1. INTRODUCTION

1.1. Motivation:

How extreme would users go in deciding between buying from an e-commerce company (aka online shopping) or a physical shop? Recently, technology has been a part of most aspects of life, and businesses are not an exception. The main aims and motivations of e-commerce businesses are to harness the advancement in technology, especially the Internet, to reduce expenses and make shopping more comfortable for consumers to thrive the business. E-commerce businesses are faced with the problem of not being able to reliably predict the intention of visitors to their websites. He, Lu and Zhou found that one major barrier hindering the growth of e-commerce is the inability to effectively determine the intention to purchase.

1.2. PROBLEM DEFINITION

Unlike in the traditional brick and mortar setting, where shop attendants can have physical interactions to know reasons why visitors did not purchase particular goods, the e-commerce presence leaves businesses no idea why transactions never took place. According to prior research, the nature of e-commerce websites with regards to navigation, content and simplicity of use determines how useful these websites appear to customers. The rate of usefulness plays a vital role in convincing customers in deciding whether or not to proceed with a transaction. Owing to different marketing strategies that have been adopted by the e-commerce industry, advertisements reach thousands of customers and hence websites are able to drive a lot of traffic. On the marketing point of view, this initiative is successful since numerous customers are attracted. However, from the business perspective, unless the website visitors make actual purchases or transactions, their goal is not yet achieved.

1.3. OBJECTIVE OF PROJECT

The introduction of data science techniques (including machine learning) has been embraced in most decision-making processes. Several machine learning (ML) models have been developed to forecast the shopping intention of consumers within the e-commerce space. The ML algorithms used in these models include support vector machines (SVM), random forest and decision tree.

However, it is realized that most of the dataset used is biased towards false revenue which was a reason for accuracy levels falling between 82% and 89% for all the models used.

This project uses a large dataset from University of California Irvin (UCI) machine learning online repository, pre-processes the data using one-hot encoding and label encoding, builds an effective ML model using feature importance technique in the sklearn's random forest classifier to select the top 15 features. Final training is performed and compares the accuracy of the random forest model with that of naïve bayes classifier, Extra Trees Classifier, and other ensemble algorithms as done in previous research.

The primary objective of this research is to develop a high scoring random forest algorithm for e-commerce businesses to predict shoppers' purchase intention. The objective is achieved through the three related sub-objectives:

1. Identify existing weaknesses in the random forest algorithm for predicting purchase intention.
2. Train a random forest model using same data as in Objective 1 to provide better decisions and compare the results with that of random forest algorithm, Extra Trees Classifier and Naïve Bayes algorithms.
3. Identify basic factors that influence actual purchase intention based on the model.

1.4. LIMITATIONS OF PROJECT

The drawback of such a project is that, we need huge amount of data for proper accurate prediction values. If the data is small and inconsistent the prediction might not be completely reliable, and it may affect the firm's business model. Also, the final prediction is always reliable in an ideal market environment, as the e-commerce market is an unpredictable and unprecedented area, too many anomalies could cause the algorithm to output faulty values, which would be hazardous to the firm.

2. LITERATURE SURVEY

2.1 INTRODUCTION

Online shopping intention became a crucial concern to e-commerce business owners in 2000 when the Internet became a major infrastructure. There are a number of proposals addressing how e-commerce businesses can track customers' logs and use them to reliably predict the likelihood of purchase. Different studies have introduced diverse ideologies and perspectives in techniques for predicting decisions of shoppers based on patterns from shopping behavior.

2.2 EXISTING SYSTEM

2.2.1. Features of E-commerce that Affect Purchase Decision

The theory of buyer behavior shows a repetitive nature of shopping pattern taking into consideration the exogenous variables through the traditional brick and mortar mode of selling. Although behavioral patterns might be similar in both traditional and e-commerce businesses, other factors which include trust and security relating to the Internet also affect the rate of purchase in e-commerce businesses. The theory concluded that factors such as the price of items, unavailability of needed products, pressure, financial capabilities of customers and societal traits are the major inhibitory behaviors of online shoppers which could affect their decisions in the presence of e-commerce-based challenges.

First, focus is placed on the physical design features of e-commerce websites and their interactivity with consumers, an aspect of human-computer interaction. This design features covers usability, information quality, platform quality, service quality and playfulness. Whether customers visit a website just to "window shop" or to look for something specific, their stay must be made simple and effective. It could be either suggesting similar items, use of corrective searches other than an error message due to e.g., wrongful spellings or by categorizing products. Owing to this, it is not prudent to use primitive data to predict current purchase trends but could be a way of providing comparative analysis between both mediums. Gupta [14] applied neural networks to predict the likelihood of purchase by a visitor taking into consideration the prices. Gupta's algorithm was improved by Kukar-Kinney and Angeline[9] by extending it to online shopping which was not focused on only pricing but also other factors such as the loading time, ease of use and design which is now of key concern to shoppers. Next, empirical studies on how the design features and business models can be intertwined to create an experience that would lead to purchase decisions is considered. How well do consumers trust and have confidence in doing business at these websites?

2.2.2. Factors Influencing Consumer Purchase

Scientists focused on the dynamic pricing of goods and services by enabling e-commerce supply vendors to provide optimum prices for goods based on the current competition. The model used logistic regression to predict the optimal price of goods and services with four clusters of eight main features which had a significance level between 0.01 and 0.05. Using K-means clustering algorithm, the coefficient of the variation was calculated as 83% that indicates a large area of the data being used. Pricing of items is a deciding factor in all business models. Higher online prices compared to the physical market could reduce the purchases from e-commerce sites. These researches outlined the factors that affect the decision of purchase which later also highlighted benefit and risk perspective and behavioral pattern trend as the theory of reasoned action which could lead to the abortion of online transactions.

The mental and societal/environmental behaviors, such as impatience and frustration, with which people go through the shopping and check-out process of e-commerce sites were identified. These researches outlined the factors that affect the decision to purchase which later also highlighted benefit and risk perspective and behavioral pattern trend as the theory of reasoned action which could lead to the discontinue of online transactions.

Existing model for such projections is built using naïve bayes or decision trees. User behaviors are extracted quantitatively and then the purchase prediction model is built, then recommended system model is finalized to find out which product could be purchased. Based on the obtained user behaviors and massive data, we apply collaborative filtering-based method to recommend items for different consumers, and predict whether purchase will happen.

2.3 DISADVANTAGES OF EXISTING SYSTEM

1. The main factor for quantitatively assessing any prediction model is the Accuracy level for that model. And these models generate comparatively low accuracy score, which could be considerably improved.
2. It is not possible to build models using data sets which contain huge number of attributes due to limitations of the algorithms used in existing system.
3. The final prediction generated by these algorithms is not completely reliable.

2.4 PROPOSED SYSTEM

2.4.1. Modern ML Models

Sakar et al. [15] identified under sampling of the dataset as an effect on previous models used. Oversampling and under sampling are very common in most machine learning algorithms and this leads to higher accuracies but with high rates of false positives and true negatives, respectively. Therefore, to deal with oversampling of the dataset, the paper introduced more negative decision data to the original dataset. Also, due to a large number of features in the dataset used, a feature selection method was used to reduce the original features captured in the data. This was tested on a SVM, decision tree and multi-layer perceptron classifiers. The output revealed that, although SVM and decision tree had fair accuracies, the multi-layer perceptron had a good balance of accuracy and F1-score than both the SVM and decision tree.

Access to a large dataset, which is not readily available, led to a predictive model on fewer data with different features for the prediction of purchasing behavior of online consumers. The paper suggested a single feature with high accuracy for determining the purchase intention of consumers.

In the proposed system we aim to use the Ensemble learning methods which include Random Forest classifier and Extra Trees Classifier. The reason of using these Ensemble methods is specifically to overcome the problem of high-variance in Decision Trees and other classification algorithms.

At a high-level, in *pseudo-code*, Random Forests algorithm follows these steps:

1. Take the original dataset and create *N* bagged samples of size *n*, with *n* smaller than the original dataset.
2. Train a Decision Tree with each of the *N* bagged datasets as input. But, when doing a node split, don't explore all features in the dataset. Randomly select a smaller number, *M* features, from all the features in training set. Then pick the best split using impurity measures, like Gini Impurity or Entropy.
3. Aggregate the results of the individual decision trees into a single output.
4. Average the values for each observation, produced by each tree, if you're working on a Regression task.
5. Do a majority vote across all trees, for each observation, if you're working on a Classification task.

All the advantages of Decision Trees, but more powerful

At the core of this algorithm is a Decision Tree so, Random Forests shares all its advantages.

It's a data robust algorithm, being able to handle different types of data, and doesn't require any data pre-processing.

The true potential of Random Forests comes from combining the results different Decision Trees.

3. ANALYSIS

3.1 Introduction

This project reports a classifier model built with two random forest algorithms to predict the purchase intentions of consumers who visit e-commerce websites. The model is built using Python and its libraries numpy, pandas, sklearn and matplotlib. Data on online shoppers' activities derived from Google Analytics was downloaded from the UCI machine learning repository by Sakar et al. [15] for the model building. To facilitate the learning of the algorithm, we started with a large amount of data and then extracted important features that are relevant to predicting shoppers' intention. Then, the random forest algorithm was used to predict the purchase intention of consumers.

3.2 Hardware and Software Requirement

3.2.1 Hardware specifications:

- Minimum of 1 GB RAM
- At least 10 GB of Hard Disk space
- Intel Processor
- LAN or Wi-Fi connetivity

3.2.2 Software specifications:

- Windows or Linux or MAC
- Python GUI or Anaconda Navigator

3.2.3 System Requirements

- Windows 7 / Windows XP

3.3 System Architecture

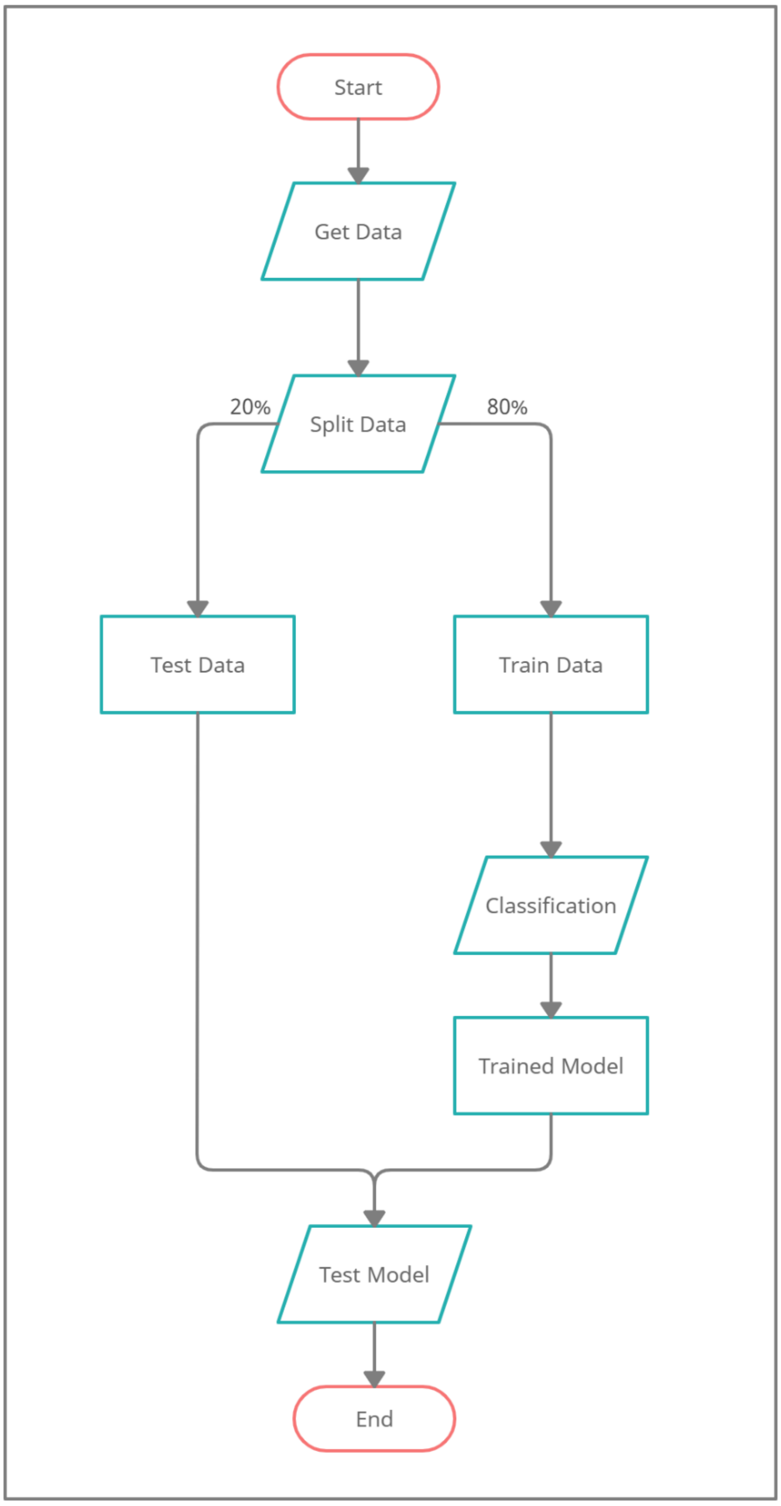


Fig. 3.1. system architecture

4. DESIGN

4.1. Introduction

4.1.1. Machine Learning System Design

The process of defining an interface, algorithm, data infrastructure, and hardware for ML Learning system to meet specific requirements of reliability, scalability, maintainability, and adaptability.

4.1.2. Reliability

The system should be able to perform the correct function at the desired level of performance under a specified environment. Testing for the reliability of ML systems where learning is done from data is tricky as its failure does not mean it will give an error, it may simply give garbage outputs i.e. it may produce some outputs even though it does not have seen that corresponding ground truth during training.

4.1.3. Scalability

As the system grows (in data volume, traffic volume, or complexity), there should be reasonable ways of dealing with that growth. There should be automatic provision to scale compute/storage capacity because for some critical applications even 1 hour of downtime/failure can cause in loss of millions of dollars/credibility of the app.

4.1.4. Maintainability

Over the time period, data distribution might get change, and hence model performance. There should be a provision in the ML system to first identify if there is any model drift/data drift, and once the significant drift is observed, then how to re-train/re-fresh and enable updated ML model without disturbing the current operation of the ML system.

4.1.5. Adaptability

Other changes that most frequently comes in the ML systems are the availability of new data with added features or changes in business objectives e.g. conversion rate vs engagement time from customers for e-commerce. Thus, the system should be flexible to quick updates without any service interruptions.

4.2. UML DIAGRAMS

4.2.1. ACTIVITY DIAGRAM

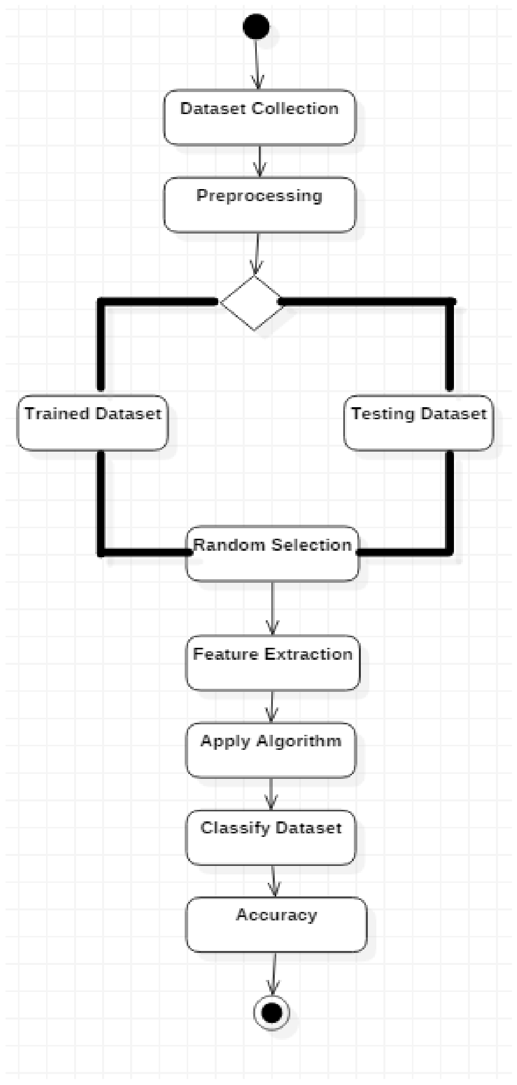


Fig. 4.1 Activity diagram

4.2.2 Class Diagram:

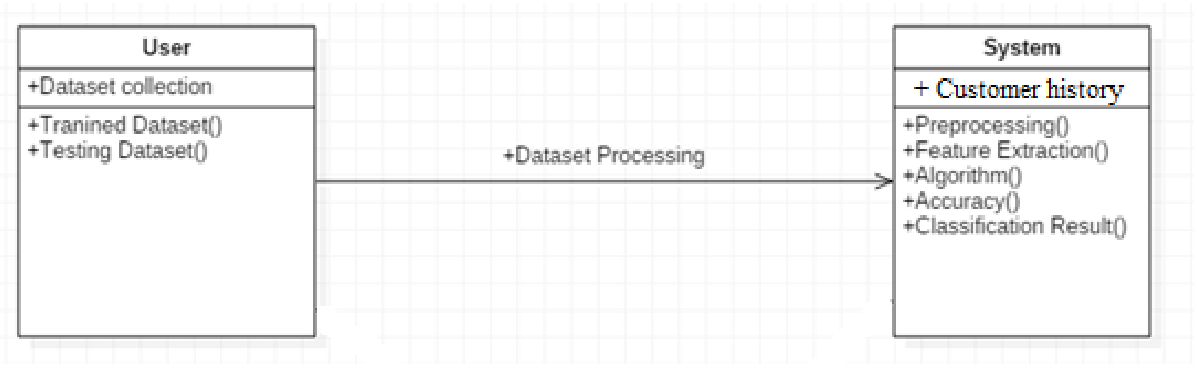


Fig 4.2 Class diagram

4.2.3 Use Case Diagram:

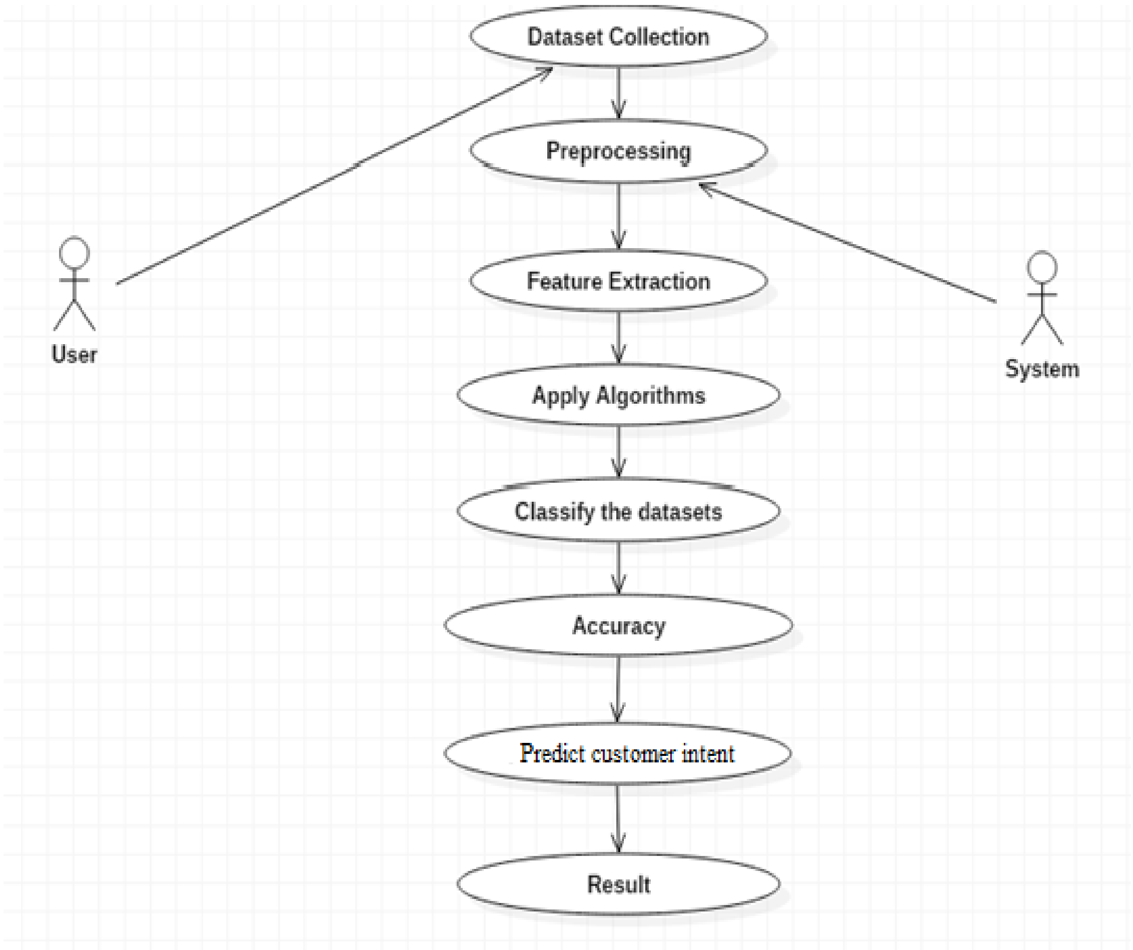


Fig. 4.3 Use Case Diagram

4.2.4 Sequence Diagram:

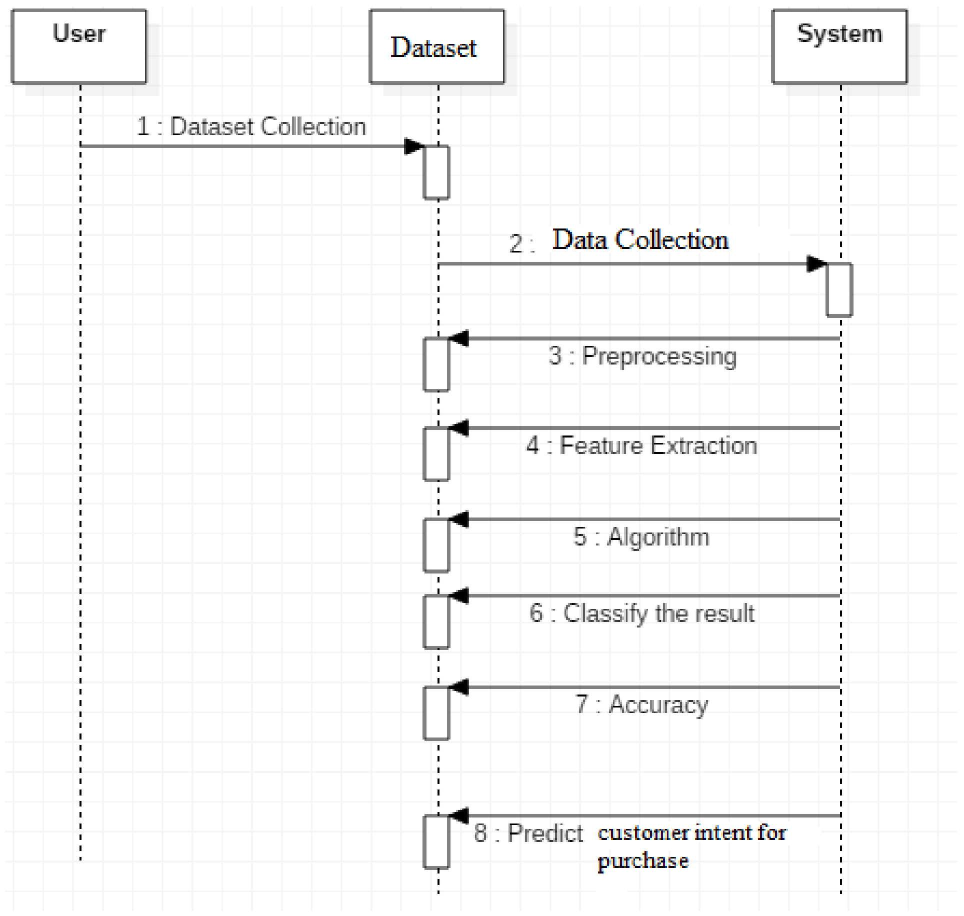


Fig 4.4 Sequence diagram - 1

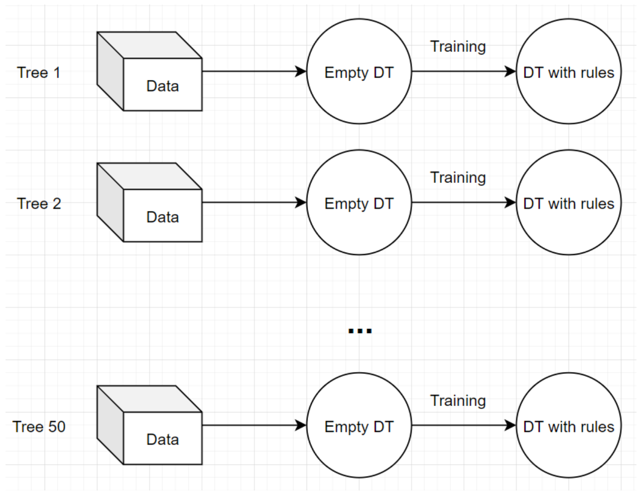


Fig 4.5 Sequence diagram - 2

4.2.5. Data Flow Diagrams

LEVEL 0

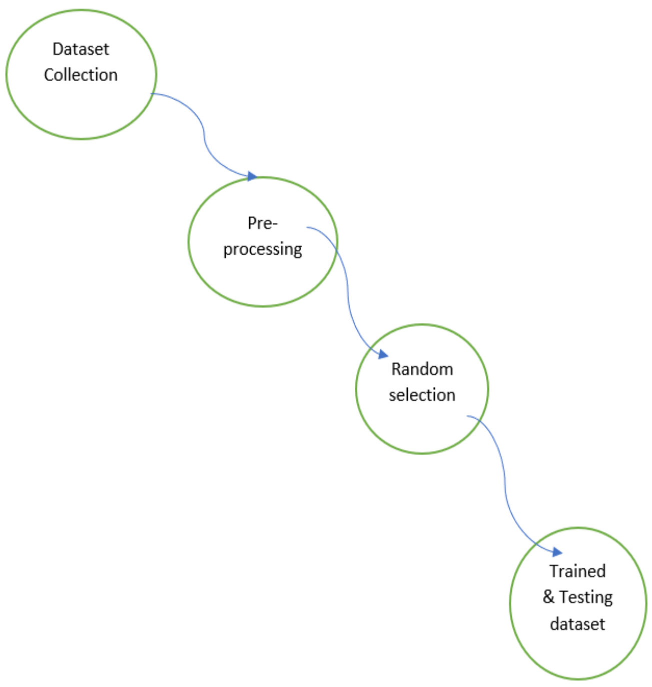


Fig 4.6 Level-0 Data Flow Diagram

LEVEL 1

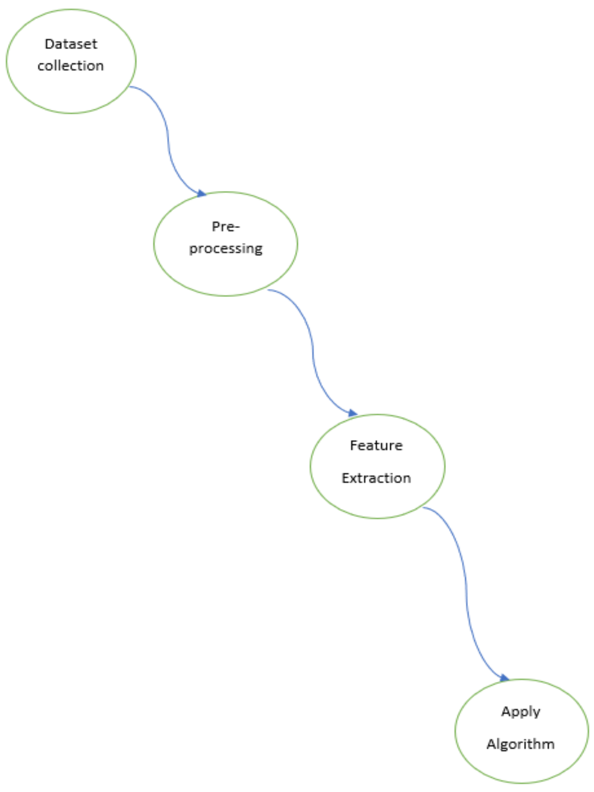


Fig 4.7 Level-1 Data Flow Diagram

LEVEL 2

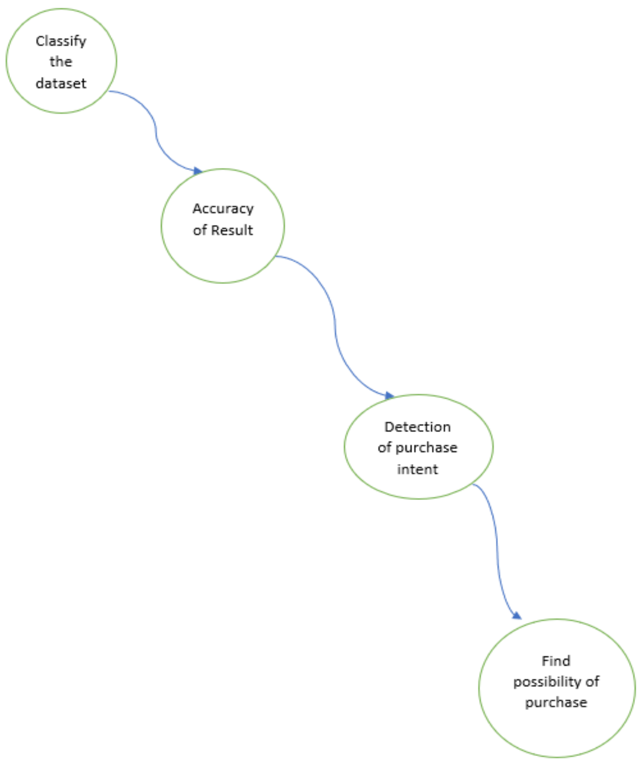


Fig 4.8 Level-2 Data Flow Diagram

5. MODULE DESIGN

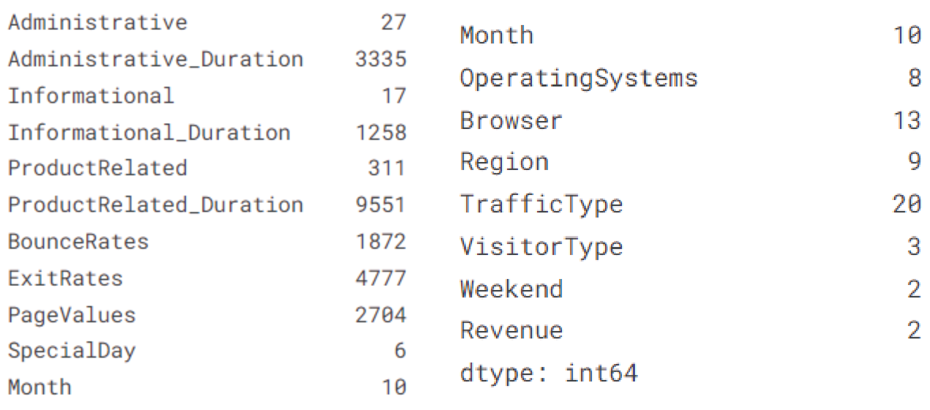
5.1. Introduction

The UCI dataset contains 12,330 rows of data from users of a popular e-commerce site, Amazon. The dataset contains their visits to the Amazon website and the navigations made through the site to the decision at the end of the visit. The full dataset was randomly split into a 70/30 per cent train to test ratio respectively. The dataset contained 18 features which were gathered from each user visiting the page. An overview of the structure of the raw dataset obtained is shown in Table 1 with detailed explanation to the headers in Table 2. Out of the 18 initial features, feature selection using the random forest classifier (feature importance model) was performed to reduce the features to be used by the algorithm to 15. It was observed that the purchasing intention is imbalanced which is common for online consumers to not make a purchase of any product during most visits to their shopping sites.

5.2. Data Collection

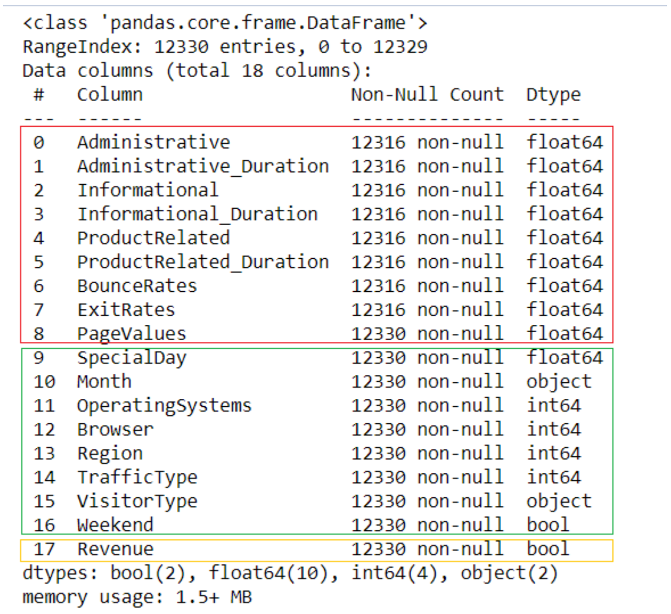
Dataset Description:

The dataset consists of feature vectors belonging to 12,330 sessions. The dataset was formed so that each session would belong to a different user in a 1-year period to avoid any tendency to a specific campaign, special day, user profile, or period. Of the 12,330 sessions in the dataset, 84.5% (10,422) were negative class samples that did not end with shopping, and the rest (1908) were positive class samples ending with shopping.



```
Administrative                27
Administrative_Duration     3335
Informational                 17
Informational_Duration      1258
ProductRelated               311
ProductRelated_Duration     9551
BounceRates                 1872
ExitRates                   4777
PageValues                  2704
SpecialDay                     6
Month                         10
```

```
Month                 10
OperatingSystems       8
Browser               13
Region                 9
TrafficType           20
VisitorType            3
Weekend                2
Revenue                2
dtype: int64
```



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12330 entries, 0 to 12329
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Administrative           12316 non-null  float64
 1   Administrative_Duration  12316 non-null  float64
 2   Informational            12316 non-null  float64
 3   Informational_Duration   12316 non-null  float64
 4   ProductRelated           12316 non-null  float64
 5   ProductRelated_Duration  12316 non-null  float64
 6   BounceRates              12316 non-null  float64
 7   ExitRates                12316 non-null  float64
 8   PageValues               12330 non-null  float64
 9   SpecialDay               12330 non-null  float64
 10  Month                    12330 non-null  object
 11  OperatingSystems         12330 non-null  int64
 12  Browser                  12330 non-null  int64
 13  Region                   12330 non-null  int64
 14  TrafficType              12330 non-null  int64
 15  VisitorType              12330 non-null  object
 16  Weekend                  12330 non-null  bool
 17  Revenue                  12330 non-null  bool
dtypes: bool(2), float64(10), int64(4), object(2)
memory usage: 1.5+ MB
```

Fig 5.1 dataset collection

5.3. Data Pre-Processing

Organizing the selected data by formatting, cleaning and sampling from it.

Three common data pre-processing steps are:

- Formatting: The data you have selected may not be in a format that is suitable for you to work with. The data may be in a relational database and you would like it in a flat file, or the data may be in a proprietary file format and you would like it in a relational database or a text file.
- Cleaning: Cleaning data is the removal or fixing of missing data. There may be data instances that are incomplete and do not carry the data you believe you need to address the problem. These instances

may need to be removed. Additionally, there may be sensitive information in some of the attributes and these attributes may need to be anonymized or removed from the data entirely.

- Sampling: There may be far more selected data available than you need to work with. More data can result in much longer running times for algorithms and larger computational and memory requirements. You can take a smaller representative sample of the selected data that may be much faster for exploring and prototyping solutions before considering the whole dataset.

5.3.1. Feature Cleaning

We will try classification prior to any feature engineering to see a baseline classification based on our features. In order to prepare our data, we need to switch our labels to the correct format. We have a few features that need to be adjusted.

First, we drop the month column. The 'Month' column only has 10 unique types, which indicates that it is missing two months of data. Each month has varying numbers of entries, which could unfairly bias our data to prefer classification by month. We can see below the distribution of each month in the 'Month' column. Additionally, time-sensitivity is already contained in the 'Special Day' column, which influences buying decision, so the month column is slightly redundant.

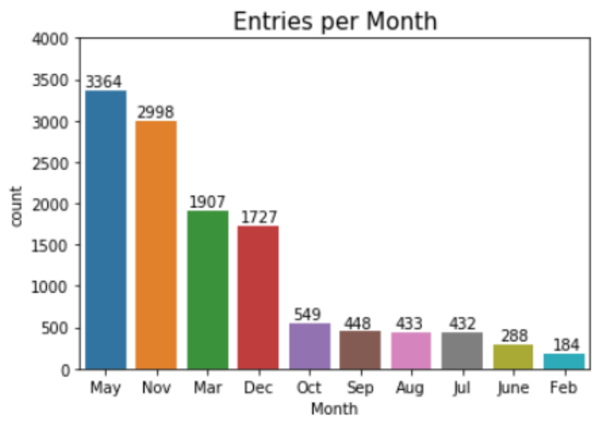


Fig 5.2 month data

We can see here that the 'Month' column is missing January and April. We can see visually that several months have many samples (May, Nov) and a couple have very few samples (Feb, June). We will remove this column.

Next, let us take a look at the 'Operating Systems' and 'Browser' columns.

Here we have the Operating Systems, labelled by number. Low-usage browsers have been consolidated into label '5'. We can see that a majority of users use operating system #2. Operating systems can indicate users of a sepcifc type of computer and may portray certain user archetypes (Windows users, Mac users, Linux users). For now, we will forgo usage of this column for our classifier.

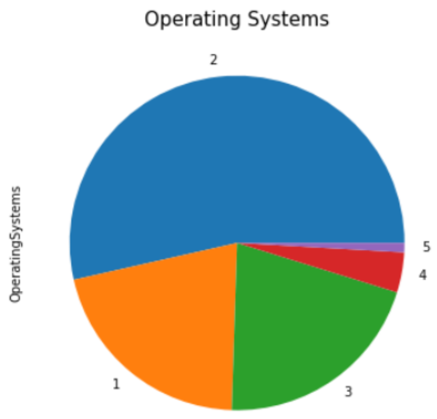


Fig 5.3 OS data

Here we have the Operating Systems, labelled by number. Low-usage browsers have been consolidated into label '5'. We can see that a majority of users use operating system #2. Operating systems can indicate users of a specific type of computer and may portray certain user archetypes (Windows users, Mac users, Linux users). For now, we will forgo usage of this column for our classifier.

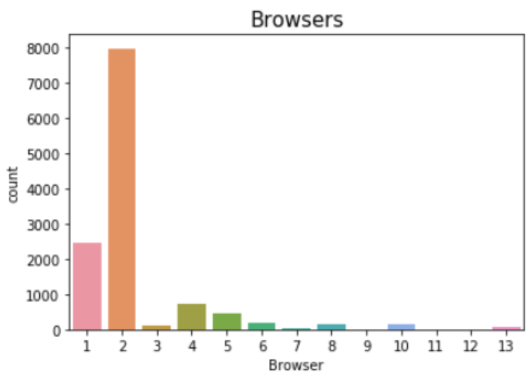


Fig 5.4 browser data

Browser choice is even more polarizing than Operating System. Here we see that a large majority of users use browser 2, with a smaller number of users using browser 1. All other browsers represent a small subsection of online users. We will not use this as it does not contribute much to our model.

There are several other columns that we leave out:

'Region': We leave regionality out because the regionality may be slightly tied to purchase likelihood, but we want to train our model on a smaller set of features if possible.

'Traffic Type': We leave this column out because Traffic sources are not quite useful for classifying if a user will make a purchase. It usually aids website owners in gauging traffic sources and can assist with determining where they should invest in advertisement.

'Weekend': There is weak correlation between days of the week and online shopping. https://blog.workarea.com/trends-when-do-people-shop-online asserts that Sundays and Mondays have the highest traffic for eCommerce, but only by 16% of weekly revenue, and mostly on Monday, which is not classified as a weekend.

5.3.2. Label and One Hot Encoding

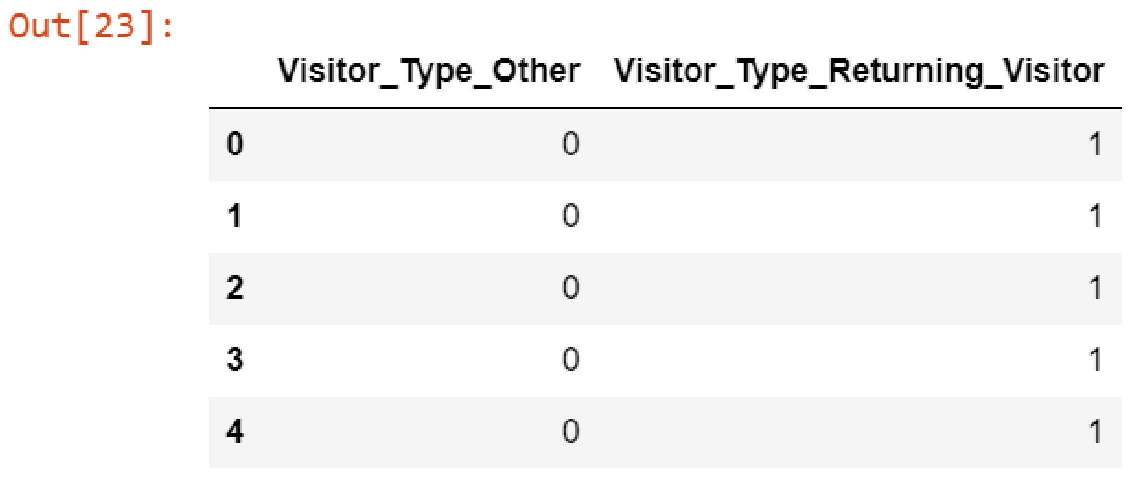
In order to properly fit our classifier, we will need to encode our string variables to integer labels, then convert our labels from integers to One Hot columns to remove any implied hierarchy.

Label Encoding

Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering.

One-Hot Encoding

One-Hot Encoding is another popular technique for treating categorical variables. It simply creates additional features based on the number of unique values in the categorical feature. Every unique value in the category will be added as a feature.



```
Out[23]:
```

	Visitor_Type_Other	Visitor_Type_Returning_Visitor
0	0	1
1	0	1
2	0	1
3	0	1
4	0	1

Fig 5.5 label and one-hot encoding

5.4. Feature Extraction

Next thing is to do Feature extraction is an attribute reduction process. Unlike feature selection, which ranks the existing attributes according to their predictive significance, feature extraction actually transforms the attributes. The transformed attributes, or features, are linear combinations of the original attributes. Finally, our models are trained using Classifier algorithm. We use classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered. The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Random Forest. These algorithms are very popular in text classification tasks.

Some selected columns from the data are used to create the machine learning model since there are some features which are irrelevant to the model or do not have an impact on the revenue introduced in this research. The random forest classifier filter-based feature selection method which is focused on the feature importance other than the statistical coefficient was implemented. Sklearn's

SelectKBest was used to find the top scoring features in the processed dataset. The initial dataset had 18 features (as explained in Table 2) which were increased to 57 after preprocessing using one-hot encoding to affect features such as the Operating System, Month, Browser, Region and VisitorType. Feature selection then reduced the number of columns from the 57 in the dataset to 15 relevant features. This included Informational, Informational_Duration, ProductRelated, ProductRelated_ Duration, Administrative, Administrative_Duration, Boune Rates, ExitRates, PageValues, SpecialDay, Month (December), TrafficType, VisitorType, Region, and Weekend. The final selected features were also based on the assumptions made in the Section 1 and the classification method used (i.e. random forest) in this research.

5.5. CLASSIFICATION

5.5.1.Naïve Bayes Classifier:

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, all of these properties independently contribute to the probability that this fruit is an apple and that is why it is known as 'Naive'.

Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. Look at the equation below:

$$\underbrace{P(c\mid x)}_{\text{Posterior Probability}} = \frac{\overbrace{P(x\mid c)}^{\text{Likelihood}}\,\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$$

$$P(c\mid \mathbf{X}) = P(x_1\mid c)\times P(x_2\mid c)\times\cdots\times P(x_n\mid c)\times P(c)$$

Above,

- *P*(*c*|*x*) is the posterior probability of *class* (c, *target*) given *predictor* (x, *attributes*).
- *P*(*c*) is the prior probability of *class*.
- *P*(*x*|*c*) is the likelihood which is the probability of *predictor* given *class*.
- *P*(*x*) is the prior probability of *predictor*.

4 Applications of Naive Bayes Algorithms

- **Real time Prediction:** Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- **Multi class Prediction:** This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- **Text classification/ Spam Filtering/ Sentiment Analysis:** Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)

- **Recommendation System:** Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not

5.5.2.Random Forest Classifier:

Introduction

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems.

Steps involved in random forest algorithm:

Step 1: In Random Forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on Majority Voting or Averaging for Classification and regression respectively.

Important Features of Random Forest

- Diversity- Not all attributes/variables/features are considered while making an individual tree, each tree is different.
- Immune to the curse of dimensionality- Since each tree does not consider all the features, the feature space is reduced.
- Parallelization-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.
- Train-Test split- In a random forest we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.
- Stability- Stability arises because the result is based on majority voting/ averaging.

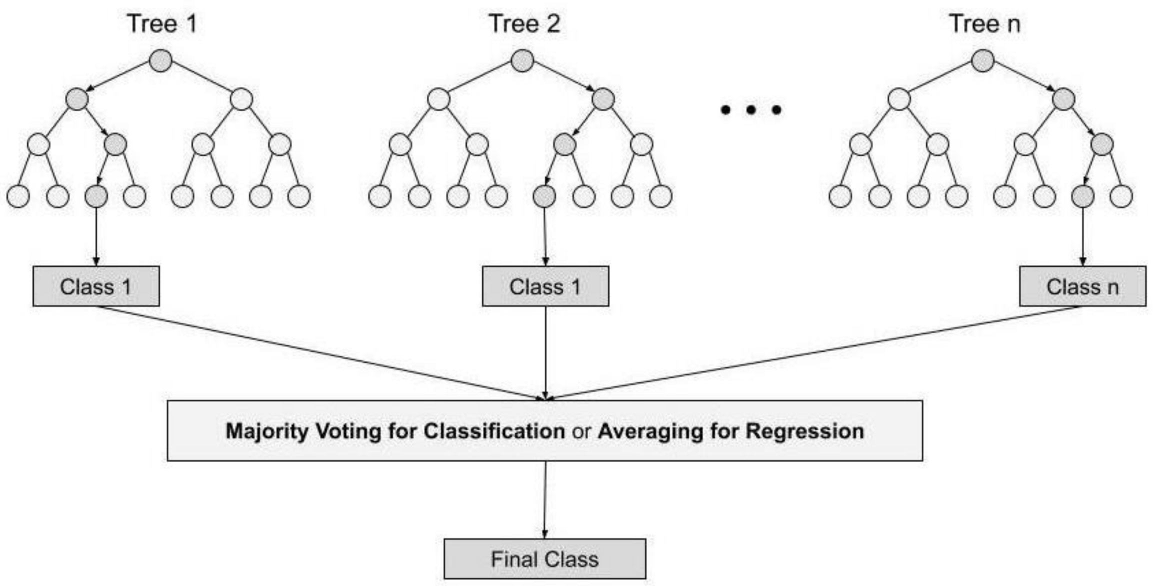

Tree 1
Tree 2
Tree n
Class 1
Class 1
Class n
Majority Voting for Classification or Averaging for Regression
Final Class

Fig 5.6 working of random forest

Difference between Decision Tree & Random Forest

Random forest is a collection of decision trees; still, there are a lot of differences in their behaviour.

Decision trees	Random Forest
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data and the final output is based on average or majority ranking and hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree it will formulate some set of rules to do prediction.	3. Random Forest randomly selects observations, builds a decision tree and the average result is taken. It doesn't use any set of formulas.

Thus, random forests are much more successful than decision trees only if the trees are diverse and acceptable.

Important Hyperparameters

Hyperparameters are used in random forests to either enhance the performance and predictive power of models or to make the model faster.

Following hyperparameters increases the predictive power:

- n_estimators– number of trees the algorithm builds before averaging the predictions.
- max_features– maximum number of features random forest considers splitting a node.
- mini_sample_leaf– determines the minimum number of leaves required to split an internal node.

Following hyperparameters increases the speed:

- n_jobs– it tells the engine how many processors it is allowed to use. If the value is 1, it can use only one processor but if the value is -1 there is no limit.
- random_state– controls randomness of the sample. The model will always produce the same results if it has a definite value of random state and if it has been given the same hyperparameters and the same training data.
- oob_score – OOB means out of the bag. It is a random forest cross-validation method. In this one-third of the sample is not used to train the data instead used to evaluate its performance. These samples are called out of bag samples.

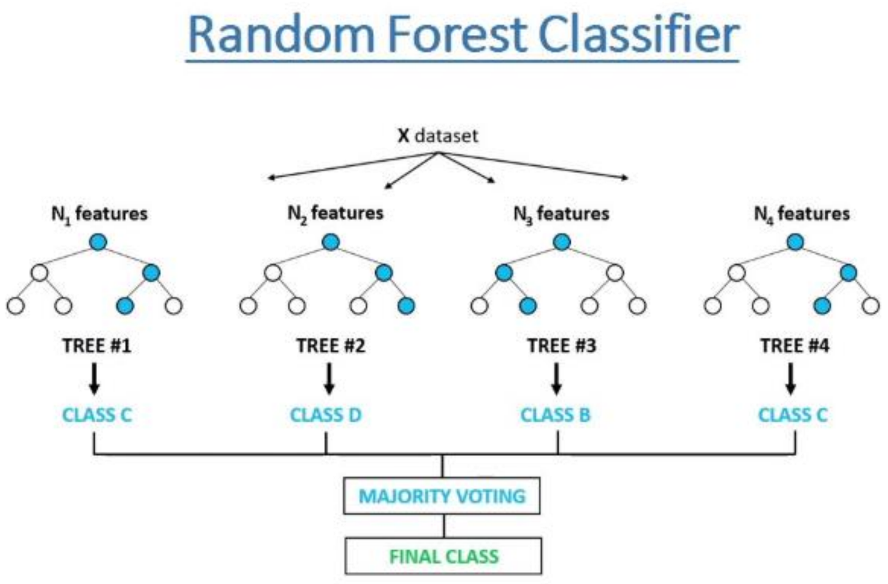


Fig 5.7 random forest classification

5.5.3. Extra Trees Classifier:

Similar to a Random Forest classifier we have the Extra Trees classifier — also known as Extremely Randomized Trees. To introduce more variation into the ensemble, we will change how we build trees.

Each decision stump will be built with the following criteria:

- All the data available in the training set is used to built each stump.
- To form the root node or any node, the best split is determined by searching in a subset of randomly selected features of size sqrt(number of features). The split of each selected feature is chosen at random.
- The maximum depth of the decision stump is one.

Notice that in an Extra Trees classifier, the features and splits are selected at random; hence, "Extremely Randomized Tree". Since splits are chosen at random for each feature in the Extra Trees Classifier, it's less computationally expensive than a Random Forest.

The Extra Trees classifier performed similarly to the Random Forest. However, there are performance differences such as, Decision Trees show high variance, Random Forests show medium variance, and Extra Trees show low variance.

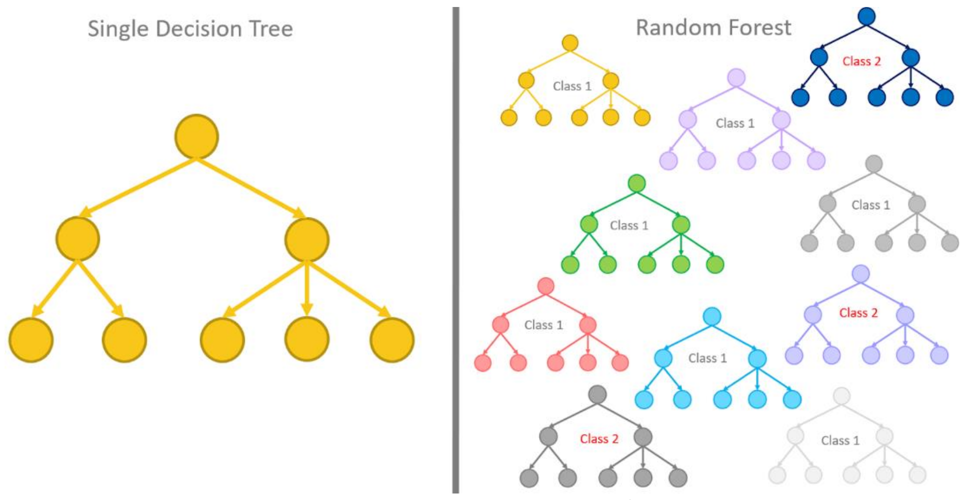


Fig 5.8 decision tree vs random forest

6. Implementation and Results

6.1. Introduction

The implementation of the following frame work can help with the advancement of CRM which is Customer Relationship Management:

Customer relationship management (CRM) is a technology for managing all your company’s relationships and interactions with customers and potential customers. The goal is simple: Improve business relationships. A CRM system helps companies stay connected to customers, streamline processes, and improve profitability.

When people talk about CRM, they are usually referring to a CRM system, a tool that helps with contact management, sales management, productivity, and more.

A CRM solution helps you focus on your organisation’s relationships with individual people — including customers, service users, colleagues, or suppliers — throughout your lifecycle with them, including finding new customers, winning their business, and providing support and additional services throughout the relationship.

A CRM system gives everyone — from sales, customer service, business development, recruiting, marketing, or any other line of business — a better way to manage the external interactions and relationships that drive success. A CRM tool lets you store customer and prospect contact information, identify sales opportunities, record service issues, and manage marketing campaigns, all in one central location — and make information about every customer interaction available to anyone at your company who might need it.

With visibility and easy access to data, it's easier to collaborate and increase productivity. Everyone in the company can see how customers have been communicated with, what they’ve bought, when they last purchased, what they paid, and so much more. CRM can help companies of all sizes drive business growth, and it can be especially beneficial to a small business, where teams often need to find ways to do more with less.

6.2. Technologies Used

6.2.1. Machine Learning

Machine Learning is a system that can learn from example through self-improvement and without being explicitly coded by programmer. The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results.

Machine learning combines data with statistical tools to predict an output. This output is then used by corporate to makes actionable insights. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input, use an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

Machine learning is also used for a variety of task like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

6.2.2. Machine Learning vs. Traditional Programming

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

How does Machine learning work?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.

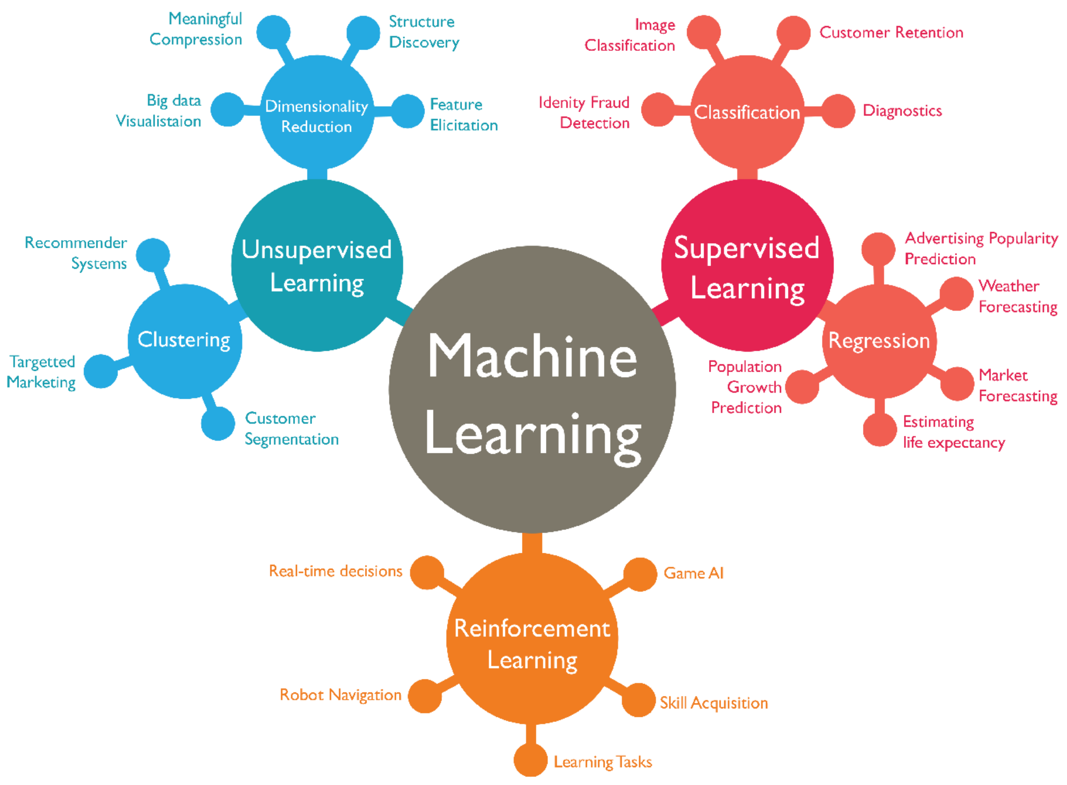


Fig 6.1. Types of Machine Learning

The core objective of machine learning is the learning and inference. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the data. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a feature vector. You can think of a feature vector as a subset of data that is used to tackle a problem.

The machine uses some fancy algorithms to simplify the reality and transform this discovery into a model. Therefore, the learning stage is used to describe the data and summarize it into a model.

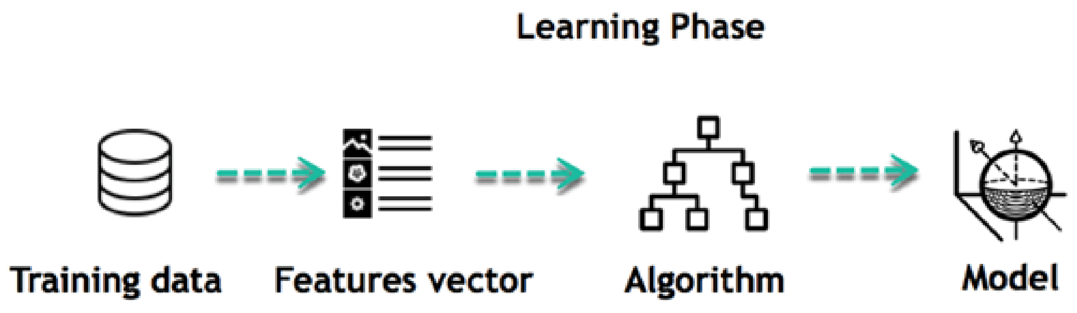


Fig 6.2. Process of Learning

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

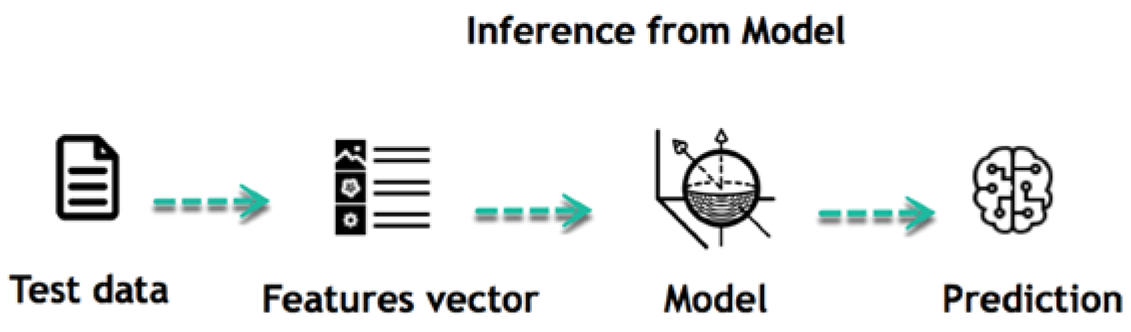


Fig 6.3. Inference Model

The life of Machine Learning programs is straightforward and can be summarized in the following points:

- Define a question
- Collect data
- Visualize data
- Train algorithm
- Test the Algorithm
- Collect feedback
- Refine the algorithm
- Loop 4-7 until the results are satisfying
- Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

6.2.3. Jupyter Notebooks

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating notebook documents.

A Jupyter Notebook document is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media. Underneath the interface, a notebook is a JSON document, following a versioned schema, usually ending with the ".ipynb" extension.

Jupyter notebooks are built upon a number of popular open-source libraries:

- IPython
- ZeroMQ
- Tornado
- jQuery
- Bootstrap (front-end framework)
- MathJax

Jupyter Notebook can connect to many kernels to allow programming in different languages. A Jupyter kernel is a program responsible for handling various types of requests (code execution, code completions, inspection), and providing a reply. Kernels talk to the other components of Jupyter using ZeroMQ, and thus can be on the same or remote machines. Unlike many other Notebook-like interfaces, in Jupyter, kernels are not aware that they are attached to a specific document, and can be connected to many clients at once. Usually kernels allow execution of only a single language, but there are a couple of exceptions By default Jupyter Notebook ships with the IPython kernel. As of the 2.3 release (October 2014), there are 49 Jupyter-compatible kernels for many programming languages, including Python, R, Julia and Haskell.[14]

A Jupyter Notebook can be converted to a number of open standard output formats (HTML, presentation
slides, LaTeX, PDF, ReStructuredText, Markdown, Python) through "Download As" in the web interface, via the nbconvert library or "jupyter nbconvert" command line interface in a shell. To simplify visualisation of Jupyter notebook documents on the web, the nbconvert library is provided as a service through NbViewer which can take a URL to any publicly available notebook document, convert it to HTML on the fly and display it to the user.

The notebook interface was added to IPython in the 0.12 release (December 2011), renamed to Jupyter notebook in 2015 (IPython 4.0 is Jupyter 1.0). Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s.[19] Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

JupyterLab is a newer user interface for Project Jupyter. It offers the building blocks of the classic Jupyter Notebook (notebook, terminal, text editor, file browser, rich outputs, etc.) in a flexible user interface. The first stable release was announced on February 20, 2018.

6.2.4. Python

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

Python Features

Python's features include:

- **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases:** Python provides interfaces to all major commercial databases.
- **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries, and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

- **Scalable:** Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- IT supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

6.2.5. Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, mac OS and Linux.

Why use Navigator?

In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages, and use multiple environments to separate these different versions.

The command line program conda is both a package manager and an environment manager, to help data scientists ensure that each version of each package has all the dependencies it requires and works correctly.

Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages and update them, all inside Navigator.

What applications use anaconda navigator?

The following applications are available by default in Navigator:

- Jupyter Lab
- Jupyter Notebook
- QT Console
- Spyder
- VS Code
- Glue viz
- Orange 3 App
- Rodeo
- RStudio

Advanced conda users can also build your own Navigator applications
How can I run code with Navigator?

The simplest way is with Spyder. From the Navigator Home tab, click Spyder, and write and execute your code.

You can also use Jupyter Notebooks the same way. Jupyter Notebooks are an increasingly popular system that combine your code, descriptive text, output, images and interactive interfaces into a single notebook file that is edited, viewed and used in a web browser.

What’s new in 1.9?

- Add support for Offline Mode for all environment related actions.
- Add support for custom configuration of main windows links.
- Numerous bug fixes and performance enhancements.

6.3. SOURCE CODE

6.3.1.1.Importing Required Packages

```
import os
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib
import sklearn
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedShuffleSplit
from sklearn.preprocessing import OneHotEncoder,
LabelEncoder
from sklearn.naive_bayes import GaussianNB
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier,
ExtraTreesClassifier
from sklearn.metrics import roc_auc_score
from sklearn.metrics import log_loss
from sklearn.model_selection import cross_val_score
     from sklearn.dummy import DummyClassifier
```

6.3.2. Importing Data

```
shopping =
pd.read_csv('C:\\Users\\Krishna\\Desktop\\College\\Mini
Project\\online_shoppers_intention.csv')
shopping.head()
```

6.3.3. Data Preprocessing

```
# Check for null values in data
nullcount = shopping.isnull().sum()
print('Total number of null values in dataset:',
nullcount.sum())
# Checking for number of unique values for each feature
uniques = shopping.nunique(axis=0)
print(uniques)
# Visualize the data
sns.countplot(shopping['Revenue'])
plt.ylim(0,12000)
plt.title('Was the transaction completed?', fontsize= 15)
plt.xlabel('Transaction Completed', fontsize=12)
plt.ylabel('Count (Entries)', fontsize=12)
plt.text(x=-.175, y=11000 ,s='10,422', fontsize=15)
plt.text(x=.875, y=2500, s='1908', fontsize=15)
plt.show()
monthly = shopping['Month'].value_counts()
sns.countplot(shopping['Month'], order=monthly.index)
plt.title('Entries per Month', fontsize=15)
xval = -.42
plt.ylim(0,4000)
for index, value in monthly.items():
    plt.text(x=xval, y=value+50, s=str(value))
    xval += 1.02
shopping['OperatingSystems'] =
shopping['OperatingSystems'].replace([5,6,7,8],5)
os_plot =
shopping['OperatingSystems'].value_counts().plot.pie(figsi
ze=(6,6))
plt.title('Operating Systems', fontsize=15)
plt.show()
sns.countplot(shopping['Browser'])
```

```
plt.title('Browsers', fontsize=15)
plt.show()
# Removing unneccessary columns from data: shopping_clean
shopping_clean =
shopping.drop(['Month','Browser','OperatingSystems','Regio
n','TrafficType','Weekend'], axis=1)
shopping_clean.head()
```

6.3.4. Label and One-Hot Encoding

```
# Encoding Vistor Type
visitor_encoded =
pd.get_dummies(shopping_clean['VisitorType'],
prefix='Visitor_Type', drop_first = True)
shopping_clean_merged = pd.concat([shopping_clean,
visitor_encoded], axis=1).drop(['VisitorType'], axis=1)
visitor_encoded.head()
```

6.3.5. Split Train and Test Data

```
# Split train and test data
X = shopping_clean_merged.drop('Revenue', axis=1)
y = shopping_clean_merged['Revenue']
X_train, X_test, y_train, y_test = train_test_split(X,
y,random_state=2, test_size=.2)
```

6.3.6. Naive Bayes Classifier

```
# Fit Gaussian Naive Bayes Classifier to our training data
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# Make prediction using our test data and model
y_pred = gnb.predict(X_test)

# Comparing our prediction to response values
print("Gaussian Naive Bayes model accuracy(in %):",
round(metrics.accuracy_score(y_test, y_pred)*100,2))
```

6.3.7. Random Forest Classifier

```
# Fit Random Forest Classifier to our Training Data
rfc = RandomForestClassifier(max_depth=5, random_state=2,
n_estimators=750)
rfc.fit(X_train, y_train)

# make prediction using our test data and model
y_pred_rfc = rfc.predict(X_test)
y_prob_rfc = rfc.predict_proba(X_test)[:, 1]

# Comparing our prediciton to response values
print('Random Forest Classifier model accuracy(in %):',
round(metrics.accuracy_score(y_test, y_pred_rfc)*100,2))
```

6.3.8. Fit Extra Trees Classifier to our Training Data

```
# Fit Extra Trees Classifier to our Training Data
etc = ExtraTreesClassifier(random_state=2,
n_estimators=1000)
etc.fit(X_train, y_train)
# make prediction using our test data and model
y_pred_etc = etc.predict(X_test)
y_prob_etc = etc.predict_proba(X_test)[:, 1]

# Comparing our prediciton to response values
print('Extra Trees Classifier model accuracy(in %):',
round(metrics.accuracy_score(y_test, y_pred_etc)*100,2))
```

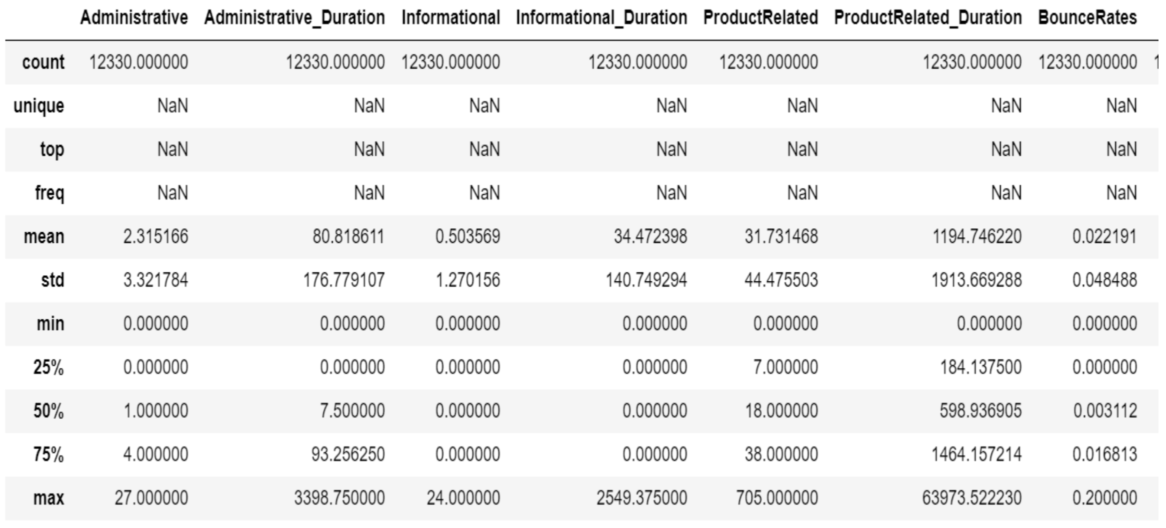
6.3.9. Feature Importance

```
importances = rfc_stratified.feature_importances_
std = np.std([tree.feature_importances_ for tree in
rfc_stratified.estimators_],axis=0)
indices = np.argsort(importances)[::-1]
print("Feature ranking:")
for f in range(X_train_stratified.shape[1]):
     print("%d. Feature %d (%f)" % (f + 1, indices[f],
     importances[indices[f]]))
# Generating a dataframe for our feature importance
col_names = pd.Series([col for col in
X_train_stratified.columns])
```

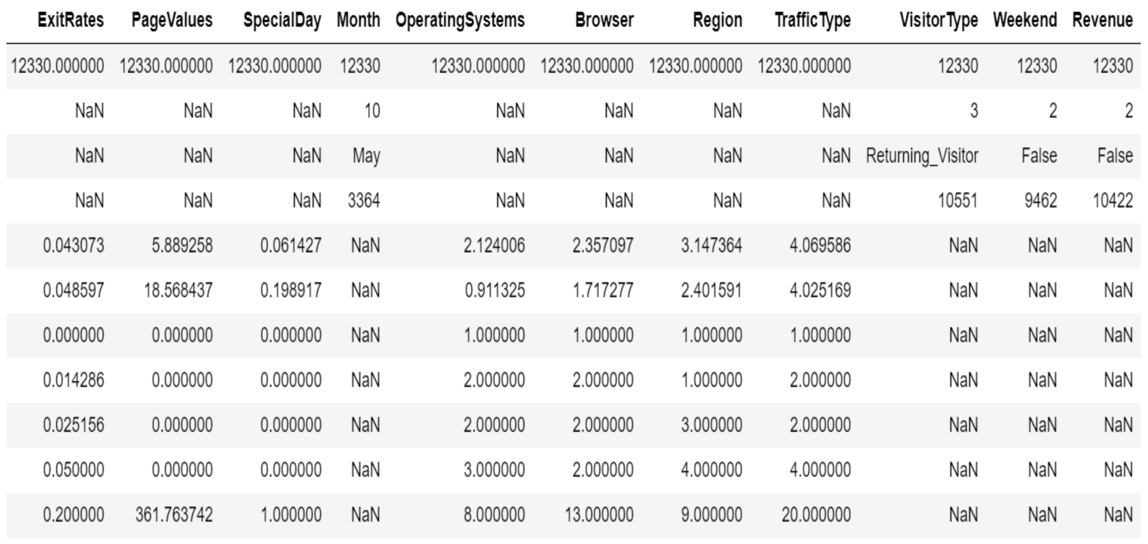
```
importance_df = pd.DataFrame(importances)
importance_df.rename(columns={0:'Importance'},
inplace=True)
importance_df.set_index(col_names,inplace=True)
imp_sorted = importance_df.sort_values(by='Importance',
ascending=False)
plt.title("Feature importances")
plt.bar(range(X_train_stratified.shape[1]),importances[ind
ices])
plt.xticks(range(X_train_stratified.shape[1]),
imp_sorted.index, rotation=90)
plt.xlim([-1, X_train_stratified.shape[1]])
plt.show()
```

6.4. Output Screens

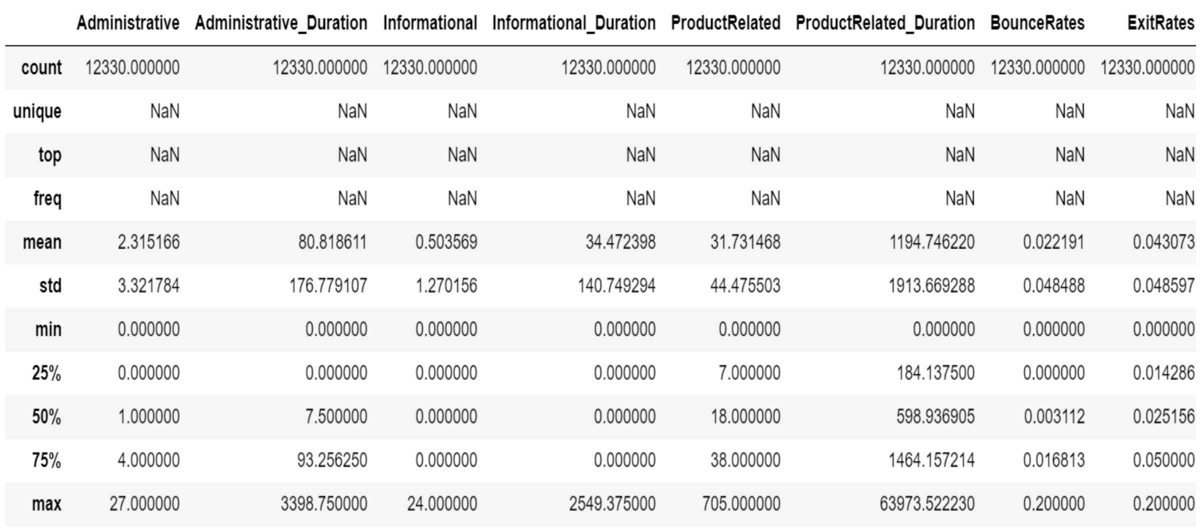
6.4.1. Initial Data Imported



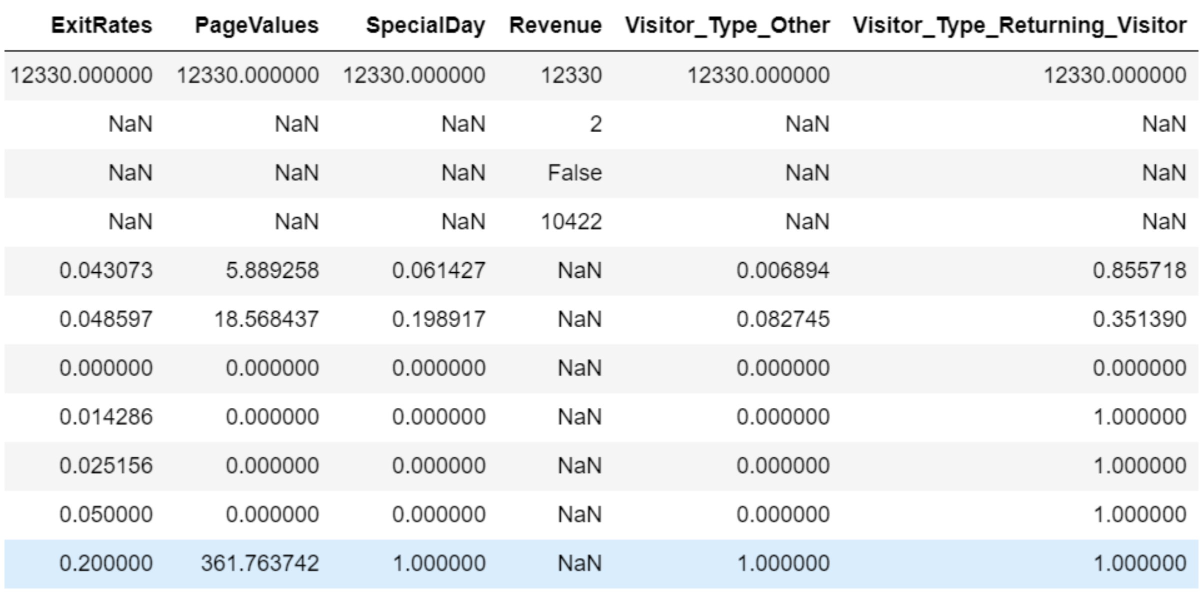
	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates
count	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	2.315166	80.818611	0.503569	34.472398	31.731468	1194.746220	0.022191
std	3.321784	176.779107	1.270156	140.749294	44.475503	1913.669288	0.048488
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	7.000000	184.137500	0.000000
50%	1.000000	7.500000	0.000000	0.000000	18.000000	598.936905	0.003112
75%	4.000000	93.256250	0.000000	0.000000	38.000000	1464.157214	0.016813
max	27.000000	3398.750000	24.000000	2549.375000	705.000000	63973.522230	0.200000



ExitRates	PageValues	SpecialDay	Month	OperatingSystems	Browser	Region	TrafficType	VisitorType	Weekend	Revenue
12330.000000	12330.000000	12330.000000	12330	12330.000000	12330.000000	12330.000000	12330.000000	12330	12330	12330
NaN	NaN	NaN	10	NaN	NaN	NaN	NaN	3	2	2
NaN	NaN	NaN	May	NaN	NaN	NaN	NaN	Returning_Visitor	False	False
NaN	NaN	NaN	3364	NaN	NaN	NaN	NaN	10551	9462	10422
0.043073	5.889258	0.061427	NaN	2.124006	2.357097	3.147364	4.069586	NaN	NaN	NaN
0.048597	18.568437	0.198917	NaN	0.911325	1.717277	2.401591	4.025169	NaN	NaN	NaN
0.000000	0.000000	0.000000	NaN	1.000000	1.000000	1.000000	1.000000	NaN	NaN	NaN
0.014286	0.000000	0.000000	NaN	2.000000	2.000000	1.000000	2.000000	NaN	NaN	NaN
0.025156	0.000000	0.000000	NaN	2.000000	2.000000	3.000000	2.000000	NaN	NaN	NaN
0.050000	0.000000	0.000000	NaN	3.000000	2.000000	4.000000	4.000000	NaN	NaN	NaN
0.200000	361.763742	1.000000	NaN	8.000000	13.000000	9.000000	20.000000	NaN	NaN	NaN



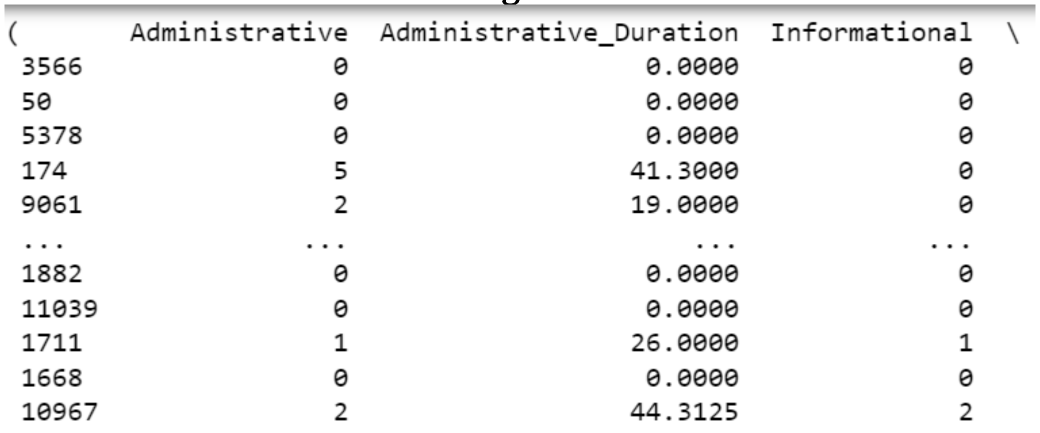
	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates
count	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	2.315166	80.818611	0.503569	34.472398	31.731468	1194.746220	0.022191	0.043073
std	3.321784	176.779107	1.270156	140.749294	44.475503	1913.669288	0.048488	0.048597
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	7.000000	184.137500	0.000000	0.014286
50%	1.000000	7.500000	0.000000	0.000000	18.000000	598.936905	0.003112	0.025156
75%	4.000000	93.256250	0.000000	0.000000	38.000000	1464.157214	0.016813	0.050000
max	27.000000	3398.750000	24.000000	2549.375000	705.000000	63973.522230	0.200000	0.200000



ExitRates	PageValues	SpecialDay	Revenue	Visitor_Type_Other	Visitor_Type_Returning_Visitor
12330.000000	12330.000000	12330.000000	12330	12330.000000	12330.000000
NaN	NaN	NaN	2	NaN	NaN
NaN	NaN	NaN	False	NaN	NaN
NaN	NaN	NaN	10422	NaN	NaN
0.043073	5.889258	0.061427	NaN	0.006894	0.855718
0.048597	18.568437	0.198917	NaN	0.082745	0.351390
0.000000	0.000000	0.000000	NaN	0.000000	0.000000
0.014286	0.000000	0.000000	NaN	0.000000	1.000000
0.025156	0.000000	0.000000	NaN	0.000000	1.000000
0.050000	0.000000	0.000000	NaN	0.000000	1.000000
0.200000	361.763742	1.000000	NaN	1.000000	1.000000

Screen 8.10.1 dataset

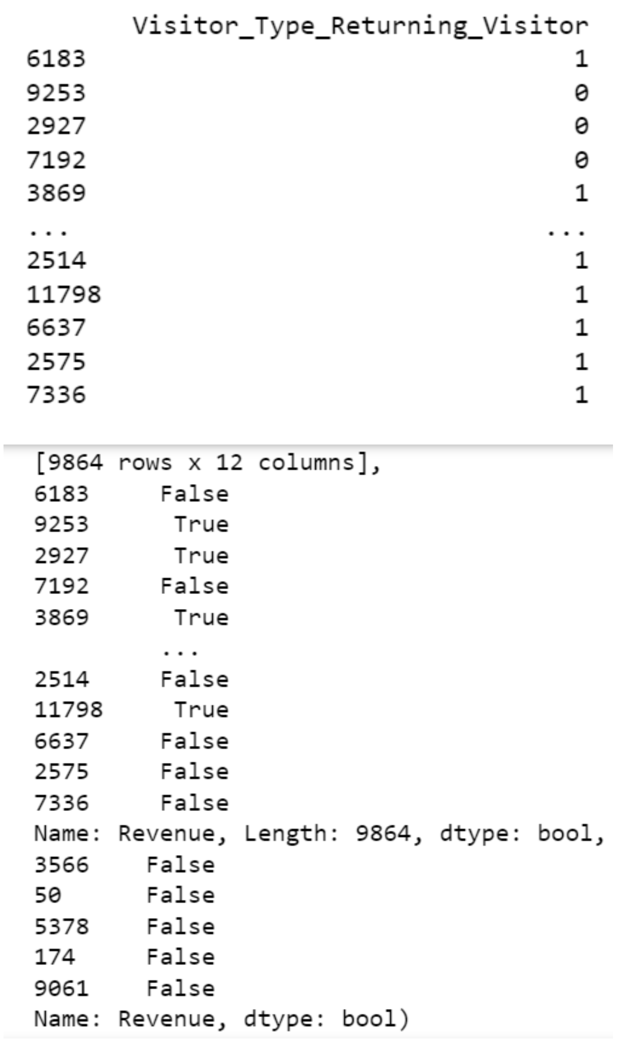
6.4.2. Data After Pre-Processing



```
(       Administrative  Administrative_Duration  Informational  \
 3566                0                   0.0000              0
 50                  0                   0.0000              0
 5378                0                   0.0000              0
 174                 5                  41.3000              0
 9061                2                  19.0000              0
 ...               ...                      ...            ...
 1882                0                   0.0000              0
 11039               0                   0.0000              0
 1711                1                  26.0000              1
 1668                0                   0.0000              0
 10967               2                  44.3125              2
```

```
       Informational_Duration  ProductRelated  ProductRelated_Duration  \
3566                      0.0               1                 0.000000
50                        0.0               1                 0.000000
5378                      0.0               5                 0.000000
174                       0.0              24               446.927778
9061                      0.0              11               658.916667
...                       ...             ...                      ...
1882                      0.0              31              3888.166667
11039                     0.0               1                 0.000000
1711                     55.0              12               457.000000
1668                      0.0               9               355.166667
10967                     7.0              39              2447.548611

       BounceRates  ExitRates  PageValues  SpecialDay  Visitor_Type_Other  \
3566      0.200000   0.200000    0.000000         0.0                   0
50        0.200000   0.200000    0.000000         0.0                   0
5378      0.200000   0.200000    0.000000         0.0                   0
174       0.000000   0.008602    0.000000         1.0                   0
9061      0.000000   0.015385    0.000000         0.0                   0
...            ...        ...         ...         ...                 ...
1882      0.020000   0.031111    0.000000         0.0                   0
11039     0.200000   0.200000    0.000000         0.0                   0
1711      0.014286   0.040476    0.000000         0.0                   0
1668      0.000000   0.004167   18.025330         0.0                   0
10967     0.028355   0.057999    5.291989         0.0                   0

       Visitor_Type_Returning_Visitor
3566                                1
50                                  1
5378                                1
174                                 1
9061                                1
...                               ...
1882                                1
11039                               1
1711                                1
1668                                1
10967                               1
```

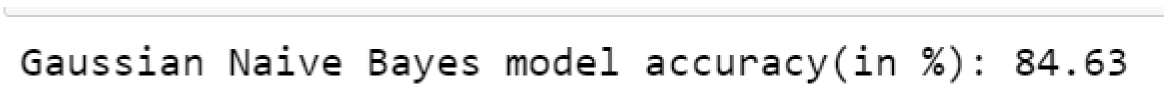
```
[2466 rows x 12 columns],
       Administrative  Administrative_Duration  Informational  \
6183                0                 0.000000              0
9253                9                65.166667              0
2927                0                 0.000000              1
7192                7               204.100000              0
3869                2                25.500000              0
...               ...                      ...            ...
2514                2                40.000000              0
11798               0                 0.000000              0
6637                0                 0.000000              0
2575                1                13.000000              0
7336                1                 6.000000              0

       Informational_Duration  ProductRelated  ProductRelated_Duration  \
6183                      0.0               3                50.400000
9253                      0.0             123              5969.573168
2927                     44.0              14               549.500000
7192                      0.0               5               128.700000
3869                      0.0              13               908.500000
...                       ...             ...                      ...
2514                      0.0              21              1494.500000
11798                     0.0               6                81.750000
6637                      0.0              28               527.833333
2575                      0.0              13               220.333333
7336                      0.0              27              1496.466667

       BounceRates  ExitRates  PageValues  SpecialDay  Visitor_Type_Other  \
6183      0.000000   0.066667    0.000000         0.0                   0
9253      0.001538   0.013923   10.692380         0.0                   0
2927      0.000000   0.013333   43.252000         0.2                   0
7192      0.000000   0.012500    0.000000         0.0                   0
3869      0.000000   0.013333   19.785333         0.0                   0
...            ...        ...         ...         ...                 ...
2514      0.009091   0.004545    0.000000         0.0                   0
11798     0.000000   0.066667    0.000000         0.0                   0
6637      0.029630   0.048148    0.000000         0.0                   0
2575      0.000000   0.002198    0.000000         0.0                   0
7336      0.046914   0.069982    7.468469         0.0                   0
```



```
       Visitor_Type_Returning_Visitor
6183                                1
9253                                0
2927                                0
7192                                0
3869                                1
...                               ...
2514                                1
11798                               1
6637                                1
2575                                1
7336                                1

[9864 rows x 12 columns],
6183     False
9253      True
2927      True
7192     False
3869      True
         ...
2514     False
11798     True
6637     False
2575     False
7336     False
Name: Revenue, Length: 9864, dtype: bool,
3566    False
50      False
5378    False
174     False
9061    False
Name: Revenue, dtype: bool)
```

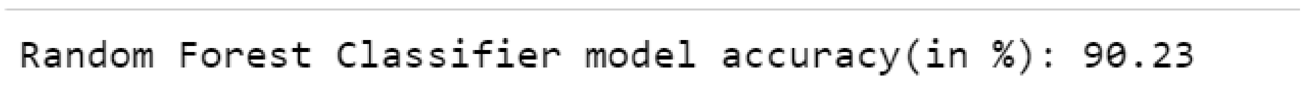
Screen 8.10.2. Data After Pre-Processing

6.4.3. Accuracy of Naïve Bayes Classifier



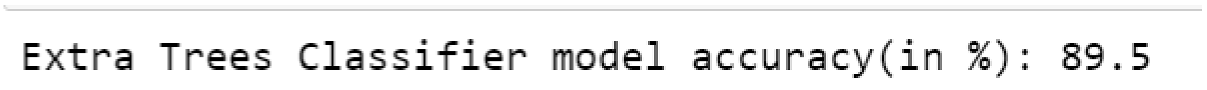
```
Gaussian Naive Bayes model accuracy(in %): 84.63
```

6.4.4. Accuracy of Random Forest Classifier



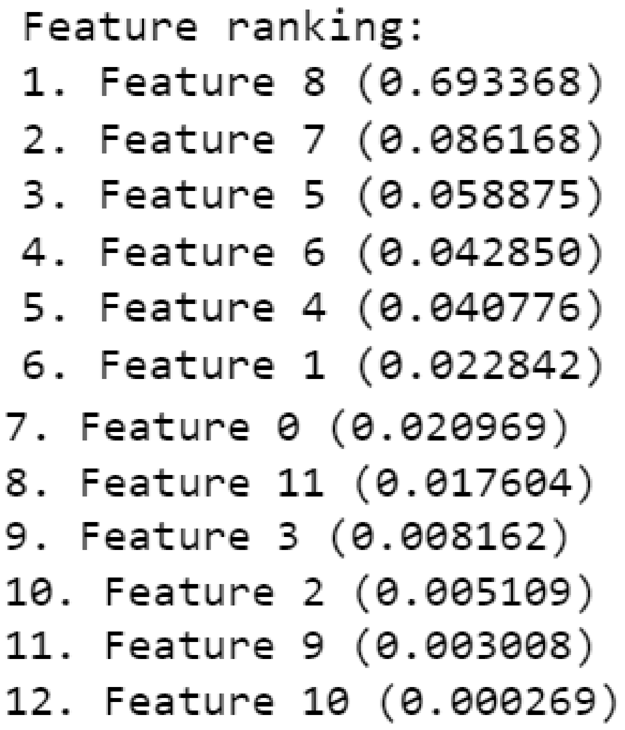
```
Random Forest Classifier model accuracy(in %): 90.23
```

6.4.5. Accuracy of Extra Trees Classifier



```
Extra Trees Classifier model accuracy(in %): 89.5
```

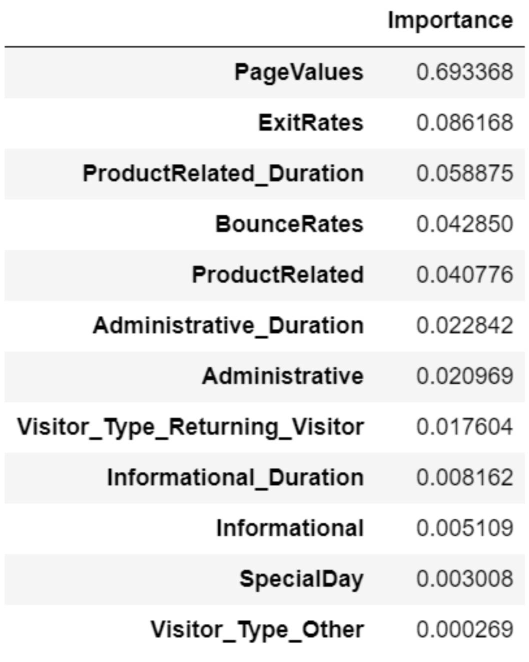
6.4.6. Feature Importance



```
Feature ranking:
 1. Feature 8 (0.693368)
 2. Feature 7 (0.086168)
 3. Feature 5 (0.058875)
 4. Feature 6 (0.042850)
 5. Feature 4 (0.040776)
 6. Feature 1 (0.022842)
7. Feature 0 (0.020969)
8. Feature 11 (0.017604)
9. Feature 3 (0.008162)
10. Feature 2 (0.005109)
11. Feature 9 (0.003008)
12. Feature 10 (0.000269)
```

Screen 8.10.3. Data After Pre-Processing

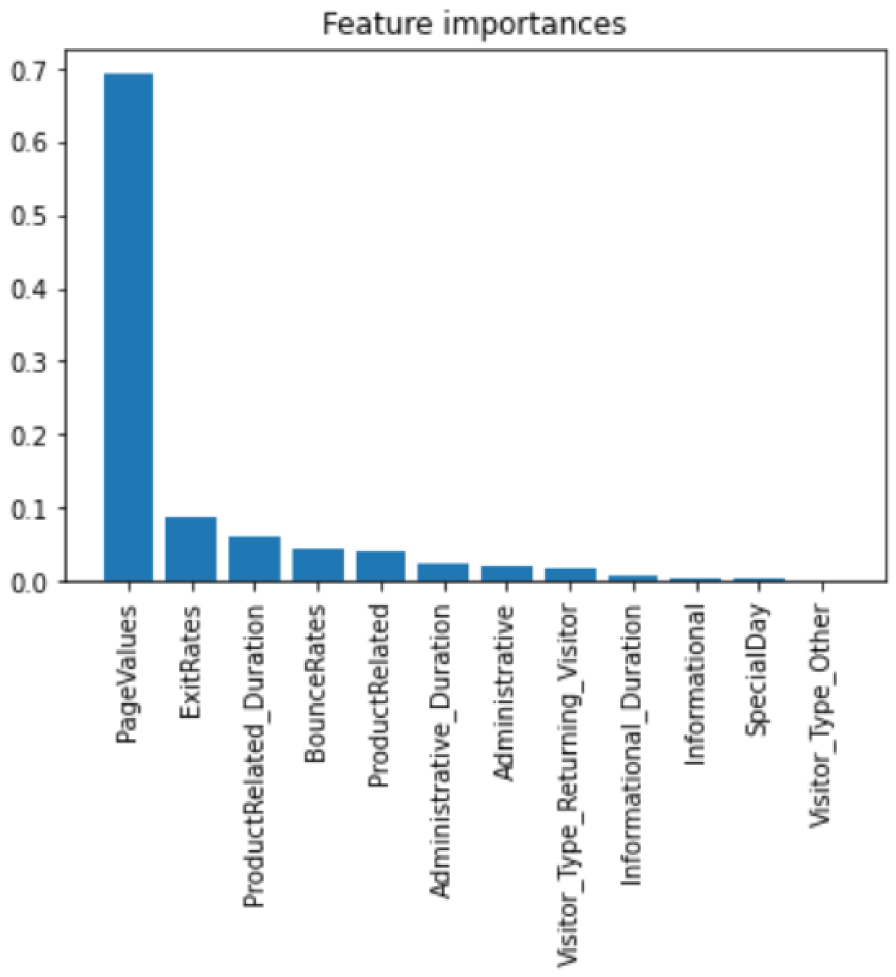
6.4.7. Importance of attributes



	Importance
PageValues	0.693368
ExitRates	0.086168
ProductRelated_Duration	0.058875
BounceRates	0.042850
ProductRelated	0.040776
Administrative_Duration	0.022842
Administrative	0.020969
Visitor_Type_Returning_Visitor	0.017604
Informational_Duration	0.008162
Informational	0.005109
SpecialDay	0.003008
Visitor_Type_Other	0.000269

Screen 8.10.4. Importance of attributes

6.4.8. Final Output



Screen 8.10.5. Final Output

7. CONCLUSION

In this project, the random forest algorithm was applied on an online dataset of shoppers to perform the experiments with much emphasis on the data pre-processing. After analysing the raw dataset and identified biases towards the outcomes of a positive decision, oversampling was used to create an equal distribution of data. The random forest algorithm's parameters were tweaked until a suitable accuracy, precision and recall values were obtained.

The results provide a better prediction as compared to other existing models and classifiers used. With an accuracy of 90% and above for the random forest model, the findings show that data preprocessing and feature selection is of utmost importance in the process of building an ML model. Through this, it was observed that although geographic location and special days have not been in consideration of most researches, it plays a vital role in determining whether or not a purchase decision would be made. Different groups of people react to different interfaces in their own way of which e-commerce is not an exception. The nature of the dataset plays a vital role in determining how the model behaves and hence affecting the decisions of the model.

Due to the imbalance within the revenue decisions, further improvements such as exploring other oversampling or under sampling techniques to the dataset could be made to cater for the imbalances. Further research could also take into consideration the optimization of the models to be able to determine an efficient output in terms of compilation and results.

8. FUTURE WORK

Applications is CRM (customer relationship management)

A customer relationship management (CRM) solution helps you find new customers, win their business, and keep them happy by organising customer and prospect information in a way that helps you build stronger relationships with them and grow your business faster. CRM systems start by collecting a customer's website, email, telephone, social media data, and more, across multiple sources and channels. It may also automatically pull in other information, such as recent news about the company's activity, and it can store personal details, such as a client's personal preferences on communications. The CRM tool organises this information to give you a complete record of individuals and companies overall, so you can better understand your relationship over time.
A CRM platform can also connect to other business apps that help you to develop customer relationships. CRM solutions today are more open and can integrate with your favourite business tools, such as document signing, accounting and billing, and surveys, so that information flows both ways to give you a true 360-degree view of your customer.

By understanding your customers better, cross-selling and upselling opportunities become clear — giving you the chance to win new business from existing customers.

With better visibility, you'll also be able to keep your customers happy with better service. Happy customers are likely to become repeat customers, and repeat customers spend more — up to 33% more according to some studies.

9. REFERENCES

[1] Z. Huang, M. Benyoucef, From e-commerce to social commerce: A close look at design features, Electron. Commer. Res. Appl. 12 (2013) 246–259. https://doi.org/10.1016/j.elerap.2012.12.003.

[2] Y. Christian, Comparison of Machine Learning Algorithms Using WEKA and Sci-Kit Learn in Classifying Online Shopper Intention, J. Informatics Telecommun. Eng. 3 (2019) 58. https://doi.org/10.31289/jite.v3i1.2599.

[3] İ. Topal, Estimation of Online Purchasing Intention Using Decision Tree, J. Manag. Econ. Res. 17 (2019) 269–280. https://doi.org/10.11611/yead.542249.

[4] C.O. Sakar, S.O. Polat, M. Katircioglu, Y. Kastro, Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks, Neural Comput. Appl. 31 (2019) 6893–6908. https://doi.org/10.1007/s00521-018-3523-0.

[5] K. Baati, M. Mohsil, Real-Time Prediction of Online Shoppers' Purchasing Intention Using Random Forest, Artif. Intell. Appl. Innov. AIAI 2020. IFIP Adv. Inf. Commun. Technol. 583 (2020) 43–51. https://doi.org/10.1007/978-3-030-49161-1.

[6] 1-G.H. Haines, J.A. Howard, J.N. Sheth, The Theory of Buyer Behavior., J. Am. Stat. Assoc. 65 (1970) 1406. https://doi.org/10.2307/2284311.

[7] R.C. Marchany, J.G. Tront, E-commerce security issues, in: Proc. 35th Annu. Hawaii Int. Conf. Syst. Sci., IEEE Comput. Soc, 2002: pp. 2500–2508. https://doi.org/10.1109/HICSS.2002.994190.

[8] M.G. Helander, H.M. Khalid, Modeling the customer in electronic commerce, Appl. Ergon. 31 (2000) 609–619. https://doi.org/10.1016/S0003-6870(00)00035-1.

[9] M. Kukar-Kinney, A.G. Close, The determinants of consumers' online shopping cart abandonment, J. Acad. Mark. Sci. 38 (2010) 240–250. https://doi.org/10.1007/s11747-009-0141-5.

[10] J. Cho, Likelihood to abort an online transaction: influences from cognitive evaluations, attitudes, and behavioral variables, Inf. Manag. 41 (2004) 827–838. https://doi.org/10.1016/j.im.2003.08.013.

[11] F. Almeida, J. D. Santos, J. A. Monteiro, E-Commerce Business Models in the Context of Web 3.0 Paradigm, Int. J. Adv. Inf. Technol. 3 (2013) 1–12. https://doi.org/10.5121/ijait.2013.3601.

[12] D.L. Olson, D. Delen, Performance Evaluation for Predictive Modeling, in: Adv. Data Min. Tech., 1st ed., Springer, 2008: pp. 137–147. https://doi.org/10.1007/978-3-540-76917-0.

[13] N. V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic Minority Over-sampling Technique, J. Artif. Intell. Res. 16 (2002) 321–357. https://doi.org/10.1002/eap.2043.

[14] R. Gupta, C. Pathak, A Machine Learning Framework for Predicting Purchase by Online Customers based on Dynamic Pricing, Procedia Computer. Sci. 36 (2014) 599–605. https://doi.org/10.1016/j.procs.2014.09.060.

[15] C.O. Sakar, S.O. Polat, M. Katircioglu, Y. Kastro, Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks, Neural Comput. Appl. 31 (2019) 6893–6908. https://doi.org/10.1007/s00521-018-3523-0.