is binary in nature. For example, when the target is either 1 or 0. Another example would be if the outcome to email could be “spam” or “not spam”.

Multinomial Logistic Regression: Multinomial logistic regression is a classification method that generalizes logistic regression to multiclass problems[9]. This version of Logistic regression model is used when the dependent variable or target we are trying to model has more than two outcomes. It is a model used to predict the probabilities of the different possible outcomes of a distributed dependent variable. For example, predicting which food is preferred more (Veg, Non-Veg, Vegan).

Ordinal Logistic Regression: Ordinal Logistic Regression model is like Multinomial Logistic regression. It is used when the outcome has a meaningful order, and when more than two categories or possible outcome exist.

Logistics regression uses the logistic function which is a sigmoid function that is used as the target class estimator. Here it takes a linear combination of features and applies a nonlinear function to produce an output. It takes in any real value t and outputs a value between 1 and 0. Below is an equation and graph of a sigmoid function.

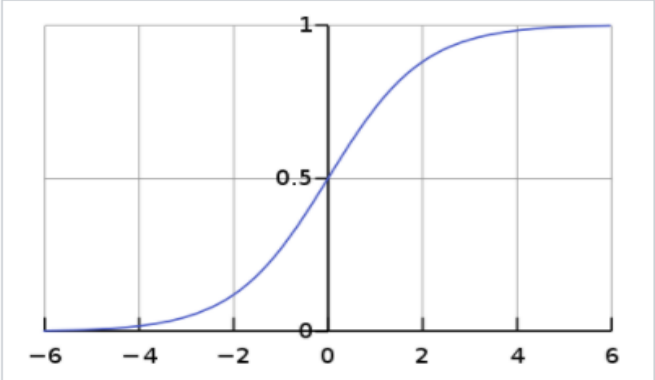

Fig.1

$$g(t) = \frac{1}{1-e^{-z}}$$

The goal in Logistic Regression ML algorithm is to model the probability of a random variable being 0 or 1 given experimental data[9]. For the implementation of the Logistic Regression on the dataset, I will be selecting the *Binary Logistic Regression* as it is the most suitable type of logistic regression. The intent of the solution is predict two potential targets which are either “1” or “0” for the customer intent where 1 and 0 represent true of false indicating the nature of customer purchase intension.

Random Forest: Random Forest is a collection of decision trees. Decision trees have nodes and leaves. The nodes are decisions, and the leaves represent the choices to the decisions. Decision nodes are where the data is split. Each choice to the decision then becomes a node and may have further leaves that represents further decision choices and so on and the trail continues. Following the path of the decision tree by choosing nodes and leaves leads to the final decision or outcome. Below is an example of a decision tree.

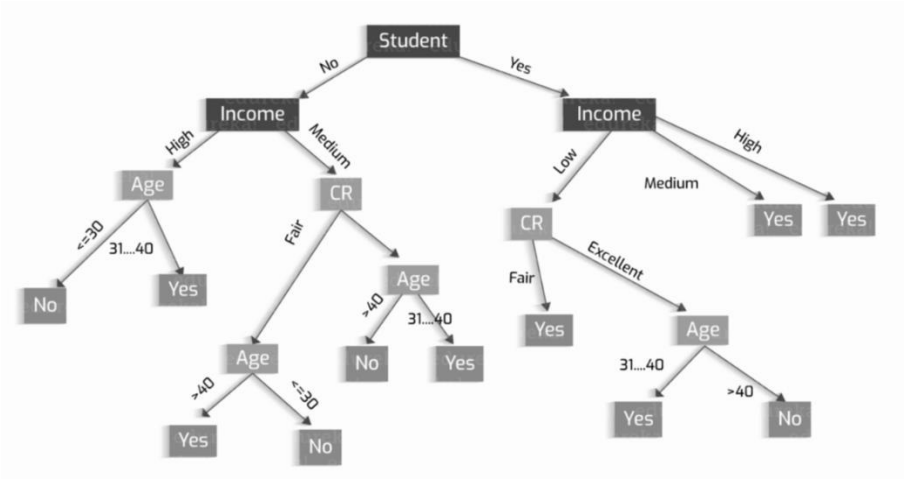

Fig.2 Decision Trees

Random Forest uses a collection of individual decision trees to make decision. It makes use of **randomly created decision trees**. Here each node in the decision tree works on a random subset of features to calculate the output. The random forest then combines the output of individual decision trees to generate the final output [10]. The process of combining the output of multiple individual models is called Ensemble Learning. To compare between decision tree and a decision tree, the decision tree is built on an entire dataset, using all the features/variables of interest, whereas a random forest randomly selects observations/rows and specific features/variables to build multiple decision trees from and then averages the results. After many trees are built using this method, each tree "votes" or chooses the class, and the class receiving the most votes by a simple majority is the "winner" or predicted class. This works well in the dataset where there more uncorrelated features as seen the data exploration stage earlier. The reason random forest works better in datasets with large uncorrelated data is because while some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. Below is an illustration of a Random Forest

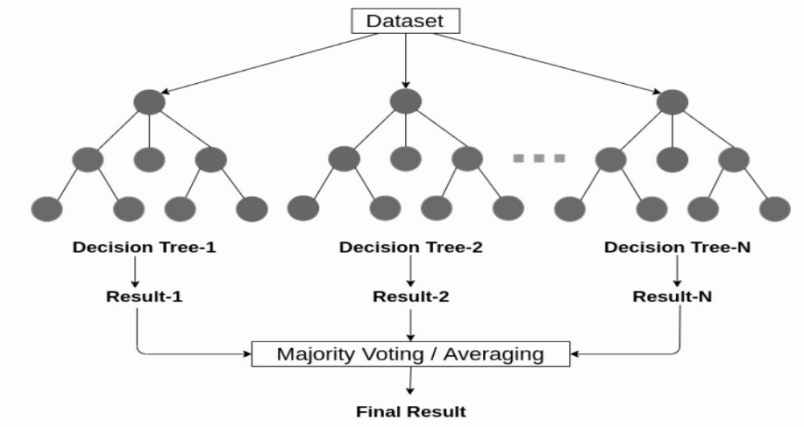

Fig.3 Illustration of Random Forest Algorithm

Extra Trees: Extra tree is similar to random forest tree as it also makes use of the ensemble method where multiple collection of individual decision tree to make a prediction. However, In terms of computational cost the Extra Trees algorithm is less expensive, and therefore execution time Is **faster than Random Forest**. In this report I am using decision tree as an alternative approach to the Random Forest algorithm. This algorithm saves time in learning but is nearly as effective as random forest this is because the most of Extra Tree's algorithm procedure is the same as that of Random Forest, but it randomly chooses the split point instead of calculating the optimal one as Random Forest does.

Another difference is the selection of cut points to split nodes. Random Forest chooses the optimum split while Extra Trees chooses it randomly. However, once the split points are selected, the two algorithms choose the best one between all the subset of features. Therefore, Extra Trees adds randomization but still has optimization [11]. The randomization itself adds a layer an additional feature to the algorithm, this can greatly improve prediction performance if there are irregular features that could not be detected during pre-processing and there it can generalize better [20] than Random Forest when there is noise in the data. However, there are exceptions to the case where extra tree do not perform better than Random forest when there is noise left in the dataset. In general, extra tree does better at fitting the data and generalizing than Random Forest.

Deep Learning Neural Network or Deep Learning is an extension of Artificial Neural Network. Artificial neural networks (ANNs)—mimic the human brain through a set of algorithms. A basic artificial neural network is comprised of four main components: inputs, weights, a bias or threshold, and an output.[21] The input is the information received by the neuron or unit. This information generally comes from a fellow neuron unit. The weights are the conditions set or factors identified during the learning process that has an impact on the outcome. The weights control the signal (or the strength of the connection) between two neurons. A weight decides how much influence the input will have on the output. If the output of any individual node is above the specified threshold value then that node is activated sending data to the next layer of the network. Threshold controls the communication information between the neurons. Bias is a constant value that is needed for the neurons to function, if the input is all 0 then the neuron may not get activated.

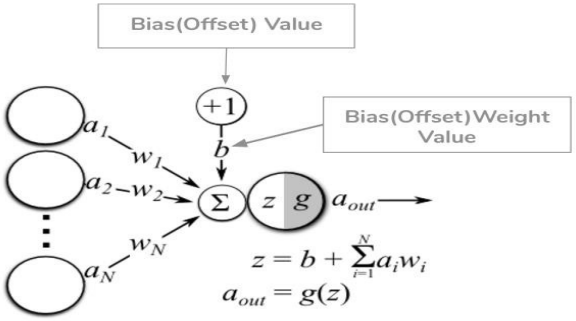

Fig.8 Neuron

These neurons are connected to each other through a network. These neurons or units receive various forms and structures of from the input data and based on an internal weighting algorithm, and the neural network attempts to learn about the data presented to it.

Each of the node or neurons function like a regression model composed of input data, weights, a bias (or threshold), and an output. The regression formula can be represented as below.

$$Y = \sum (weight * input) + (bias + threshold)$$

The main difference between regression and a neural network is the impact of change on a single weight. In regression, you can change a weight without affecting the other inputs in a function. However, this is not the case with neural networks. **Deep Learning** [22]is when the layers of ANN is at least 3 or more. The additional hidden layers can help to optimize and refine for accuracy than an average Artificial Neural Network.

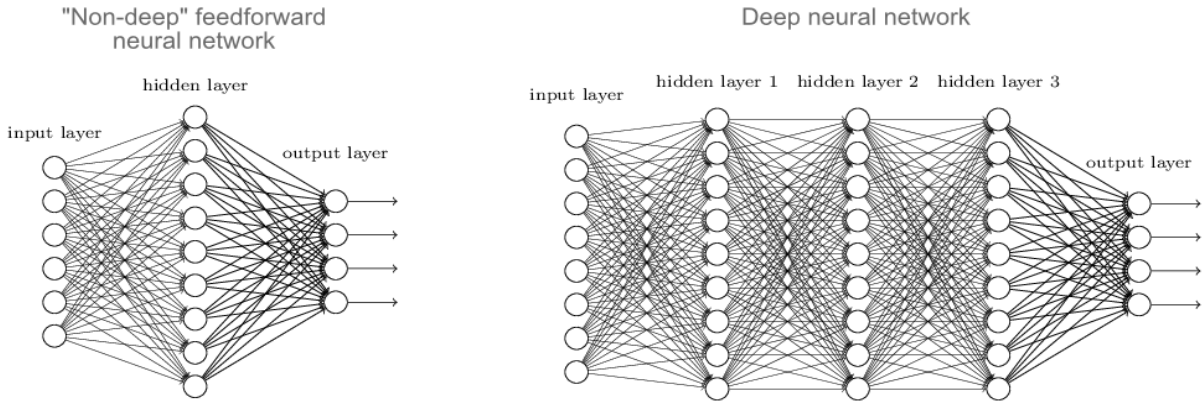

Fig.9 Artificial Neural Network vs Deep Neural Network

Software, Plugins and Other Tools

Python: is a programming language that support large number of open-source libraries that are used in implementing machine learning algorithm and for data visualization and analysis.

pandas: is open-source data analysis and manipulation tool, built on top of the Python programming language.

Imbalanced-learn: is an open source, MIT-licensed library relying on scikit-learn and provides tools when dealing with classification with imbalanced classes[17].

Scikit-learn: is an open source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection and evaluation, and many other utilities [16].

category_encoders: Is a python library that allows to implement One Hot Encoding on features in the dataset.[19]

Keras: is a deep learning framework will be used to implement the ANN(Artificial Neural Network) and deep learning models.

Google CoLab: It is an online programming platform that is especially well suited to machine learning, data analysis and education. I have used google colab as it provides better hardware for Machine Learning computing for example for training the Machine Learning.

Wider context of the project

The algorithms discussed to solve problem of predicting customer purchase intention could be useful in improving recommendation system of ecommerce website to improve suggestion based on customer purchase intention. It can also benefit ecommerce organization understand their average customer's purchase intention and it can also help in assessing the extent of success of their marketing strategy. Any changes to website page layout, external marketing etc can have an impact on the average customer purchase intent. Meaning more people are arriving to the ecommerce website with a strong intension to make a purchase.

Stakeholders within the problem area.

The stakeholders in this problem area are both the ecommerce retailers as well as the customers themselves. This is so because by predicting customer purchase intention firms can make their marketing strategy more effective towards each customer by tailoring their marketing strategy to suit each user arriving to their ecommerce website by predicting if the customer has higher or lower intension of purchase and applying specific strategies such as discounts or "buy one get one free" deals if the customer intension is low or suggesting better and more expensive products if purchasing intention is high as it can be inferred that customer would be interested in considering purchasing more expensive or higher quality product if their intention to purchase a type of product is high. This will help convert more sales effectively and it would also reduce wastage in in marketing as there will be extra datapoints for the business to study and implement better marketing strategy. At the same time, the customer also benefits as they can get better deals and suggestions by the ecommerce website's algorithm which would otherwise take time a lot of time and effort to find.

Constraints on the approach to be adopted.

One of the constraints of the Machine Learning approach is the quality of data. The quality of data and the accuracy of the data can determine the quality of the prediction. If the dataset is not accurate and representative of actual customer behaviour, then the Machine Learning algorithm will create biases in the model and give false predictions. We would also need to a balanced dataset that has approximately equal number of possible outcomes. i.e Customer purchase intention “Yes” or “No”.

3.Specification & Design

The system being designed is a software application that takes in specific data collected from google analytics and predicts the customer purchase intention. The application produces a result in as either 1 or 0 for the feature “revenue” which is represented by Boolean value True or False. Generally, the machine learning algorithm model will learn the probability of each event based on the variables or features of the dataset and make a prediction. The internal working of each machine learning algorithm will differ. I shall be measuring the fitness and accuracy of each of the machine learning algorithms on the processed dataset before and after optimization of the models and comparing the performance for most accurate solution. In this project, I will be aiming to achieve as high an accuracy as possible in predicting the

The system architecture is shown below.

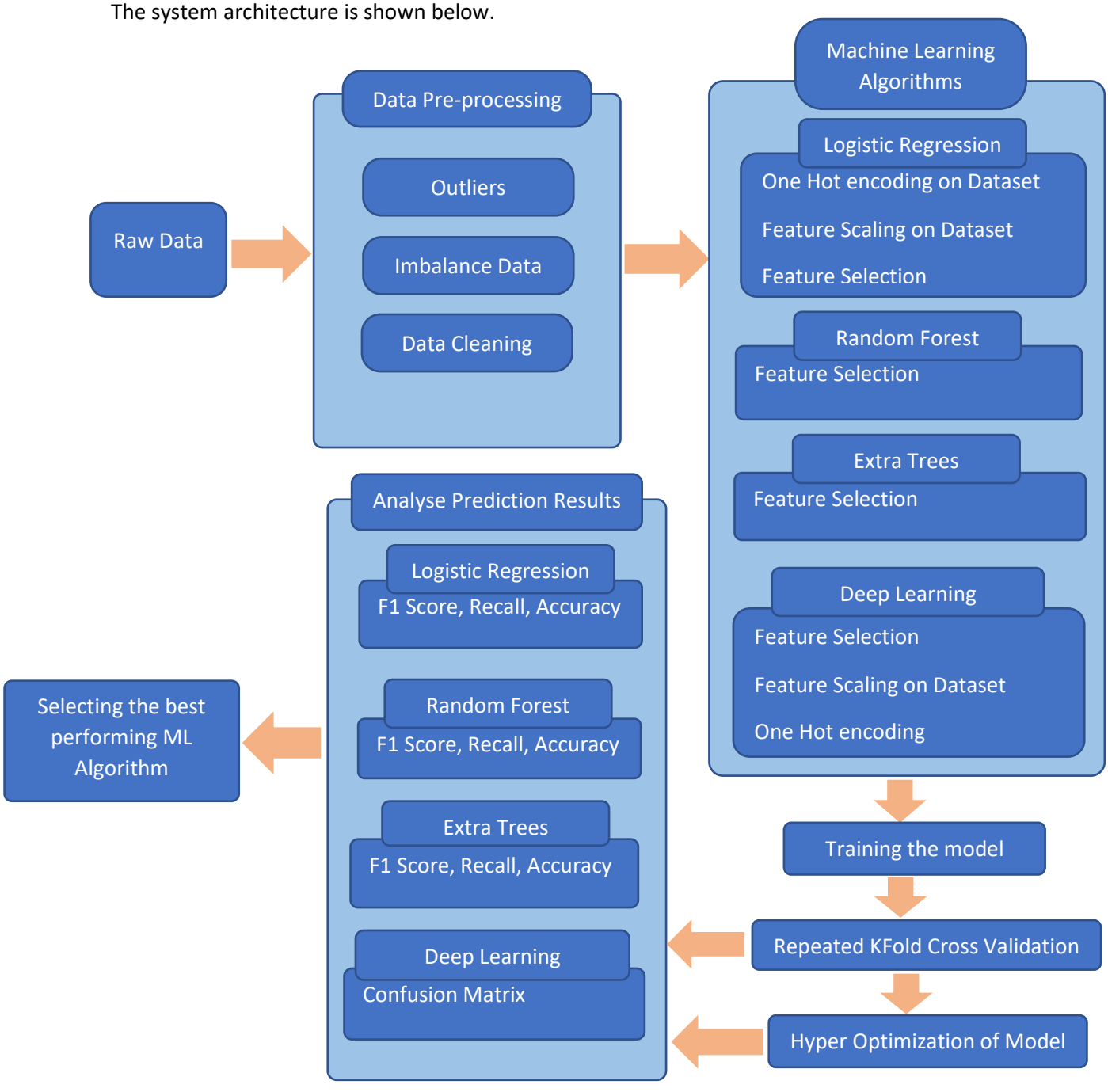

Fig.4 Architecture of the solution

Input Data: The original dataset will be split into training and testing data with a ratio of 8:2 proportion, respectively. 80% of the dataset will be used for training the Machine Learning Model and the rest 20% will be used to test the accuracy of the model built. Once the model is built with a sufficient level of accuracy then new data will be used to predict customer purchase intention at the same time the data will reused to improve the current ML model through continuous training as new data arrives.

Machine Learning Model: The machine learning model is designed by considering various aspects which are explain below.

Data Pre-processing: Here I will be formatting the dataset to so that the machine learning algorithm will be able to create more effective prediction model. Through the process of data pre-processing, I will be cleaning the dataset of missing values and other such human errors. These could include wrong entry of data in a field for example, String value in an Integer Datatype or even typing errors. However, the dataset being used for this project has data formatting at the point of collection and so typing errors and incorrect datatypes are unlikely. The dataset used for this project is data collected through google analytics which has certain formatting in place at point of data collection. Although areas such as missing values, imbalanced dataset, outliers, and other abnormalities in the dataset will have to be addressed before the dataset can be used for building the machine learning model. By removing unwanted data from the dataset, the final dataset used will be more effective as there will be less incorrect information or noise in the data.

Outliers: Outliers are certain data values that are extremely rare occurrences. For example, in the case of this dataset, one of the information gathered is the duration of time spent on a product page. If a user happens to leave their desktop whilst on a product page of the ecommerce website and happens to arrive back to their desktop half an hour later to continue using the ecommerce website, then the analytics would calculate that the user has spent half an hour reading the product page. This is unlikely to occur and is a rare occurrence and unlikely to represent a typical user to the ecommerce website. By removing such outliers, it makes the dataset more representative of actual user behaviour and therefore the machine learning algorithm can build a more predictive model. Removing Outliners is a part of data pre-processing.

Imbalanced dataset: Imbalanced dataset is when the dataset does not have an equal number of each outcome. In the current dataset used for predicting if the customer's intention is to purchase or not to purchase, there are two possible outcomes which are yes or no. For the Machine Learning algorithm to build a reliable model there must be approximately equal instances of both outcomes in the training dataset. If the dataset is imbalanced the Machine Learning model will not fully learn how to predict the unbalanced output. Here I will be using the application of plugins to create synthetic data to adjust the dataset and balance it. Balancing the dataset is also part of data pre-processing.

Optimization: Optimization or Hyperparameter Optimization is the process of tuning the machine learning model by adjusting the parameters of the algorithm to best fit the dataset. The parameters have an impact in the learning process of the machine learning algorithm. I will be implementing random search algorithm to find the optimal parameters suited for the dataset.

Feature Scaling: Feature scaling is a method used to normalize data. Normalization is the process of scaling integer values of independent variables or features of data between 0 and 1 range. Feature Scaling is helpful in machine learning algorithms especially those algorithms that use that use gradient descent. However, algorithms such as Random Forest and Extra trees do not use gradient

descent and therefore does not have a significant impact in the model's performance. Among the machine learning algorithm being implementing, Logistics regression makes use of gradient descent. I will be explaining gradient descent used in logistic regression in the implementation section of the report.

One Hot Encoding: One Hot Encoding is a type of data formatting by which a machine learning model can interpret the categorical features information better or non integer variable better. For example, In the dataset there is a categorical feature called "Month". If this data is encoded as 1 to 12 representing months January to December, then the machine learning model can build a bias as it would consider December higher in value than say January as December is represented by 12 where as January by 1. In order to solve this problem, One Hot Encoding will represent each of the month with a combination of 1s and 0s much like bits. An example is shown below.

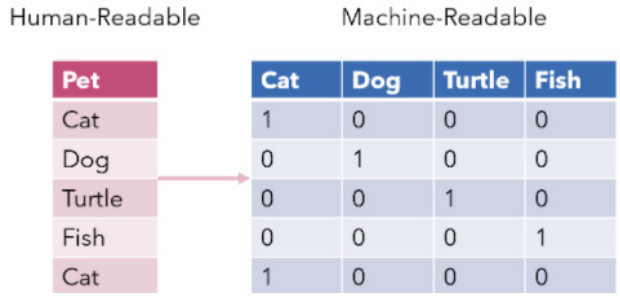

Fig.5 Example of OneHotEncoding

Feature Selection: Feature Selection is process of selecting only features that an impact on the prediction and removing irrelevant features. This helps in improving prediction accuracy as the model does not learn and incorporate irrelevant data into the learning process. Feature selection can be done manually as well as automatically. Either through statistics or python plugins, respectively.

Algorithms: The algorithms being implemented are Random Forest, Extra Trees and Logistics Regression and Artificial Neural Network. Although they are all supervised learning algorithms. Each of these algorithms will produce different levels of accuracy and different levels of accuracy at different stages of pre-processing. I have chosen a range of algorithms that approach prediction in different ways. Artificial Neural Network uses inference model to predict the output which in this case is the customer purchase intension. Logistic regression uses gradient descent to build the prediction model while Random Forest and Extra Trees uses decision trees. These unique approaches will understand the dataset in different approaches and help in creating a highly predictable solution for customer purchase intension. I will also be comparing accuracy of the machine learning algorithms at different stages of pre-processing and produce the results in the results and evaluation section of this report.

Training the Model: The training of the model means that a portion of the dataset after pre-processing is complete is used to train or teach the machine learning algorithm to build model that will be able to predict the customer purchase intention. I will be using 80% of the processed dataset for training and the rest 20% for testing.

Cross Validation: One of the techniques used to test the effectiveness of a machine learning models. If a Machine learning model is too overfit/underfit then the Cross Validation Score will low. Here we check to see if the model build has able to generalize the problem well and not simply overfit the model to just work well only on the training data.

Testing: Testing the accuracy of the model among various machine learning algorithms with and without Hyper parameter optimization. The method of testing is based on the key metrics as follows

$TP = True\ Positive$: This is when the model predicts True and in reality, it is also True.
$TN = True\ Negative$. This is when the model predicts False in reality, it is also False.
$FP = False\ Positive$. This is when model predicts True but, it is False
$FN = False\ Negative$. This is when the model predicts False and it is also True

Another way to view this data is through the confusion matrix as shown below.

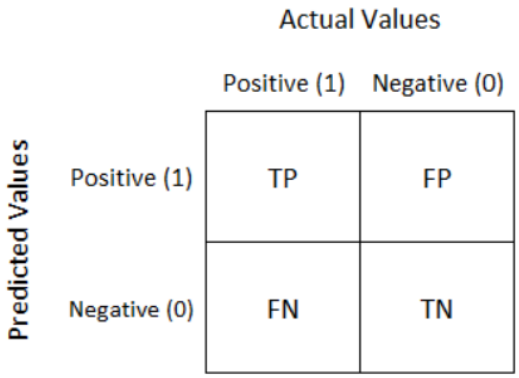

Fig.6

By using the sum of each of these variables the Accuracy, Recall, F1 Score and Precision can be calculated.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Accuracy: shows a value for the number prediction made by the machine learning model was right.
Recall: shows us the proportion of actual positives that were identified correctly.
Precision: shows the proportion of positive identifications that was correct for both True and False prediction.
F1 Score: Score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.

4.Implementation

4.1. Data Pre-processing

I have investigated the dataset and performed data exploration to gain insight into the dataset. The dataset is contained in a csv file. A basic description of the dataset is that it has 18 features or fields containing a range of Integer values, String and Boolean values. The target field in this dataset is the "revenue" field. This field represents weather customer has purchase intention was true or false. The dataset in total consists of 10 numerical and 8 categorical attributes.

I have loaded the csv dataset file using python's pandas plugin.

```
import pandas as pd

rawDataSet = pd.read_csv('online_shoppers_intention.csv', index_col=0)
```

Checking for missing values

Checking missing data by counting the sum of null values in each column or feature. **No null** values found in any feature in the dataset. Since this dataset is retrieved from google analytics, formatting at the point of collection may have helped elimination common types of data corruption such as missing values. Here I have used the .isnull() function to check if any fields within the dataset for all

```
rawDataSet.isnull().sum()
```

```
[12330 rows x 17 columns]
Administrative_Duration    0
Informational              0
Informational_Duration     0
ProductRelated             0
ProductRelated_Duration    0
BounceRates                0
ExitRates                  0
PageValues                 0
SpecialDay                 0
Month                      0
OperatingSystems           0
Browser                    0
Region                     0
TrafficType                0
VisitorType                0
Weekend                    0
Revenue                    0
dtype: int64
```

One Hot Encoding.

As mentioned earlier this One Hot Encoding is a process of formatting categorical data to bits. To implement this, I have used the python library category_encoders. This library helps format features within the dataset to one hot encoding. I have one hot encoded the following features in the dataset. These feature are “Month”, “VisitorType”, “OperatingSystem”, “Browser”, “ Region“. Additionally, features with Boolean values have been formatted to integer value 1 and 0. One Hot Encoding was not applied to Boolean features as it is redundant. This data formatting will be an positive impact in the logistics regression algorithms learning process as it uses gradient descent as part of the Machine learning algorithm. However, this data cleaning may not increase performance of Random Forest and Extra Trees as they do not use gradient descent in their Machine learning algorithm.

```
import category_encoders as ce

y0 = rawDataSet.drop('VisitorType', axis = 1)
ce_one_hot = ce.OneHotEncoder(cols = ['VisitorType'])
X = ce_one_hot.fit_transform(X, y0)
```

After the One Hot Encoding the dataset now has 56 numerical features in total.

Feature Scaling.

I have implemented Feature Scaling using the MinMaxScaler() function from the sklearn’s preprocessing library. This function is applied across the whole dataset from all rows and columns. Like One Hot Encoding, feature scaling will not have much of an impact on machine learning algorithms such as Random Forest or Extra Trees as they do not use gradient descent unlike Logistics Regression which will see an improve performance of formatting data with feature scaling.

```
from sklearn import preprocessing

min_max_scaler = preprocessing.MinMaxScaler()
X_train_minmax = min_max_scaler.fit_transform(oneHotEncodedDataSet)
featureScalingDataSet = pd.DataFrame(X_train_minmax)
```

Balancing the dataset.

I have investigated the dataset to check if it has an equal number of each possible outcome of the target field or class label. Here I have used matplot.lib to visual the dataset based on the number of each outcome present in the dataset. If the dataset is not balanced it can be to inaccurate model being created. As shown below the data is initially unbalanced and is later made balanced.

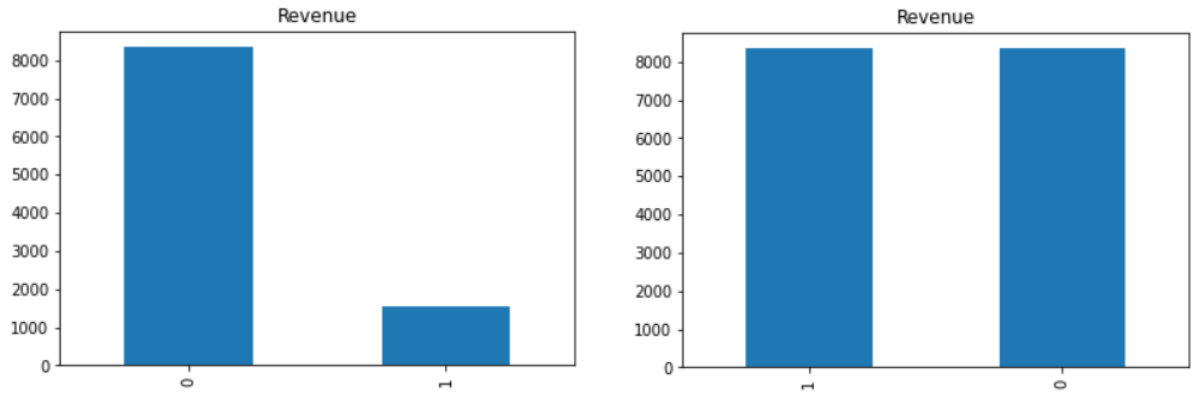

Fig.6 Before and After Oversampling using SMOTE

I have used *sklearn.model_selection.train_test_split* to split the dataset into four portions these are as follows.

X_train - This includes all feature variables that will be used to train the model. Also I have specified the test_size = 0.2, this means 80% of observations from your complete data will be used to train the model and rest 20% will be used to test the model.

X_test - The remaining 20% portion of the non-class label features from the dataset will not be used in the training phase and will be used to make predictions to test the accuracy of the model.

y_train - This is the "revenue" variable which needs to be predicted by this model. This is the class label against which the other features variables, we need to specify our dependent variable while training/fitting the model.

y_test - This data has class label "revenue", this class label data will be used to test the accuracy between actual and predicted categories.

To balance the dataset, I have implemented **SMOTE** using python's imblearn library. SMOTE stands for Synthetic Minority Oversampling Technique. It generates synthetic samples for the minority class label category. In the case of this project the minority class label it "1" which represents by positive customer purchase intension. The SMOTE implementation overcome the overfitting problem posed by random oversampling. It generates new instances of the minority class using linear interpolation. Which helps to create a realistic dataset post oversampling. SMOTE implementation shown below.

```
x_train, x_test, y_train, y_test = train_test_split(smote_x, smote_y, test_size = 0.2, random_state = 45)

sm = SMOTE(random_state=45)
x_train, y_train = sm.fit_sample(x_train,y_train.ravel())
```

Identifying Outliers

Identifying and removing outliers are important so that the model does not overfit to the training data. When outliers exist in the dataset machine learning algorithms can confuse “noise” or irrelevant and/or random data for signal. This can reduce the accuracy of the model when tested against non-training dataset. As the problem will be properly generalized due to outliers. Below if an example of the “Informational_Duration” feature in the dataset. Here we can see that this feature has quite a few and far in between outliers circled in red while after outliers are removed the feature has lesser and fewer far in between outliers than before. The data is now looks more clustered as outliers have been removed.

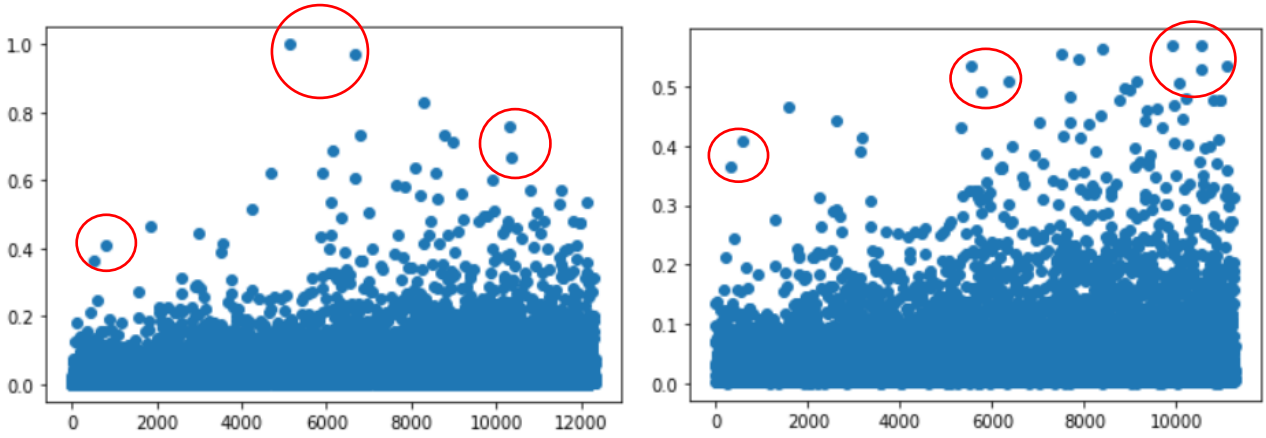

Fig.7 Before and After outliers are removed from “Informational_Duration” feature

To implement identification and removal of outliers, I have used the interquartile range method. I have calculated the percentile for each of the features to identify outliers. I have decided to cut off only very “far out” information from the dataset thus including majority of the data which is between 2rd percentile and 98th percentile. This is because there is a risk of losing signals if too much data is removed. I have chosen to calculate percentile instead of a normal distribution as the data can be assumed to not follow Gaussian distribution.

I have written code in python to slice the bottom 2nd and top 2nd percentile of the data.

```
Q1 = featureScalingDataSet.quantile(0.02)
Q3 = featureScalingDataSet.quantile(0.98)
IQR = Q3 - Q1

dataset_outliners_removed = featureScalingDataSet[~((featureScalingDataSet < (Q1 - 1.5 * IQR))
                            |(featureScalingDataSet > (Q3 + 1.5 * IQR))).any(axis=1)]
```

Feature Selection.

Feature selection will help the algorithm use only the features that have the highest weight on the impact on the out of the prediction. I have used recursive feature elimination (RFE) to shortlist those features that contribute the most to the prediction. RFE searches for a subset of features by starting with all features in the training dataset and eliminating features until the desired number remains. RFE uses a supervised learning estimator that creates a model that provides information about the importance of features or subset of features. I have used ExtraTreeClassifier() as the supervised learning estimator as it is faster than other ensemble algorithms and can also help in identifying relative importance of each feature. The dataset features after one hot encoding had gotten very large with a total of 56 features. By reducing dimensionality or removing features with least impact on the prediction, it has greatly improved the training speed and has improved the overall quality of the training dataset for predicting customer intention. Using REF method, I have ranked the top twelve features which I will be using going forward.

```
X = x_train
Y = y_train
rfe = RFE(estimator= ExtraTreesClassifier(), n_features_to_select=12)

rfe.fit(X,Y)
# summarize all features
for i in range(X.shape[1]):
  if(rfe.support_[i] == True):
    print('Column: %d, Rank: %.3f' % (i, rfe.ranking_[i]))
```

Implementing Machine Learning Algorithm.

Below is a snippet of the implementation of the logistic regression model. I am also doing cross validation on all the Machine learning models using “RepeatedKFold” method to gauge if the model is not over fitted due to the changes during the pre-processing stage. Similarly, I will be implementing the Random Forest, Extra Trees and Artificial Neural Network.

```
def LogisticReg(x_train, x_test, y_train, y_test):

 logisticRegression = LogisticRegression()
 logisticRegression.fit(x_train, y_train)
 y_pred_logisticRegression = logisticRegression.predict(x_test)
 # evaluating the model
 print("LogisticRegression")

 cv = RepeatedKFold(n_splits=10, n_repeats=3, random_state=1)
 # evaluate model
 scores = cross_val_score(logisticRegression, X_train_FeatureSelect, y_train, scoring='accuracy', cv=cv, n_jobs=-1)
 # report performance
 print('Accuracy_CV: %.3f (%.3f)' % (mean(scores), std(scores)))
 print("Testing Accuracy :", logisticRegression.score(x_test, y_test))
 # classification report
 cr = classification_report(y_test, y_pred_logisticRegression)
 print(cr)
```

Implementing Deep Learning Model

Based on the dataset I have decided to design the Deep Learning Neural Network with 3 layers along with one input layer and an output layer. I will be using Keras Sequential library to creates a Sequential model. I shall then be adding hidden layers using “dense” function with most of the layers using “relu” activation and layers I will also be adding dropout layers to account for any overfitting of the model.

```
def NueralNetworks(X_train, y_train, x_test, y_test,featureSize):
 model = Sequential()
 model.add(Dense( 128, input_dim=featureSize, activation='relu'))
 model.add(Dropout(0.30))
 model.add(Dense( 128, activation='relu'))
 model.add(Dropout(0.30))
 model.add(Dense( 256 ,activation='relu'))
 model.add(Dropout(0.30))
 model.add(Dense(128, activation='relu'))
 model.add(Dropout(0.30))
 model.add(Dense(1, activation='sigmoid'))

 # compile the keras model
 optimizer = keras.optimizers.Adamax(lr=0.015) #Default #0.01
 model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])
 # fit the keras model on the dataset
 model.fit(X_train,  y_train, epochs=50, batch_size=12)
 # evaluate the keras model
 #_, accuracy = model.evaluate(X_train,  y_train, verbose=0)
 _, accuracy = model.evaluate(x_test,  y_test, verbose=0)
 print('Accuracy: %.2f' % (accuracy*100))
 y_pred = model.predict(x_test)
 y_pred = y_pred > 0.5

 print("Confusion matrix:")
 print(confusion_matrix(y_test, y_pred))
```

Parameter Hyper Optimization

For optimizing the learning process to the dataset, I have chosen to implement the Random Search method using “RandomizedSearchCV” which performs Cross Validation on model Optimized model as well. It can discovery different hyperparameter combinations that you would be difficult to intuitively create by randomly sampling various setting of parameters. Since the algorithms that I will be implementing such as Random Forest has many parameters, it will be useful to have the computer generate the various combinations and find the best parameter combination. The best parameters based of Random Search for each of the ML algorithms are as follows.

Brief description of the main parameters for ensemble-based Machine learning algorithms such as Random Forest and Extra Tree.

n_estimators : Number of Decision trees to create

max_features : Number of features to consider at every split

max_depth : Maximum number of levels in tree

min_samples_split : Minimum number of samples required to split a node

min_samples_leaf : Minimum number of samples required at each leaf node

bootstrap : Method of selecting samples for training each tree

For non-ensemble based algorithms such as Logistic Regression

C: parameter controls the penalty strength

Solver: different solver have different levels of performance

Penalty: helps shrink the coefficients in the regression towards zero.

Algorithm and their Hyperparameter Optimized settings

Random forest: `{'n_estimators': 800, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'sqrt', 'max_depth': 20, 'bootstrap': False}`

Extra Trees: `{'n_estimators': 1600, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 80, 'bootstrap': True}`

Logistics Regression: `{'C': 3.223866243239879, 'penalty': 'l1', 'solver': 'liblinear'}`

Deep Learning: I have optimized artificial neural network parameters manually by fine tuning the model in an iterative process. GridSearch and RandomSearch have been computationally very demanding to process. I have chosen the manual fine-tuning approach by studying the ANN algorithm and documentations. However, I choice of parameters that I am using does not exhaustively consider every combination thus the parameters I have chosen may not be completely optimum.

5. Results and Evaluation

The main aim of the project was to achieve as high an accuracy as possible in predicting the customer purchase intention. Having gone through the background research and implementation of the solution as intended. I will now be looking at how each stage of the solution impacted the accuracy of the prediction among all four selected Machine Learning algorithm namely Random Forest, Extra Tree, Logistic Regression and Deep Learning Artificial Neural Network. I will also be summarizing the results and discussing the impact of data pre-processing and how it has had an impact on the performance of these algorithms.

Random Forest Test Summary

Test Accuracy **before** feature selection

```
Random Forest
Accuracy_CV: 0.937 (0.006)
Testing Accuracy : 0.8992901508429458
              precision    recall  f1-score   support

         0.0       0.95      0.93      0.94      1897
         1.0       0.67      0.73      0.70       357

    accuracy                           0.90      2254
   macro avg       0.81      0.83      0.82      2254
weighted avg       0.90      0.90      0.90      2254
```

Fig. I

Test Accuracy **after** feature selection

```
Random Forest
Accuracy_CV: 0.933 (0.006)
Testing Accuracy : 0.8975155279503105
              precision    recall  f1-score   support

         0.0       0.95      0.93      0.94      1897
         1.0       0.66      0.74      0.70       357

    accuracy                           0.90      2254
   macro avg       0.80      0.83      0.82      2254
weighted avg       0.90      0.90      0.90      2254
```

Fig. II

Test Accuracy **after** Hyperparameter Optimization

```
Random Forest
Testing Accuracy : 0.8966282165039929
              precision    recall  f1-score   support

         0.0       0.95      0.93      0.94      1897
         1.0       0.66      0.72      0.69       357

    accuracy                           0.90      2254
   macro avg       0.80      0.83      0.81      2254
weighted avg       0.90      0.90      0.90      2254
```

Fig. III

Accuracy Summary

Fig	Accuracy
I	89.92%
II	89.75%
III	89.66%

Looking at the test accuracy report of the Random Forest algorithm, the accuracy of the model dropped after feature selection. It went down by 0.002. This can be seen as an indication that the feature selection was not optimal for this model. Although features were provided by FRE feature selection method and the top 15 features were used to build the model, the performance has fallen instead although by a very small amount. However, after hyperparameter optimization the model accuracy was increased by 0.004. The cross-validation score and mean value can be seen above by the Accuracy_CV label in the image. The model has a very high Cross validation score indicating a very robust solution. For hyper parameter optimized model, cross validation is done during Hyper parameterization processing.

Extra Trees Test Summary

Test Accuracy **before** feature selection

```
ExtraTrees
Accuracy_CV: 0.955 (0.007)
Testing Accuracy : 0.8806566104702751
              precision    recall  f1-score   support

         0.0       0.92      0.94      0.93      1897
         1.0       0.63      0.59      0.61       357

    accuracy                           0.88      2254
   macro avg       0.78      0.76      0.77      2254
weighted avg       0.88      0.88      0.88      2254
```

Fig. I

Test Accuracy **after** feature selection

```
ExtraTrees
Accuracy_CV: 0.940 (0.006)
Testing Accuracy : 0.8895297249334516
              precision    recall  f1-score   support

         0.0       0.94      0.92      0.93      1897
         1.0       0.64      0.71      0.67       357

    accuracy                           0.89      2254
   macro avg       0.79      0.82      0.80      2254
weighted avg       0.90      0.89      0.89      2254
```

Fig. II

Test Accuracy **after** Hyperparameter Optimization

```
ExtraTrees
Testing Accuracy : 0.8917480035492458
              precision    recall  f1-score   support

         0.0       0.95      0.92      0.93      1897
         1.0       0.63      0.76      0.69       357

    accuracy                           0.89      2254
   macro avg       0.79      0.84      0.81      2254
weighted avg       0.90      0.89      0.90      2254
```

Fig. III

Accuracy Summary

Fig	Accuracy
I	88.06%
II	88.95%
III	89.17%

The Extra Tree performs like the Random Forest algorithm in its accuracy. It can be seen from the above data that Extra Tree algorithm model improved in accuracy after applying feature section and the models accuracy further improved. Ensemble based Machine Learning algorithm like Extra Trees and Random forest do not always get an increase in performance due to the data pre-processing stages of One Hot Encoding and feature scaling as can be seen in Random Forest. However, in the scale of Extra Tree algorithm the accuracy was increased after applying feature selection as well as after Hyperparameter optimization. The cross-validation score and mean value can be seen above by the Accuracy_CV label in the image. The model has a very high Cross validation score indicating a very robust solution. For hyper parameter optimized model, cross validation is done during Hyper parameterization processing.

Logistics Regression Test Summary

Test Accuracy **before** feature selection

```
LogisticRegression
Accuracy_CV: 0.825 (0.009)
Testing Accuracy : 0.831410825199645
              precision    recall  f1-score   support

         0.0       0.95      0.85      0.89      1897
         1.0       0.48      0.75      0.59       357

    accuracy                           0.83      2254
   macro avg       0.71      0.80      0.74      2254
weighted avg       0.87      0.83      0.85      2254
```

Fig. I

Test Accuracy **after** feature selection

```
LogisticRegression
Accuracy_CV: 0.825 (0.011)
Testing Accuracy : 0.8371783496007098
              precision    recall  f1-score   support

         0.0       0.95      0.85      0.90      1897
         1.0       0.49      0.76      0.60       357

    accuracy                           0.84      2254
   macro avg       0.72      0.81      0.75      2254
weighted avg       0.88      0.84      0.85      2254
```

Fig. II

Test Accuracy **after** Hyperparameter Optimization

```
logisticRegression
Testing Accuracy : 0.8717834960070985
              precision    recall  f1-score   support

         0.0       0.95      0.89      0.92      1897
         1.0       0.57      0.76      0.65       357

    accuracy                           0.87      2254
   macro avg       0.76      0.82      0.79      2254
weighted avg       0.89      0.87      0.88      2254
```

Fig. III

Accuracy Summary

Fig	Accuracy
I	83.14%
II	83.71%
III	87.17%

It can be seen from the test summary that the data pre-processing has improved the accuracy of the prediction model for logistic regression. The logistics regression model's accuracy improved after feature selection. The score improved by 0.006. Logistic regression gained the highest improvement through Hyperparameter Optimization where the accuracy was increase by 0.04. Logistic regression uses gradient descent as part of their learning model. Here having feature selection and scaling and help the model learn and predict better. However, It is also sometimes important to have the correct parameter to allow the algorithm suite dataset properly. This was done during the Hyperparameter Optimization stage which the resulted in an increase in accuracy by 4.5%.

Deep Learning Neural Network Test summary

Below is the confusion Matrix of the Deep Learning Solution's prediction.

Test results **before** feature Selection

```
Accuracy: 87.67
Confusion matrix:
[[1730  167]
 [ 111  246]]
```

Test results **after** feature selection

```
Accuracy: 88.64
Confusion matrix:
[[1709  188]
 [  68  289]]
```

After training the dataset using deep learning, the model has been able to produce an accuracy 88.64%. After applying feature section with Top 12 features according RFE method the accuracy of the model went up by approximately 1%. However, the model is not completely Hyperparameter optimized due to limitation in computing hardware and computational power needed for processing. Methods such as RandomSearch, GridSearch or Bayesian Optimization can be used to uncover the optimal parameter which would increase the accuracy over 89%.

Overall Summary

Performance ranking of each Algorithm on Prediction Customer Purchase Intention

Highest Accuracy achieved: 89.6% using **Random Forest**.

Machine Learning Algorithm	**Accuracy**
Random Forest	89.6%
Extra Trees	89.1%
Deep Learning using Artificial Neural Network	88.64%
Logistics Regression	87.1%

6.Conclusions

The project aim was to build a solution that can predict customer purchase intention with as high an accuracy as possible. The highest accuracy It has managed to achieve was **89.9% accuracy** . It was recorded as the highest accuracy by comparing and ranking the various model's performances with each other. It can be seen that Random Forest model performed the best among the various algorithms implemented. After testing the algorithm with various states of pre-processed data it can be concluded that different machine learning models would work better with different types of pre-processed data. I have shown the accuracy of the models based on different states of pre-processed at in the Results and Evaluation section. Comparing the results after the dataset had been cleaned for irregularities. The dataset was then transformed to suit the machine learning algorithms

6.Future work

The customer purchase intention prediction can be combined to the ecommerce website's product recommendation system. Further work can be done to see if recommending products based on customer intension could have an impact on increases sales or not. I.e. Does recommending discounts and special deals to customer who have no intention to purchase choose to buy instead? This can also be used to recommend more expensive or higher quality products to a customer if they already have a strong intention to purchase a similar product. As it is likely they would be open to consider other more expense alternative product . The Artificial Neural Network model I built has potential to be Hyperparameter optimized further. Perhaps, in the future if there are more hardware capabilities then one can perform a RandomSearch , GridSearch or Bayesian Optimization with granular parameter intervals to obtain best hyperparameter results. There is also the potential of implementing reinforcement learning to solve this problem and researching how Reinforcement Learning solution compares to Artificial Neural Networks and other algorithms used in this report. It would also be useful to collect and perform the research on a larger dataset by which algorithm performance can be measured better because of better training with a larger dataset.

7.Reflection

My experience of undertaking this dissertation has allowed me to gain deep insight into Machine Learning and Machine Learning concepts. I have gained new skills and understanding of various python libraries. Undertaking this project has also given me an insight into the life of research and development. The problem required reading and researching various information on Machine Learning Algorithms and learning and applying those concepts. One of such researches included researching and studying Artificial Neural Networks and Deep Learning which was an interesting and fulling experience. Although I am happy with my approach to research and achieving the set project aims, I believe there were still many more Machine Learning Algorithms I could have further researched on and tested it on the dataset. For Example, applying a Reinforcement Learning technique would been an interesting solution worth researching. Going forward, I would make sure to allocate sufficient time to deeper study and research of newer concepts within my future area of research to develop more cutting-edge solutions. Overall, I have enjoyed my time studying a new field of computer science that is currently being used in every industry extensively. My experience from this machine learning project will help me to develop more interesting Machine Learning solutions in future having learnt necessary skill from programming using python's extensive ML libraries to developing a research focused mind set.

References

[1] Stuart J. Russell, Peter Norvig (2010) Artificial Intelligence: A Modern Approach, Third Edition, Prentice Hall ISBN 9780136042594.

[2] Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar (2012) Foundations of Machine Learning, The MIT Press ISBN 9780262018258.

[3] https://towardsdatascience.com/a-brief-introduction-to-supervised-learning-54a3e3932590 [Accessed 11/01/2021]

[4] https://www.ibm.com/cloud/learn/unsupervised-learning [Accessed 16/02/2021]

[5] Roman, Victor (2019-04-21). "Unsupervised Machine Learning: Clustering Analysis". Medium. Retrieved 2019-10-01.

[6] https://towardsdatascience.com/unsupervised-learning-and-data-clustering-eeecb78b422a

[7] https://courses.cs.duke.edu//fall03/cps260/notes/lecture18.pdf [Accessed 16/02/2021]

[8] Tolles, Juliana; Meurer, William J (2016). "Logistic Regression Relating Patient Characteristics to Outcomes"

[9] Greene, William H. (2012). Econometric Analysis (Seventh ed.). Boston: Pearson Education. pp. 803–806. ISBN 978-0-273-75356-8.

[10] https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/

[11] Geurts, Pierre, Damien Ernst, and Louis Wehenkel. "Extremely randomized trees." Machine learning 63.1 (2006): 3-42.

[12]Intelligent System Design Proceedings of Intelligent System Design: INDIA 2019 By Suresh Chandra Satapathy, Vikrant Bhateja, B. Janakiramaiah, Yen-Wei Chen · 2020. pp. 130-131

[13] Y. Ku and Y. Tai, "What Happens When Recommendation System Meets Reputation System? The Impact of Recommendation Information on Purchase Intention," 2013 46th Hawaii International Conference on System Sciences, 2013, pp. 1376-1383, doi: 10.1109/HICSS.2013.605.

[14] Mitchell, Tom (1997). Machine Learning. New York: McGraw Hill. ISBN 0-07-042807-7. OCLC 36417892.

[15] Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. Artificial Intelligence in Design '96. Springer, Dordrecht. pp. 151–170. doi:10.1007/978-94-009-0279-4_9.

[16] https://scikit-learn.org/stable/getting_started.html [Accessed 16/02/2021]

[17] https://imbalanced-learn.org/stable/ [Accessed 16/02/2021]

[18]https://seaborn.pydata.org/#:~:text=Seaborn%20is%20a%20Python%20data,can%20read%20the%20introductory%20notes. [Accessed 7/03/2021]

[19] https://contrib.scikit-learn.org/category_encoders/ [Accessed 12/03/2021]

[20] Grossman, Robert; Seni, Giovanni; Elder, John; Agarwal, Nitin; Liu, Huan (2010). "Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions". Synthesis Lectures on Data Mining and Knowledge Discovery. Morgan & Claypool. **2**: 1–126.

[21] https://www.ibm.com/cloud/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks [Accessed 2/05/2021]

[22] https://www.ibm.com/cloud/learn/deep-learning [Accessed 2/05/2021]