BIRMINGHAM CITY University

CMP6202
AI and Machine Learning Project
2023–2024

Individual Report

Predicting Shopping Intention Using Random Forest

Kryspin Marcysiak

Table of Contents

1 Report Introduction

This report will be based on identifying and exploring a dataset through visualisation. The dataset will have a value which I will be predicting using a supervised learning method and by developing a model to do so.

1.1 Dataset identification

Upon searching for a dataset through Kaggle, data.gov.uk and UCSB, I eventually sourced a dataset from the UC Irvine Machine Learning Repository. The "Online shoppers purchasing intention" dataset [here] is the dataset chosen for this project. This dataset is integer based and consists of 12,330 sessions; each session represents a unique user. The dataset description states that out of 12,330 sessions, 84.5% returned as negative because a user did not purchase anything. On the other hand, 15.5% of the sessions returned as positive because a user made a purchase. This is very important information as it helps with further stages in the report.

The data is public [here] and has been donated by C.Sakar and Y.Kastro in 2018. They have a collaborative paper analysing this dataset [here]. This data has been collected in a one-year period to avoid specific days, campaigns, and periods such as Valentine's Day. The data for the dataset has been collected using feature vectors which are features based on numerical values used by machine learning models to retrieve data. In this dataset, the data has been extracted from the URL using feature vectors [here]. An example of data acquired are user actions such as "bounce rate", "exit rate" and "page value". These values are measured by Google Analytics and pare provided in the URL of the web page.

The full attribute set is outlined below in Table 1:

Attribute	Description
Administrative	Total number of pages accessed by a visitor about managing their account
Administrative_Duration	Total time (seconds) spent on account management pages
Informational	Number of pages visited by the visitor about Web site, communication, and address information of the shopping site
Informational_Duration	Total time (seconds) spent in informational pages by the visitor
ProductRelated	Number of pages visited by the user on product related pages
ProductRelated_Duration	Total time (seconds) spend by the user on product related pages
BounceRates	Average bounce rate value of pages visited by the user
ExitRates	Average exit rate value of pages by the user
PageValues	Average page value of pages visited by the user
SpecialDay	Closeness of the site visit time by the user to a special day (e.g. Christmas)
OperatingSystems	Operating System of user
Browser	Browser of user
Region	Geographic region of user from session
TrafficType	Traffic type which directed user to web page
VisitorType	Visitor type such as new visitor, returning visitor and other
Weekend	Boolean value indicating if session is on a weekend
Month	Month value of visit
Revenue	Class label indicating whether the visit has concluded in a transaction

Table 1

1.2 Supervised learning task identification

This report aims to predict whether a user will make a purchase during a session by analysing data from the dataset. I will be targeting the revenue variable [see table 1]. This will set the ground truth for this task.

The target variable is the revenue variable as we will be predicting whether a user will make a purchase during their session or not, it'll result in a categorical outcome such as yes or no. This will make this a classification problem rather than a regression problem because classification problems aim to predict weather something will happen or won't happen. On the other hand, regression problems aim to predict a variable outcome, an example of this is a person's vertical jump which would use target attributes such as height, weight, age, and sport activity.

The ground truth variable will be predicted using other variables such as month of session where months like November and April typically drive more traffic as that is where most gifts are sought after. Another variable which may be very useful to predict the revenue variable would he the VisitorType, statistically existing customers have a 60 - 70% probability of finalising a purchase. On the other hand, new users are 5 – 20% likely to finalise a purchase [here].

In conclusion, the classification problem will be affected by each variable in some form of way. In this report, this problem will aim to predict this efficiently.

2 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial step when analysing data. It provides insights to a dataset's foundation and guide a person to make informed decisions when building algorithms and models. The EDA step can also highlight potential issues which may occur during feature selection, model development and pre-processing.

2.1 Question(s) identification

In this stage of the report, I will be creating a table of questions accompanied by a relevance and hypothesis columns. This will create a base purpose for the project and will provide tasks to do. All questions will be answered throughout several stages in this report.

Creating questions is a very useful technique to use when predicting a result using machine learning as it sets out relevant tasks to do when exploring and analysing the data using classification algorithms.

Table 2 below holds all relevant questions:

ID	Question	Relevance	Hypothesis
1	Would the user spending more time on a product page result in a purchase?	Typically, the longer a user spends on a product page, the more they will debate a purchase.	The user would be more likely to purchase a product the more they spend on a page
2	Does a session prior to a "SpecialDay" have a notable increase on the total purchases?	Demand increases typically prior to Christmas and other holidays.	There should be an increase in sales around special days.
3	Does the browser affect the sale outcome?	If a browser is safari or google, it is linked to a smartphone.	Online purchases are a part of normal life and mobile purchases should be more likely to increase revenue.
4	Does the region have a significant impact on the total sales?	Purchases vary to region, regions consist of different cultures, household	There should be a prominent and less prominent regions of both

		income thresholds and more.	users and finalised purchases.
5	What is the most profitable traffic type?	Email marketing is the most effective marketing channel with 56% of traffic being directed through email marketing.	There should be a more prominent marketing channel that drives traffic to sessions. There is likely a marketing channel that leads to purchases too.
6	Is a returning customer more likely to make a purchase than a new customer?	Statistically existing customers have a 60 - 70% probability of finalising a purchase. On the other hand, new users are 5 – 20% likely to finalise a purchase.	I expect there to be a larger number of sales made by existing customers rather than new customers.
7	Are sales more likely to increase throughout the weekend?	The weekends are classes as rest days for most people and typically results to less sales. This might be because on working days people are more likely to hear about a product.	I expect sales to increase on weekdays rather than weekends.

Table 2

2.2 Splitting the dataset

Dataset splitting is a very important step when training a machine learning model and consists of a dataset being split into two subsets: testing and training. The most common ratios that a dataset is split into are 70:30, 80:20, 90:10. This is so that the machine learning model can be trained around the training subset where it will learn patterns, features and more from the data. The testing subset is a separate subset which is not known to the training machine learning model. This is so that the model can be tested efficiently using unique data.

Upon consideration the dataset will be split in an 70/30 split. The 70% will be consumed during the training phase, whilst 30% will be consumed during the testing and validation phase. The training dataset will be used to train the model. On the other hand, the testing dataset will be used to test the model and the validation dataset will be used for further testing after making performance changes to the model.

Since further analysis will be taken of the dataset, the data slitting will occur prior to model development. If data is split now, further visualisations will not present the entire dataset's information and instead will limited data that is presented.

2.3 Exploratory Data Analysis process and results

During this stage, it is important to develop more of an understanding about the dataset. Prior to the EDA, I will present a basic understanding of the dataset. Firstly, I will be mounting a google drive which holds the dataset. For reference the dataset can be found here.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Figure 1

Following this I have assigned the dataset to the variable "ds" as seen in figure 2.

```
ds = pd.read_csv('/content/drive/MyDrive/Finalproject/dataset/online_shoppers_intention.csv')
```

Figure 2

Using the *pandas* library, I checked the total size of the dataset. This is to show that prior information is correct that this dataset has 12,330 rows of data, as seen in figure 3.

```
ds.shape
```

```
(12330, 18)
```

Figure 3

There are a total of 17 columns in this dataset where 15 of the attributes are number based such as integer, Boolean and float. On the other hand, two attributes are objects (strings). This is useful as most attributes are numerical and makes them suitable for statistical view. We retrieved this information from figure 4.

```
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   Administrative           12330 non-null  int64
 1   Administrative_Duration  12330 non-null  float64
 2   Informational            12330 non-null  int64
 3   Informational_Duration   12330 non-null  float64
 4   ProductRelated           12330 non-null  int64
 5   ProductRelated_Duration  12330 non-null  float64
 6   BounceRates              12330 non-null  float64
 7   ExitRates                12330 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(7), int64(7), object(2)
```

Figure 4

Numerical columns consist of a very wide range of values which implies of very diverse user behaviour. A good method to use to gain further understating is using the “describe” feature in the *pandas* library, this will display minimal values, max values, mean values and more. Figure 5 will only share the first three columns, see appendix 1 for the full query.

```
ds.describe()
```

	Administrative	Administrative_Duration	Informational
count	12330.000000	12330.000000	12330.000000
mean	2.315166	80.818611	0.503569
std	3.321784	176.779107	1.270156
min	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000
50%	1.000000	7.500000	0.000000
75%	4.000000	93.256250	0.000000
max	27.000000	3398.750000	24.000000

Figure 5

The query above strictly returned numerical results. The Month and VisitorType are also key to predicting shopping intention therefore we can only query to see object data. As seen in figure 6.

```
ds.describe(include ='object')
```

	Month	VisitorType
count	12330	12330
unique	10	3
top	May	Returning_Visitor
freq	3364	10551

Figure 6

The above figure provides vital data that the most popular visitor is a returning visitor, we now know that more than 10,000 records are of returning visitors. On the other hand, the Month attribute has an only 10 unique values. This clearly implies that not all months have been recorded.

There are also the Boolean attributes seen in figure 7.

```
ds.describe(include = 'bool')
```

	Weekend	Revenue
count	12330	12330
unique	2	2
top	False	False
freq	9462	10422

Figure 7

Like the object attribute, the Boolean attribute post query has presented key information. This is important because the Revenue attribute is primarily false, this means that only 1,908 records consist of a purchase.

The Weekend attribute also is vital as it is now of knowledge that more than 70% of sessions have occurred in the weekday. This partially answers my question from table 1 because we now know that majority traffic is weekday based which may lead to majority sales also being on the weekend.

2.3.1 Univariate

Univariate analysis is an analysis technique commonly used to gain more of an understanding on a dataset by directly analysing individual features. This will use visualisation methods, the *pyplot* library and the *seaborn* library. These libraries will enable the visualisation of data.

2.3.1.1 Month

As stated previously, the month attribute has missing months as it only consists of 10 unique values. It is important that I retrieve what months are in the dataset as it will allow a graph to be in month order and not be mixed, this will result in easier understanding of the data.

```
ds['Month'].unique()
```

```
array(['Feb', 'Mar', 'May', 'Oct', 'June', 'Jul', 'Aug', 'Nov', 'Sep',
       'Dec'], dtype=object)
```

Figure 8

As it is now known what months have been captured in the dataset, it is important that the data is visualised. The most fitting graph type for the month attribute would be a countplot as we need to simply present the total records in each month.

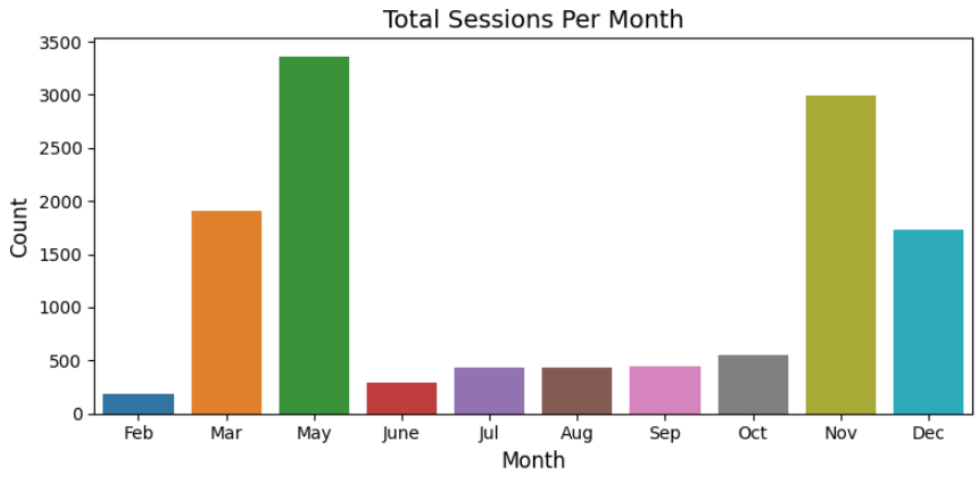

Figure 9

With the unique values being in month order, it is easier to understand what the most common months are and what the least common months are. The reason to why January and April are missing is unknown.

Code in Appendix 8.2.1

2.3.1.2 Revenue

Revenue is the attribute is the ground truth which will be analysed in this section. The revenue is a Boolean variable which states whether a purchase was made in a session or not.

Before I create a lineplot graph I need to group the revenue and month attribute into a table under a new variable named “result”. This was a fix for an issue I was fixing because the “y” parameter was not found. The variable “result” holds a group of attributes and have been put into a subset. As seen in figure 10.

```
result = ds.groupby(['Month','Revenue'])['Revenue'].agg(['count']).reset_index()
result.head()
```

	Month	Revenue	count
0	Aug	False	357
1	Aug	True	76
2	Dec	False	1511
3	Dec	True	216
4	Feb	False	181

Figure 10

A graph which will be most reasonable for this type of data is a lineplot graph. This will allow clear visualisation of revenue trends. But for this graph to make sense I will be including the month attribute. This will result in a visualisation that will represent number of true/ false revenue values in a month. As seen in figure 11.

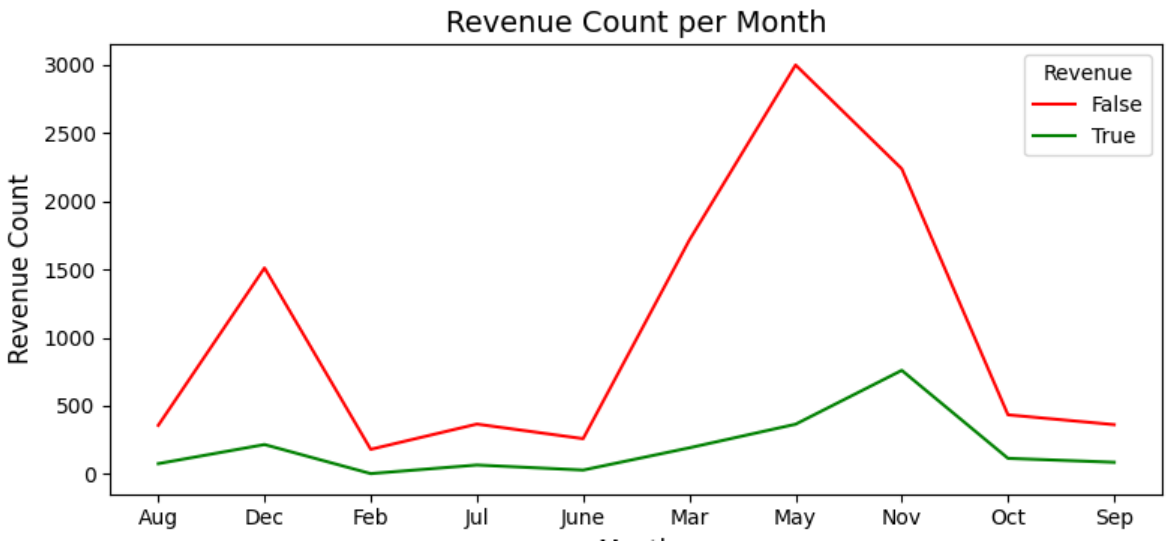

Figure 11

Unlike the visualisation for the month attribute, I was unable to sort the x-axis in month order due to an error which I could not find a fix for. But this is still clearly understandable.

Alike the Month Univariate analysis, we can see the traffic statistic for each month. But we are also able to see the number of sales completed per month. The month of May, maybe the month of most traffic but November is the month of most sales. On the other hand, February is the month with the least sales.

Code in Appendix 8.2.2

2.3.1.3 Weekend

The weekend attribute is a Boolean value whether a session was on a weekend or not. The likely hood of a sale on a weekday is higher due to people hearing through other people about a product and a weekend is classed as a weekend. Alike the lineplot in the Revenue section I had to group attributes once again. See figure 12.

```
weekend = ds.groupby(['Weekend','Revenue'])['Revenue'].agg(['count']).reset_index()
weekend.head()
```

	Weekend	Revenue	count
0	False	False	8053
1	False	True	1409
2	True	False	2369
3	True	True	499

Figure 12

The most applicable graph for this was the barplot to visualise two sets of bars, weekend false and weekend true. Each Boolean has two graph bars reaching a certain revenue count. See figure 13.

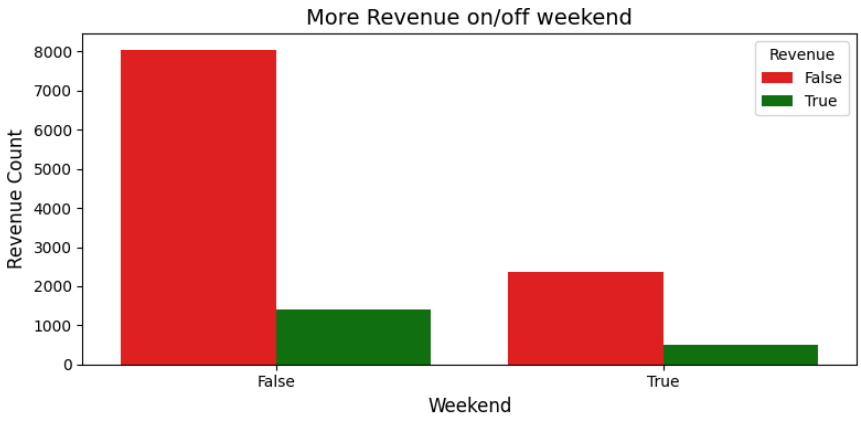

Figure 13

Majority of traffic was not on a weekend and resulted to more sales, unlike on the weekend. The weekday generated more than double sales than the weekend did (see figure 12).

Code in Appendix 8.2.3

2.3.1.4 BounceRate & ExitRate

The bounce rate and exit rate are two very vital statistics due to them being able to be visualised in a graph to show an average of how quickly a user changes tabs or leaves the session. A regplot will be a suitable graph for this as it shows a trend of where the correlation between BounceRates and ExitRates is clearly. As seen in figure 14.

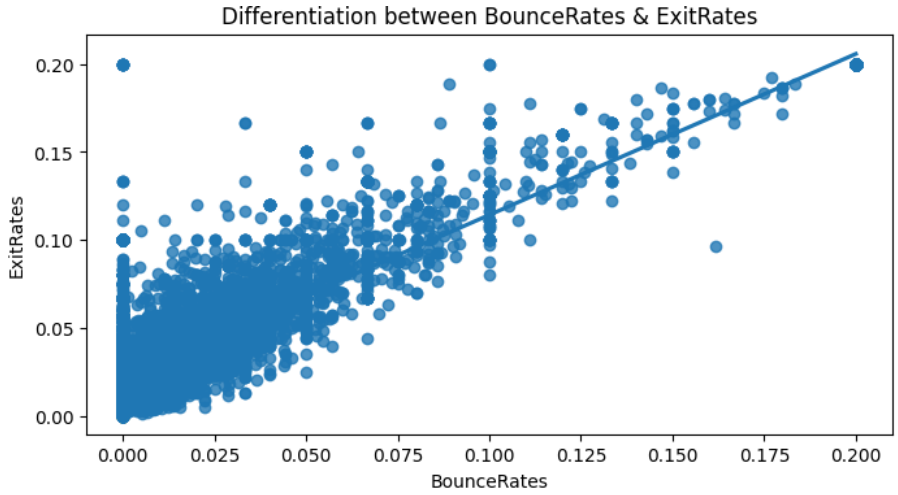

Figure 14

The average of bounce and exit rates by a user during a 10-month course is clear and visible. The average correlation between these two attributes is on the lower end meaning that sessions are not switched as often and are not exited. This is because the bounce rates been below 0.075 and the exit rates being below 0.10.

Code in Appendix 8.2.4

2.3.1.5 Region

The region is an attribute which can influence the revenue. There is not clear information on what the region values represent other than that it is an integer value. Therefore, we can run a query to find that out, as seen in figure 15.

```
ds['Region'].unique()
```

```
array([1, 9, 2, 3, 4, 5, 6, 7, 8])
```

Figure 15

I had to put the region and revenue attributes into a group to provide the lineplot with a subset to handle.

```
regionR = ds.groupby(['Region','Revenue'])['Revenue'].agg(['count']).reset_index()
regionR.head()
```

	Region	Revenue	count
0	1	False	4009
1	1	True	771
2	2	False	948
3	2	True	188
4	3	False	2054

Figure 16

The region ranges from 1-9, this could represent anything from towns, counties to countries. We can find what region does drive the most revenue by using a lineplot. As seen in figure 17.

Figure 17

This has shown that region 1 drives the most traffic but also the most revenue too. On the other hand, region 5 has the least impact on traffic and on generates least revenue.

Code in Appendix 8.2.5

2.3.1.6 Visitor Type

Existing customers are statistically existing customers have a 60 - 70% probability of finalising a purchase. On the other hand, new users are 5 – 20% likely to finalise a purchase. This is due to existing customers already knowing the quality of a product, shipping lengths and more. Firstly, we must group the target attributes into a subset to allow the model to access the data. See figure 18.

```
visitor = ds.groupby(['VisitorType','Revenue'])['Revenue'].agg(['count']).reset_index()
visitor
```

	VisitorType	Revenue	count
0	New_Visitor	False	1272
1	New_Visitor	True	422
2	Other	False	69
3	Other	True	16
4	Returning_Visitor	False	9081
5	Returning_Visitor	True	1470

Figure 18

A barplot is most efficient for this as it can clearly display relevant information. I am unaware of what "other" visitor means; this could mean a user in incognito mode or a user checking out as a guest. But we have a basic understanding of the attribute. See Figure 19 below.

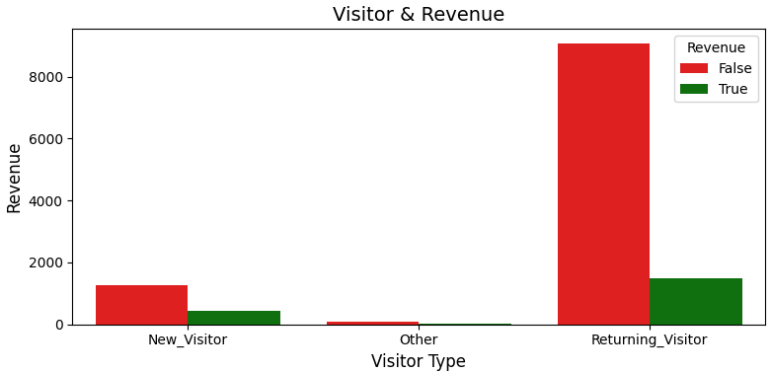

Figure 19

This visualisation clearly shows that most sales are made by returning users, whilst also generating most traffic. On the other hand, new visitors generate one third of revenue compared to its traffic. The other category has a small impact on the revenue and traffic.

Code in Appendix 8.2.6

2.3.1.7 Browser

A browser may be important as to why someone might chose to buy a product of not. The two main browsers over the past decade have been google and safari. It is important to understand the number of unique browsers as seen in figure 20.

```
ds['Browser'].unique()
```

```
array([ 1,  2,  3,  4,  5,  6,  7, 10,  8,  9, 12, 13, 11])
```

Figure 20

As seen, there are a total of 13 unique values (browsers) using this we can now group the revenue and browser attribute into a subset for the model to handle. See figure 21.

```
browserR = ds.groupby(['Browser','Revenue'])['Revenue'].agg(['count']).reset_index()
browserR
```

Figure 21

We are not able to create a lineplot graph for this attribute to visualise the data for further analysis. See figure 22 below.

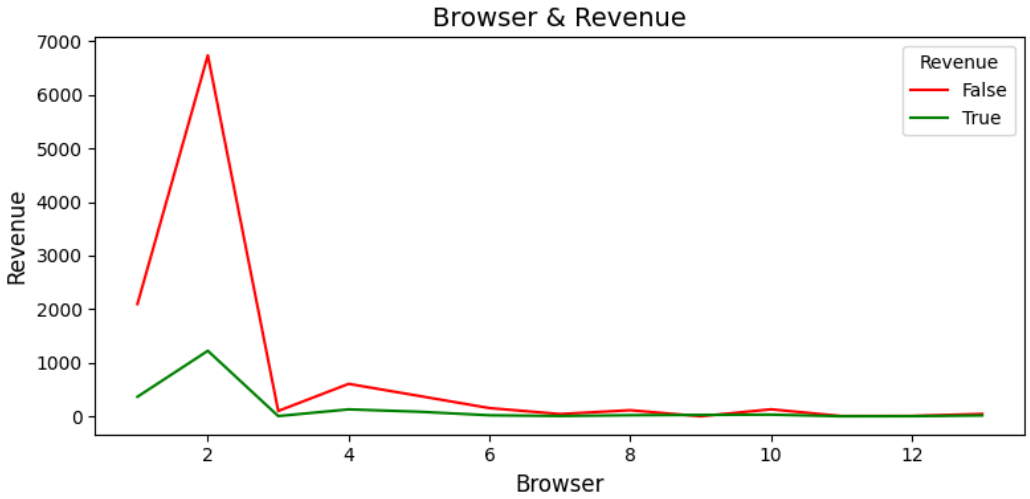

Figure 22

From this visualisation we understand that browser 2 brings most traffic and most revenue.

Code in Appendix 8.2.7

2.3.1.8 Traffic Type

The traffic type is one of the most important statistics to measure as it represents the way traffic is directed to the page, this could be in many forms such as digital marketing, email promotions, references, or social media marketing. But we first need to understand how many unique values there are. See figure 23.

```
ds['TrafficType'].unique()
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 18, 19,
       16, 17, 20])
```

Figure 23

From this we understand there is 20 different traffic channels that direct and attract users. Now by grouping the variables TrafficType and Revenue we can visualise the data into a barplot graph. See figure 24 below.

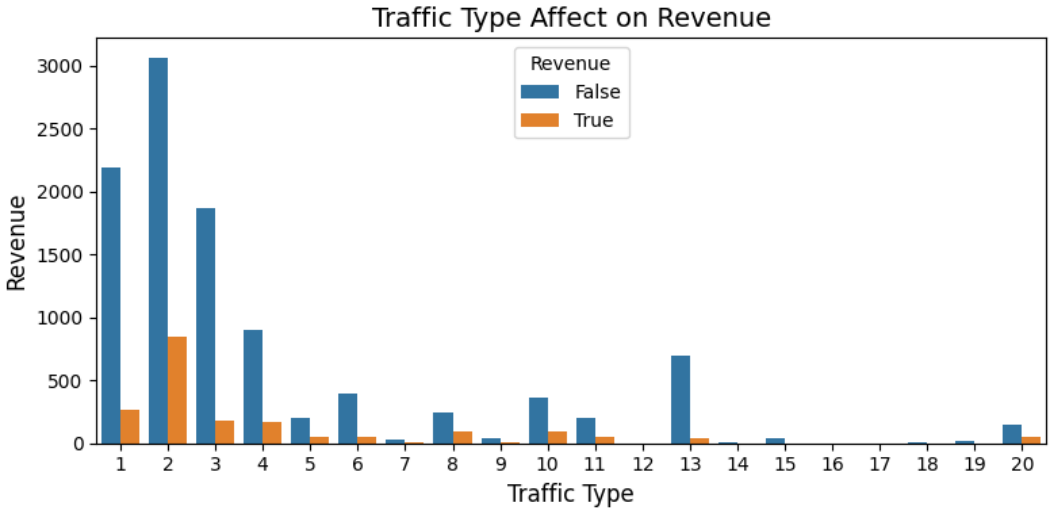

Figure 24

This indicates that majority traffic is directed by type 2, but also is the most successful due to it securing more than 500 sales. On the other hand, it is unknown what the cost for the type is. This is an important point as type 4 may be a fraction of the cost of type 2 and resulted in 20% of the traffic total has had the outcome of a sale.

Code in Appendix 8.2.8

2.3.1.9 Page Values

Page values can be an indication to a higher user session engagement as the customer may be keen after a particular item. This can be represented using a boxplot graph as it can clearly visualise the correlation between the page value and revenue attributes. See figure 25.

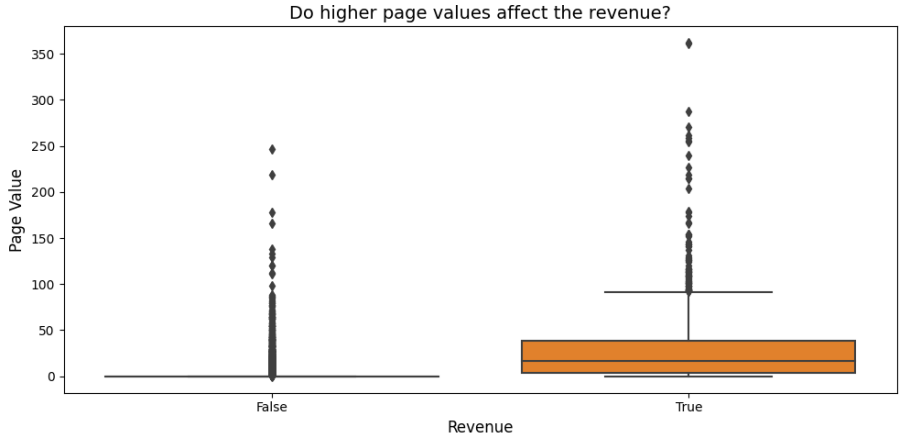

Figure 25

This indicated that the higher the page value is, the more likely a session will result in revenue. This refers to page quality, how well structured a page is and more. This may include modernised webpage design that may influence a user's experience and may affect the revenue outcome. In conclusion, the overall higher page value is, the higher probability of revenue will occur.

Code in Appendix 8.2.9

2.3.1.10 Operating Systems

An operating system is an attribute included in this dataset; it is unclear what operating systems there are. There are a total of 8 unique values, this could represent operating systems such as Windows, Android, iOS and more. See figure 26 for setup procedure.

```
ds['OperatingSystems'].unique()
```

```
array([1, 2, 4, 3, 7, 6, 8, 5])
```

```
system = ds.groupby(['OperatingSystems','Revenue'])['Revenue'].agg(['count']).reset_index()
system
```

Figure 26

An appropriate graph to visualise the Operating System attribute would be a basic barplot graph, it will visualise what browser brings in most traffic and whether it is associated with high values of sales or not. See figure 27.

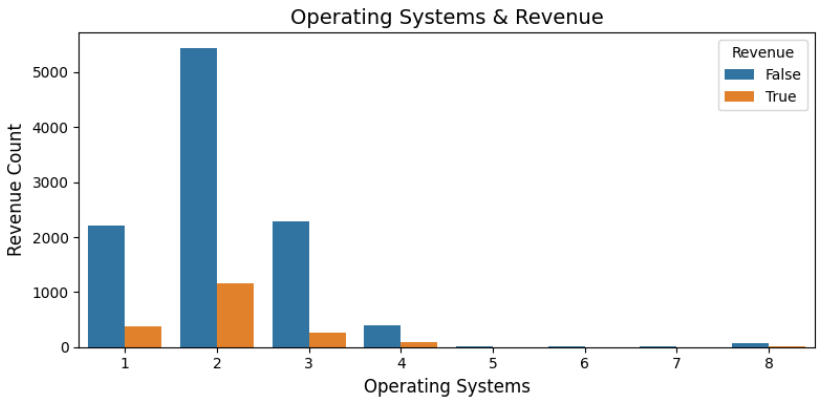

Figure 27

From this visualisation, it is understandable that browser 2 is the main traffic source as well as the browser that results in most sales. There is no specification on what browser 2 is but it is likely android or iOS as purchases are primarily made by people using hand-held devices like phones. This allows users to make purchases on the move.

Code in Appendix 8.2.10

2.3.1.11 Special Day

The special day attribute is an integer-based variable which represents the closeness of session by the user to a special day. The national days which are celebrated worldwide are typically days like, Christmas, Valentines and more. To visualise this data, I will group the SpecialDay and Revenue attributes. See figure 28.

```
ds['SpecialDay'].unique()
```

```
array([0. , 0.4, 0.8, 1. , 0.2, 0.6])
```

```
special = ds.groupby(['SpecialDay','Revenue'])['Revenue'].agg(['count']).reset_index()
special
```

Figure 28

Upon querying unique values in figure 28, the values are from 0 – 1 and have an increment of 0.2. This likely means that 1 is days prior to a special day and lower values represent a further date from a special day like Christmas. A barplot graph is most applicable visualisation method for this data, see figure 29.

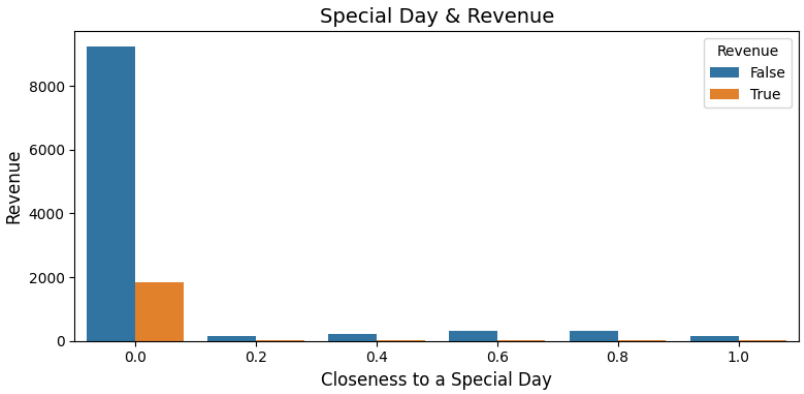

Figure 29

This visualises that more than 90% of traffic and revenue is not close to a special day. This data will not make a significant impact when predicting shopping intention, therefore this attribute is not vital.

Code in Appendix 8.2.11

2.3.1.12 ProductRelated & ProductRelated_Duration

The product related value attribute represents the total of related pages visited by the user. It is accompanied by the duration attribute which can provide support for further analysis. To visualise this data, a group subset must be queried where a “mean” value will have to be included. This is so that we can use an average value to show session trends of each user. See figure 30.

```
ProductRelated_Duration = ds.groupby('ProductRelated')['ProductRelated_Duration'].agg(['count','mean']).reset_index()
ProductRelated_Duration
```

Figure 30

The visualisation method most suitable to present time x value trends will be the lineplot graph. This is because a lineplot graph allows machine learning to visualize two revenue trends according to the Product related value and its duration. See figure 31.

Figure 31

From this visualisation it is clear that the revenue stops at around a 550 average page related value, this could be due to the user not being able to find what they are looking for. There is a peak of the PageRelated_Duration prior to the 500-page related value. But there is a decrease in revenue, this could represent possible page idling by the user hence the high duration value.

Code in Appendix 8.2.12

2.3.1.13 Informational & Informational _Duration

The informational attribute is an average number of pages visited by a user about the web site, communication and address information to gain more of a understanding of the company they will possibly purchase a product from. It is accompanied by the duration attribute which can provide support for further analysis. Alike the [2.3.1.12] section, the attributes will be grouped along a mean variable. See figure 32.

```
Informational_Duration = ds.groupby('Informational')['Informational_Duration'].agg(['count','mean']).reset_index()
Informational_Duration
```

Figure 32

The appropriate visualisation is a pointplot graph as the model will use float data to visualise the trends. This is an appropriate graph as the data has float values that a lineplot is unable to present clearly. See figure 33.

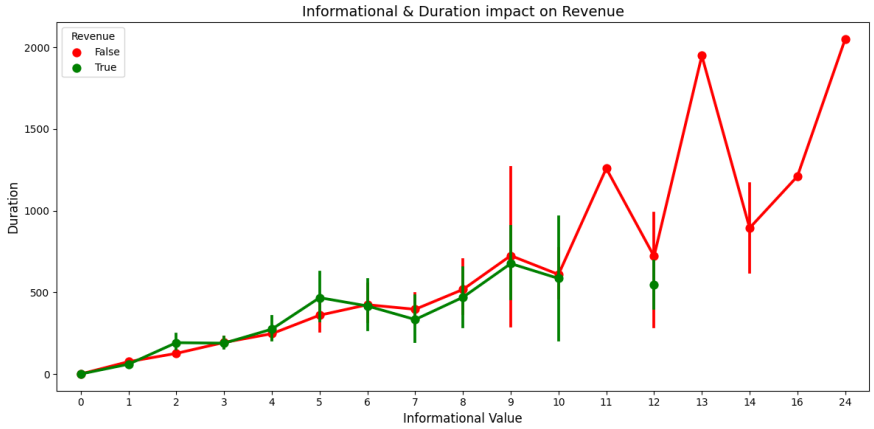

Figure 33

This visualisation clearly presents a revenue trend according to informational value alongside its duration. From this visualisation I can gather that any true revenue is strictly below 1000 duration, this may imply that customers that have done research in regard to the website (for example reviews).

Code in Appendix 8.2.13

2.3.1.14 Administrative & Administrative _Duration

The administrative attribute is a number of pages visited by users about managing their account. It is accompanied by the duration attribute which can provide support for further analysis. Alike the [2.3.1.13] section, the attributes will be grouped along a mean variable. See figure 34.

```
AdministrativeRelated = ds.groupby('Administrative')['Administrative_Duration'].agg(['count','mean']).reset_index(
AdministrativeRelated
```

Figure 34

The appropriate visualisation is a pointplot graph as the model will use float data to visualise the trends. This is an appropriate graph as the data has float values that a lineplot is unable to present clearly. See figure 35.

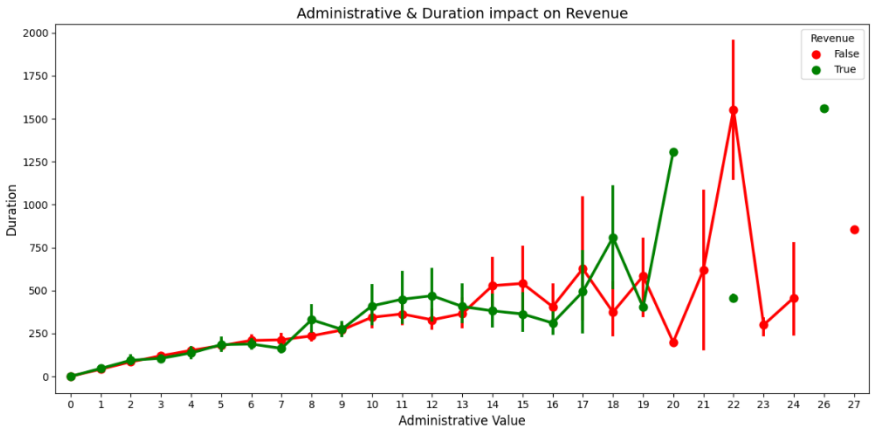

Figure 35

This visualisation presents that users which spend more time managing their account are less likely to make a purchase rather than a user that spends less time on their account. More analysis will need to be done.

Code in Appendix 8.2.14

2.3.2 Multivariate

Multivariate analysis uses two or more attributes from the dataset to gain more of an understanding of the it. This will result in deeper statistical analysis that can help find correlations to prevent using non vital attributes in further stages. There will be multivariate graphs used to .

2.3.2.1 Correlation Analysis Using Heatmap

A heatmap is a multivariate visualisation method used to present correlation between attributes via the use of the x and y axis. The correlation values range between -1 and +1. A correlation of +1 between attributes, means that one attribute increasing will increase the correlating attribute. On the other hand, a -1 correlation between attributes means that one attribute increasing will result in the correlating attribute decreasing. A correlation of 0 is equal to no correlation. Code for this heatmap will be available in Appendix 8.2.15. See figure 36 below for heatmap.

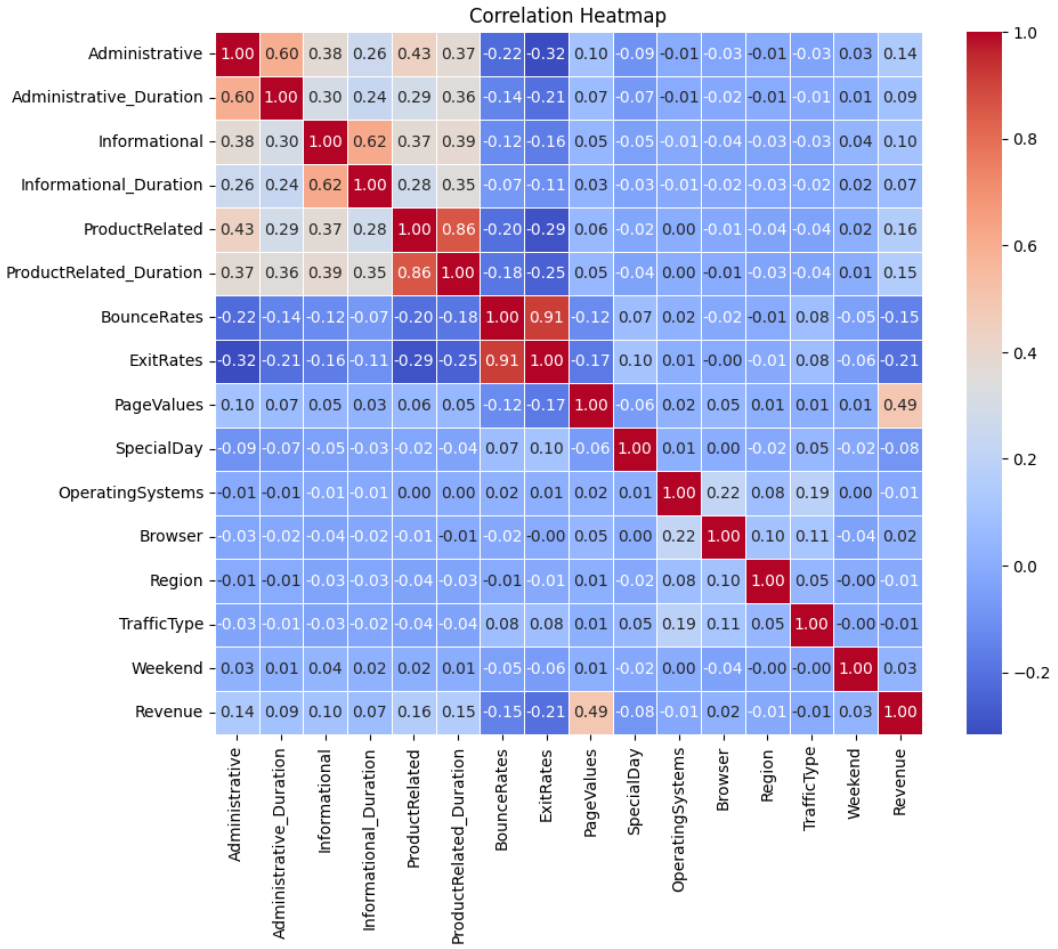

Figure 36

From this visualisation there are multiple attribute which corelate. Revenue is the ground truth variable, and it is clear that that 'PageValues' is has the strongest correlation to Revenue alongside, 'ProductRelated' and 'ProductRelated_Duration'. This implies that higher user retention based of time spend on product related pages and higher page values drastically increase the likelihood of a user making a purchase.

The bounce and exit rates show a strong negative correlation which shows that the higher the exit and bounce rates are, the probability of sales will be reduced. This could be linked to the negative correlation with the visitor type attribute, as explored previously new users have statistically made less purchases than returning customers which may imply that new users have higher likelihood of exiting the page.

A further analysis is required to understand what changes can be made to maximise customer retention and revenue.

2.3.2.2 Visitor Type Correlation with Exit Rates

The heatmap has shown heavy negative correlation between revenue, exit and bounce rates. This could be due to visitor type having an impact on user retention. See figure 37 and 38 below.

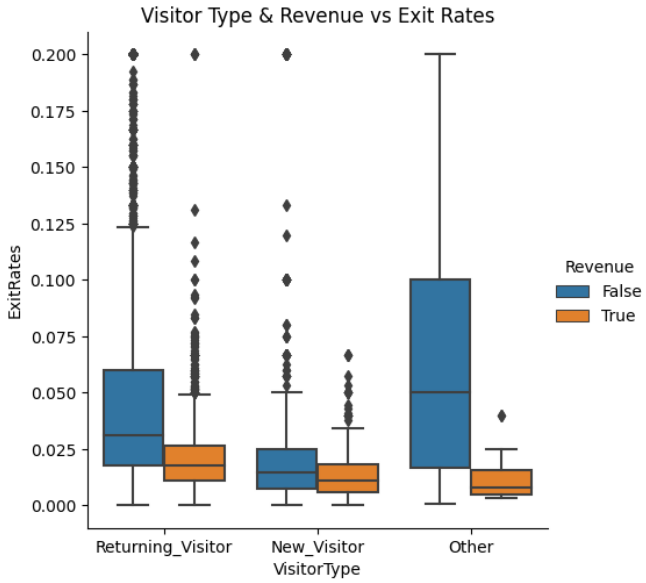

Figure 37

```
ds['VisitorType'].value_counts()
```

```
Returning_Visitor    10551
New_Visitor           1694
Other                   85
```

Figure 38

This box visualisation shows exit rates per type of user accompanied by the revenue attribute. It is clear that there is a higher value of exit rates for returning users. The 'other' value not significant and results in the visualisation showing unclear information. There are more than 10,000 returning visitors meaning that the box graph will be larger due to there being more records. In conclusion, it is clear that returning visitors are on average more likely to exit the page than new visitors. Returning customers are more likely to re-visit the website rather than new customers. This could be a result for that the returning customers may be looking for new products but coming to the fact that there are not any.

Code in Appendix 8.2.16

2.3.2.3 Visitor Type vs Page Values & Product Related Pages

Basing off the correlation visualisation shown earlier it is clear that the 'PageValue' and 'ProductRelated' attributes play a significant role in the Revenue. This is likely due to the user retention increasing if the monetary value of a page is higher. Figure 39 below.

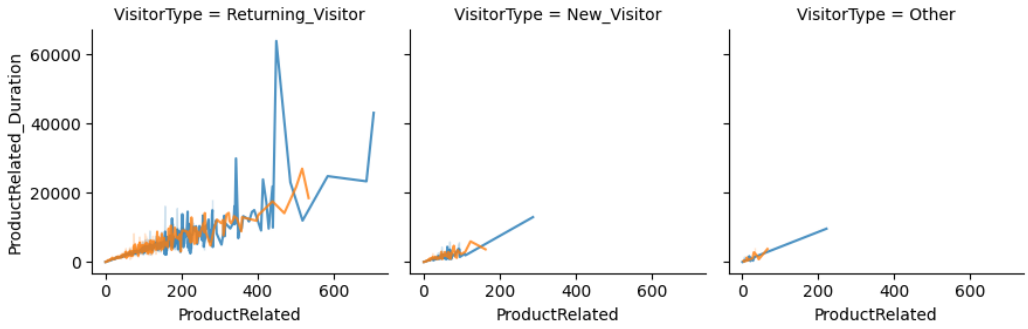

Figure 39

The above visualisation shows the retention on product related pages for the three types of users. This graph shows that the returning visitors alone follow the same trend as in figure 31. This means that new visitors and other visitors do not have an impact on these two values as expected due to the imbalance.

At this stage, it is worth nothing that the 'other' visitor type is not contributing to the dataset. Therefore, it might be worth removing those records to increase model performance.

Code in Appendix 8.2.17

2.4 EDA conclusions

Upon the completion of the EDA stage, further analysis has been taken on the dataset. Table 3 below refers to questions stated in table 2 to summarise the findings from this stage.

ID	Question	Hypothesis	Result
1	Would the user spending more time on a product page result in a purchase?	The user would be more likely to purchase a product the more they spend on a page	Opposed
2	Does a session prior to a "SpecialDay" have a notable increase on the total purchases?	There should be an increase in sales around special days.	Opposed
3	Does the browser affect the sale outcome?	Online purchases are a part of normal life and mobile purchases should be more likely to increase revenue.	Supported
4	Does the region have a significant impact on the total sales?	There should be a prominent and less prominent regions of both users and finalised purchases.	Supported
5	What is the most profitable traffic type?	There should be a more prominent marketing channel that drives traffic to sessions. There is likely a marketing channel that leads to purchases too.	Supported
6	Is a returning customer more likely to make a purchase than a new customer?	I expect there to be a larger number of sales made by existing customers rather than new customers.	Supported
7	Are sales more likely to increase throughout the weekend?	I expect sales to increase on weekdays rather than weekends.	Supported

Table 3

2.4.1 Would the user spending more time on a product page result in a purchase?

The hypothesis is incorrect as the sales decreased with longer product page session times. Instead, a higher value of the page results in more Revenue. This is closely linked to the page value attribute meaning that user retention increases with the value increasing.

2.4.2 Does a session prior to a “SpecialDay” have a notable increase on the total purchases?

The hypothesis was incorrect in this case as the revenue did not increase prior to special days. On the other hand, the revenue was drastically higher far from a special day as seen previously. This is also due to most traffic occurring in the 0.0 value of “closeness to special day”.

2.4.3 Does the browser affect the sale outcome?

The hypothesis is correct as browser 2 brought the most traffic and sales. This is very likely google as google is widely used across both mobile devices and stationary computers.

2.4.4 Does the region have a significant impact on the total sales?

Upon analysing data in regard to this question, it has became clear that region 1 has brought most traffic to the website and most revenue. Region 3 is close by but region 1 has generated most sales.

2.4.5 What is the most profitable traffic type?

The most profitable type is traffic type 2. This could represent promotional emails as that is the most popular way to direct traffic to a website close to product launch events and more. This is supported by previous findings that returning visitors are most frequent to access the website and purchase products/ services.

2.4.6 Is a returning customer more likely to make a purchase than a new customer?

After further analysis it is clear that more than 90% of traffic coming to the website are of returning users. This is supported by Figure 39 where returning users are spending more time on product related pages and generating more revenue.

2.4.7 Are sales more likely to increase throughout the weekend?

Majority traffic appears off weekend as well as more revenue generated on weekdays. Product promotions typically occur electronically but people are more likely to hear about a product/ website throughout the weekday at work, the gym etc. This is a very good promotion for the product as it relies on customer experience. This can be seen through the page value attribute; it has strong correlation with revenue, meaning that customers which are satisfied with their experience are also potentially more likely to share a product to their friends or colleagues.

2.4.8 Conclusion

In conclusion, the EDA process has provided a further understanding of the dataset and its attributes. This process has been very vital as when selecting a supervised learning algorithm, there will be more guidance as to what type of data there is, what the most important attributes are and what attributes that potentially may be dropped from the dataset to improve model performance.

3 Experimental Design

This section of the report focuses on identifying supervised learning algorithms and considering an appropriate one for the machine learning model.

3.1 Identification of your chosen supervised learning algorithm(s)

The ground truth variable which will be predicted is whether revenue will occur or not, therefore this will be a classification problem as stated previously in this report. The appropriate classification supervised learning algorithms which are explored in this report are:

- Logistic Regression: Using calculations to predict a binary outcome. Example being yes or no, pass or fail.
- K-Nearest Neighbours: This algorithm uses a dimensional space to calculate distance between points and predict data.
- Decision Trees: This algorithm will create branches highlighting the most important features and works as an if-else statement.
- Random Forest: This algorithm uses decision trees to make a more accurate prediction.

3.1.1 Logistic Regression

Despite the name of this algorithm, it is an algorithm for a classification problem rather than a regression problem. This learning algorithm is widely used in the data science industry.

It requires the dependant/ ground truth variable to be binary (1 or 0) which we are trying to predict, it will then use mathematical equations to predict an outcome. An example of where logistic regression is used is in the accounting industry where the logistic regression model will predict whether the direct debit usage by a customer is authored or unauthorised.

A benefit of logistic regression is that this is a high speed supervised learning model, this means that this model can process a large dataset at a high speed as they require less capacity from hardware components. Another benefit of using this model is that it is a beginner supervised learning model, making this the most appropriate model for beginners in machine learning.

3.1.2 K-Nearest neighbour

This algorithm is one of the easiest algorithms to implement into machine learning and it can be used for both classification and regression problems. It plots data points in a dimensional space and calculating, distance between them, assigning labels and predicting the ground truth variable.

This supervised learning algorithm is not capable of handling large datasets as it will struggle due to performance issues. On the other hand, this is a great algorithm for smaller datasets typically with less than 5,000 records.

In conclusion, this supervised learning algorithm is not appropriate for this dataset due to its scale. This is because the model will use a lot of computing power and performance may be a concern.

3.1.3 Decision Trees

This is an algorithm which is a hierarchical model that uses decision support to predict potential outcomes, chance events and more. It uses conditional control statements which are very similar to if-then statements. This algorithm can also handle both categorical and numerical attributes which is ideal for this dataset.

This is formed by a root node that has several branches that expand into a tree-like structure. An example of this in regard to the online shopping intention dataset can be; if the page values are high, the user has spend time in a product related page then the user might make a purchase. This algorithm does not require a lot of preparation during the pre-processing stage as it does not need normalisation and scaling. On the other hand, this algorithm is instable meaning if a small change to data is made then a large change in the structure may occur. It also takes longer to train a model using this algorithm.

In conclusion, this model is applicable for the prediction of revenue using machine learning.

3.1.4 Random Forest

Alike the decision tree supervised learning algorithm, random forest uses subset features that consist of decision trees to make more accurate prediction ultimately making a more accurate and robust model.

The random forest algorithm follows a set of instructions. It starts by constructing decision trees which will all generate outputs. Finally, the supervised learning algorithm will use previous output to average for a classification output. This algorithm has the ability to handle both categorical and numerical data therefore data scaling is not necessary. Another benefit of this algorithm is that it removes outlies which is data outside of the average boundary to a certain level.

In conclusion, this algorithm is perfect for this problem as it can provide a balanced high accuracy prediction.

3.1.5 Model Selection

Since the goal is to predict revenue which is a classification problem, there are majority continues values and some categorical values. The most appropriate supervised learning algorithm is the random forest supervised learning algorithm.

This is due to the fact that this algorithm can handle both numerical and categorical data which this dataset holds. There will not be scaling, or transformation of data needed as it works with types of data. The algorithm in general provides a high accuracy, balanced prediction.

This is done by sub-setting data into individual decision trees that will generate an output. The final prediction is calculated by averaging previous outputs. Parallelization does occur which this model which means that the CPU will be used fully to generate random forest for maximum performance. This algorithm does not require to be split as there is always 30% of the data not seen by the decision tree but data was split either way to ensure data is not seen and so the data can be put into a validation split.

Hyperparameters are very important in Random Forests as they can enhance the predictive power of models to increase prediction accuracy and performance. Hyperparameters are to be discussed in further stages. Below will be an image of the structure of the random forest algorithm, see figure 40.

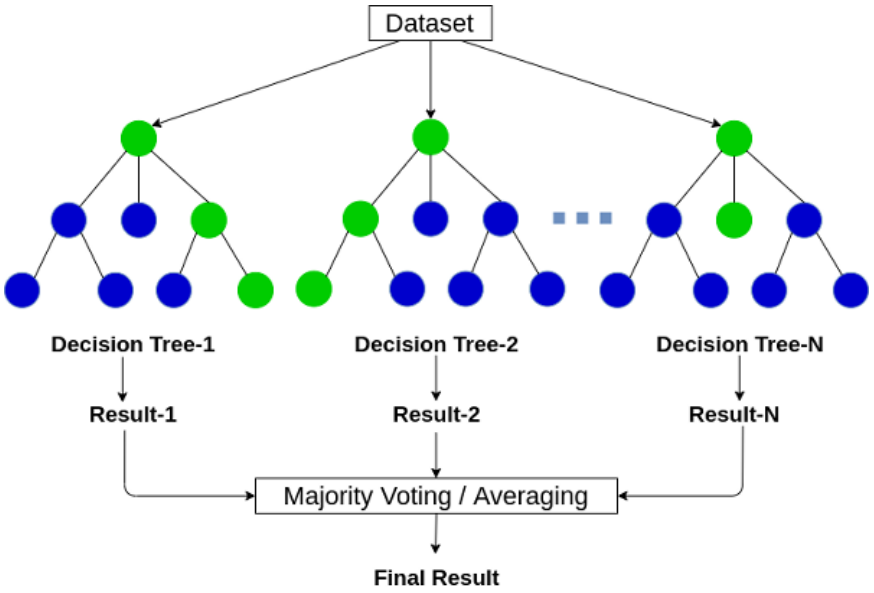

Figure 40

3.2 Identification of appropriate evaluation techniques

There are plenty evaluation techniques that are used during model training and testing to evaluate the model's performance. They are used to monitor the performance so any hyperparameters may be altered as well as, getting accuracy results. Splitting data into train, test and validation subsets is an evaluation technique because we will be maximising performance allowing us to rectify any errors or low accuracy ratings. Splitting the data into three subsets it very important and will allow for maximum model development.

A confusion matrix will be used to achieve a summary of predicted results using known attributes. This will visualize the performance of model using binary classification stating if something is positive or negative. There will be numerous important features measured after data processing using calculations to calculate:

- Accuracy which will calculate the total correct predictions.
- Precision which will help monitor the positive predictions.
- Recall can help reduce negative and false negative predictions.

These evaluation techniques will allow a clearer understanding of the model's performance and efficiency. The higher the precision value will be, the lower the recall value will be.

3.3 Data cleaning and Pre-processing transformations

Data cleaning and preprocessing is a crucial step in model development. If the data is prepared for the model, it will result in more accurate predictions and model reliability. As stated previously, this is a classification problem as we are trying to predict whether Revenue will occur or not. Therefore, there are particular data preparation techniques applicable. Random Forest does not require any data scaling/ transformation as it can handle both categorical and numerical data.

3.3.1 Checking for Missing Data

This dataset should not have any missing values as stated by the author, therefore handling of missing data is not required, see figure 41 below.

There are attributes which are not required in this model and getting rid of them will result in an increase of performance. The weekend attribute and special day attribute are two attributes that are not significant in the dataset and removing them will only benefit the model's performance. See figure 41 below.

```
ds.isna().sum()
```

```
Administrative             0
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
```

Figure 41

After running this query, it is clear that this dataset has zero missing values. This is very helpful as there is no need to manipulate the dataset to restore missing values.

3.3.2 Dropping Unnecessary Features

It is important that the model's performance is taken into consideration. There are variables which are not required when predicting Revenue as they are insignificant. The least vital columns in this dataset are:

- Weekend: Majority of sessions occurred on weekdays and a small portion on a weekend. Additionally, this has had a 0.03 correlation in the heatmap.
- SpecialDay: The majority of the sessions and revenue occurred in the 0.00 value of the value therefore, this is not vital.
- Browser: This is not a required value for prediction due to 0.02 correlation.
- OperatingSystem: This is not a required value for prediction due to -0.01 correlation.
- Region: This is not a required value for prediction due to -0.01 correlation.
- TrafficType: This is not a required value for prediction due to -0.01 correlation.

Therefore, these columns will be dropped from the dataset to maximise accuracy and efficiency. See figure 42 below.

```
ds.drop(columns=['Weekend', 'TrafficType', 'Region', 'Browser', 'OperatingSystems'], inplace=True)
```

Figure 42

Following this I noticed that I did not remove the 'SpecialDay', 'Month' and the 'Browser' column, see figure 43 below.

```
ds.drop(columns=['SpecialDay', 'Month', 'VisitorType'], inplace=True)
```

Figure 43

To verify that six columns have been dropped see figure 44.

```
0   Administrative           12330 non-null  int64
1   Administrative_Duration  12330 non-null  float64
2   Informational            12330 non-null  int64
3   Informational_Duration   12330 non-null  float64
4   ProductRelated           12330 non-null  int64
5   ProductRelated_Duration  12330 non-null  float64
6   BounceRates              12330 non-null  float64
7   ExitRates                12330 non-null  float64
8   PageValues               12330 non-null  float64
9   Revenue                  12330 non-null  bool
```

Figure 44

The above figure shows that the insignificant columns have been successfully dropped. This will improve the performance of the model due to there being less records to learn from.

3.3.3 Splitting the Dataset

This is a process mentioned previously which will limit a model to a dataset and other subsets being used for testing and validation. I will be following a 70-15-15 split where 70% of the dataset will be used to train the model. On the other hand, the 15% will be used for testing purposes and the other 15% for validation purposes which will allow changes to hyperparameters to maximise model performance. There are different splitting methods, but I will be using random dataset splitting. See figure 45 below.

```
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_iris

X=ds
y=ds

X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, train_size = .70)

X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, random_state=42, test_size = .50)
```

Figure 45

The above figure uses libraries that allow the code to split the data accordingly into a train and test data subsets. Following this the test data has been split into two datasets names test and val for validation. This is the most efficient data split for the amount of data the model will be dealing with. To ensure that the data has been split according, we can run a query in figure 46.

```
X_train.shape[0]/X.shape[0]
```

```
0.7
```

Figure 46

This clearly shows that the test dataset is split according. At this point the dataset is ready for the model development process.

3.4 Limitations and Options

I have not completed a lot of pre-processing of the dataset which may cause further issues as no data has been encoded, cleaned deeply and certain techniques have not been applied. This does not necessarily mean that I will face numerous issues but can cause a problem. I am using a large dataset and I have not scaled the dataset down. Some individuals always ensure data is scaled down to less than 3000 records, whether this will case problems or not will be found out later.

4 Predictive Modelling / Model Development

Predictive Modelling Stage can begin as all previous steps have been executed.

4.1 The predictive modelling process

I have created a random forest model supervised learning algorithm that will be trained on the training dataset. As a start I have only used default parameters. See table 4 below.

Parameter	Description	Default
n_estimators:	Increases number of trees which improves model performance but increases load time	0
Max_depth	Increasing the maximum depth of trees	0
Min_samples_split	The minimum of samples required per node	0
bootsrap	Uses bootsrap to build trees	True/False
Random_state	Set random values when building trees for consistent executes	None
verbose	Verbosity of the building process	0

Table 4

Model setup and development can be found in appendix 8.3.

4.2 Evaluation results on “seen” data

After completing the training of the data using the model. The classification report has been formed and shows very interesting information. I used prediction techniques to gather values and their probability can be found in Table 5 below.

Revenue?	Precision	Recall	F1-Score
No	0.92	0.95	0.94
Yes	0.68	0.54	0.89

Table 5 (rounded to 2 d.p)

```
Accuracy: 0.8912925905895078
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.95      0.94      1569
           1       0.68      0.54      0.60       280

    accuracy                           0.89      1849
   macro avg       0.80      0.75      0.77      1849
weighted avg       0.88      0.89      0.89      1849
```

Figure 47

The above figure shows that the current likelihood of revenue is that there will not be any revenue. But taking into consideration that the model is only 0.8912 accurate can imply that the values will change.

5 Evaluation and further modelling improvements

This stage will cover the same model with improvements to the model being made and tested on a validation dataset. In this stage I will be attempting to create the best model most robust model.

5.1 Initial Test

With the model originally being tested on a testing dataset, it will now be tested on a never seen before validation dataset. Unlike in the development stage, the model will be enhanced by the use of hyperparameters which have the ability to increase the accuracy and performance. The initial test will be executed using basic parameters due to the new dataset. Result of 0.8913 accuracy

Test No	Revenue?	Parameters	Precision	Recall	F1-Score
0	No	Default	0.91	0.96	0.94
0	Yes	Default	0.72	0.53	0.61

Table 6

This is very close to the training results from the previous stage where there is only a -0.01 or +0.01 difference in evaluation techniques.

5.2 Further modelling improvements and hyperparameter tweaks

This is a stage which will use hyperparameters to tweak the model allowing for improved performance.

Test No	Revenue?	Parameters	Precision	Recall	F1-Score
0	No	Default	0.91	0.96	0.94
0	Yes	Default	0.72	0.53	0.61
1	No	Man_estimators=200	0.91	0.96	0.94
1	Yes	Man_estimators=200	0.70	0.53	0.60
2	No	max_depth =20	0.92	0.96	0.94
2	Yes	max_depth =20	0.73	0.55	0.62

3	No	max_depth =20, min_samples_split=5	0.92	0.96	0.94
3	Yes	max_depth =20, min_samples_split=5	0.71	0.53	0.61

Table 7

5.2.1 Test 1

For this being the first tune having the *max_estimators* hyperparameter and it decreased the performance of the model by one point negatively. This was very disappointing as I expected an increase in performance.

5.2.2 Test 2

This test has raised the values back up from test one, reaching a accuracy of 0.8945 which is very close to a 90% accuracy rating.

5.2.3 Test 3

The last test decreased the performance from Test 2 resulting in nearly a 89% accuracy.

5.2.4 Final Evaluation Results

Due to me facing a lot of issues with m laptop at this stage I was unfortunately unable to accomplish what I wanted to achieve. In conclusion, there is a higher chance that revenue is not generated due to numerous values especially page values.

6 Conclusion

6.1 Summary of results

In conclusion, this report has covered the topic of data science to a higher level and deep data exploration and analysis has been conducted. The main values that affect the probability of revenue generation are page value and product related pages. This is likely due to the page quality being able to keep user retention. The overall revenue likelihood being in the 71% is still a high revenue margin and is dependant on a lot of variables and instances.

6.2 Reflection on Individual Learning

Everything in this module has been new to me, therefore I learn a lot. From the basics to the more advanced information covered in this report. I really enjoyed developing my knowledge of this module, from creating visualisations to using a supervised learning algorithm to enhance my knowledge and predict a value. I am looking forward to developing my skill further in Artificial Intelligence through more practical work and researching about the topic.

I have came across a lot of issues with the model development due to my laptop failing numerous times and me losing report progress, but I have tried my best given the circumstances. I understand a late penalty taking 10% off this report due to late submission but I unfortunately had no choice but to.

7 References

C.Sakar & Y.Kastro (2018) *Online Shoppers Purchasing Intention Dataset.* Available at: https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset

Iguazio (unknown) *What is a feature Vector?* Available at: https://www.iguazio.com/glossary/feature-vector/

G.Roots (2021) *Extracting Feature Vectors From URL Strings For Malicious URL Detection.* Available at: https://towardsdatascience.com/extracting-feature-vectors-from-url-strings-for-malicious-url-detection-cbafc24737a

K.Saleh (unknown) *Customer Acquisition Vs.Retention Costs – Statistics And Trends.* Available at: https://www.invespcro.com/blog/customer-acquisition-retention/#:~:text=The%20probability%20of%20selling%20to,when%20compared%20to%20new%20customers

B.Ward (2023) *When Are People Most Likely to Buy Online?.* Available at: https://www.salecycle.com/blog/stats/when-are-people-most-likely-to-buy-online/

Zach (2022) *How to Change the Colors in a Seaborn Lineplot.* Available at: https://www.statology.org/seaborn-lineplot-color/

A.Subasi(2020) *Logistic Regression.* Available at: https://www.sciencedirect.com/topics/computer-science/logistic-regression#:~:text=Logistic%20regression%20is%20a%20simple,algorithm%20for%20classification%20in%20industry.

J.Patel (2023) *Top 6 Machine Learning Algorithms for Classification.* Available at: https://www.linkedin.com/pulse/top-6-machine-learning-algorithms-classification-jagrat-patel/?trk=article-ssr-frontend-pulse_more-articles_related-content-card

A.Jain (2020) *Advantages and Disadvantages of Logistic Regression in Machine Learning.* Available at: https://medium.com/@akshayjain_757396/advantages-and-disadvantages-of-logistic-regression-in-machine-learning-a6a247e42b20

Unknown (unknown) *Delete a column from a Pandas DataFrame.* Available at: https://stackoverflow.com/questions/13411544/delete-a-column-from-a-pandas-dataframe

C.Sakar, S.Polat, M.Katircioglu & Y.Kastro (2018) *Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks*. Available at: https://link.springer.com/article/10.1007/s00521-018-3523-0

8 Appendix

8.1 APPENDIX 1

```
ds.describe()
```

	Administrative	Administrative_Duration	Informational	Informational_Duration	ProductRelated	ProductRelated_Duration	BounceRates	ExitRates	PageValues	SpecialDay	OperatingSystems	Browser	Region	TrafficType
count	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000	12330.000000
mean	2.315166	80.818611	0.503569	34.472398	31.731468	1194.746220	0.022191	0.043073	5.889258	0.061427	2.124006	2.357097	3.147364	4.069586
std	3.321784	176.779107	1.270156	140.749294	44.475503	1913.669288	0.048488	0.048597	18.568437	0.198917	0.911325	1.717277	2.401591	4.025169
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	1.000000	1.000000	1.000000
25%	0.000000	0.000000	0.000000	0.000000	7.000000	184.137500	0.000000	0.014286	0.000000	0.000000	2.000000	2.000000	1.000000	2.000000
50%	1.000000	7.500000	0.000000	0.000000	18.000000	598.936905	0.003112	0.025156	0.000000	0.000000	2.000000	2.000000	3.000000	2.000000
75%	4.000000	93.256250	0.000000	0.000000	38.000000	1464.157214	0.016813	0.050000	0.000000	0.000000	3.000000	2.000000	4.000000	4.000000
max	27.000000	3398.750000	24.000000	2549.375000	705.000000	63973.522230	0.200000	0.200000	361.763742	1.000000	8.000000	13.000000	9.000000	20.000000

8.2 Visualisations

8.2.1 Appendix

Months have been put in order using the order query and countplot has been created.

```
ds['Month'].unique()
```

```
array(['Feb', 'Mar', 'May', 'Oct', 'June', 'Jul', 'Aug', 'Nov', 'Sep',
       'Dec'], dtype=object)
```

```
plt.figure(figsize=(8,4))
sns.countplot(x='Month', data=ds, order=['Feb', 'Mar', 'May', 'June', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'])

plt.title('Total Sessions Per Month', fontsize=14)
plt.xlabel('Month', fontsize=12)
plt.ylabel('Count', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.2 Appendix

Lineplot has been created to represent a trend.

```
plt.figure(figsize=(8, 4))
sns.lineplot(data=result, x='Month', y='count', hue='Revenue', palette=['red', 'green'])

plt.title('Revenue Count per Month', fontsize=14)
plt.xlabel('Month', fontsize=12)
plt.ylabel('Revenue Count', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.3 Appendix

Barplot has been created to represent difference between weekend and normal days.

```
plt.figure(figsize=(8,4))
sns.barplot(data=weekend, x='Weekend', y='count', hue='Revenue', palette=['red', 'green'])

plt.title('More Revenue on/off weekend', fontsize=14)
plt.xlabel('Weekend', fontsize=12)
plt.ylabel('Revenue Count', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.4 Appendix

Regplot as been created to show correlation between exit and bounce rates.

```
plt.figure(figsize=(8, 4))

sns.regplot(data=ds, x='BounceRates', y='ExitRates')
plt.title("Differentiation between BounceRates & ExitRates", fontsize=14)

plt.xlabel("BounceRates", fontsize=12)
plt.ylabel("ExitRates", fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.5 Appendix

Lineplot created to show trend of regions.

```
plt.figure(figsize=(8, 4))
sns.lineplot(data=regionR, x='Region', y='count', hue='Revenue', palette=['red', 'green'])

plt.title('Region & Revenue', fontsize=14)
plt.xlabel('Region', fontsize=12)
plt.ylabel('Revenue by Region', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.6 Appendix

barplot created to show difference in visitor types.

```
plt.figure(figsize=(8, 4))
sns.barplot(data=visitor, x='VisitorType', y='count',hue='Revenue', palette=['red', 'green'])

plt.title("Visitor & Revenue", fontsize=14)
plt.xlabel("Visitor Type", fontsize=12)
plt.ylabel("Revenue", fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.7 Appendix

Lineplot created to show trend of browsers.

```
plt.figure(figsize=(8, 4))
sns.lineplot(data=browserR, x='Browser', y='count', hue='Revenue', palette=['red', 'green'])

plt.title('Browser & Revenue', fontsize=14)
plt.xlabel('Browser', fontsize=12)
plt.ylabel('Revenue', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.8 Appendix

barplot created to show the different traffic types.

```
plt.figure(figsize=(8, 4))
sns.barplot(data=traffic, x='TrafficType', y='count', hue='Revenue'

plt.title("Traffic Type Affect on Revenue", fontsize=14)
plt.xlabel("Traffic Type", fontsize=12)
plt.ylabel("Revenue", fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.9 Appendix

Boxplot created to show importance of page values.

```
plt.figure(figsize=(8, 4))
sns.boxplot(data=ds, x='Revenue', y='PageValues')

plt.title("Do higher page values affect the revenue?", fontsize=14
plt.xlabel("Revenue", fontsize=12)
plt.ylabel("Page Value", fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.10 Appendix

Barplot created to show number of operating systems.

```
plt.figure(figsize=(8, 4))
sns.barplot(data=system, x='OperatingSystems', y='count', hue='Revenue')

plt.title("Operating Systems & Revenue", fontsize=14)
plt.xlabel("Operating Systems", fontsize=12)
plt.ylabel("Revenue Count", fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.11 Appendix

Barplot created to analyse special days.

```
plt.figure(figsize=(8, 4))
sns.barplot(data=special, x='SpecialDay', y='count', hue='Revenue')

plt.title('Special Day & Revenue', fontsize=14)
plt.xlabel('Closeness to a Special Day', fontsize=12)
plt.ylabel('Revenue', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.12 Appendix

Lineplot created to show trend of product pages and its importance.

```
plt.figure(figsize=(8, 4))
sns.lineplot(data=ds, x='ProductRelated', y='ProductRelated_Duration', hue='Revenue', palette=['red', 'green'])

plt.title('Product Value & Duration impact on Revenue', fontsize=14)
plt.xlabel('Page Related Value', fontsize=12)
plt.ylabel('Duration', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.13 Appendix

Pointplot created to show trend of information pages.

```
plt.figure(figsize=(12, 6))
sns.pointplot(data=ds, x='Informational', y='Informational_Duration', hue='Revenue', palette=['red', 'green'])

plt.title('Informational & Duration impact on Revenue', fontsize=14)
plt.xlabel('Informational Value', fontsize=12)
plt.ylabel('Duration', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.14 Appendix

Pointplot created to show trend of administration pages.

```
plt.figure(figsize=(12, 6))
sns.pointplot(data=ds, x='Informational', y='Informational_Duration', hue='Revenue', palette=['red', 'green'])

plt.title('Informational & Duration impact on Revenue', fontsize=14)
plt.xlabel('Informational Value', fontsize=12)
plt.ylabel('Duration', fontsize=12)

plt.tight_layout()
plt.show()
```

8.2.15 Appendix

Heatmap created to show correlational values between Attributes.

```
correlation_heatmap = ds.corr()

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_heatmap, annot=True, cmap='coolwarm', fmt='.2f', linewidths=.5)
plt.title('Correlation Heatmap')

plt.show()
```

8.2.16 Appendix

Cat plot created to show importance of visitor types and exit rates.

```
plt.figure(figsize=(10, 8))
sns.catplot(data=ds, x='VisitorType', y='ExitRates', hue='Revenue', kind='box')
plt.title('Visitor Type & Revenue vs Exit Rates')
```

8.2.17 Appendix

facetgrid created to show trends of visitor types.

```
plt.figure(figsize=(10, 10))

f = sns.FacetGrid(data=ds, col='VisitorType', hue='Revenue')
f.map(sns.lineplot, "ProductRelated", "ProductRelated_Duration", alpha = .8)
```

8.3 Model Development

The model chosen is the random forest classifier. This is a model discussed previously and below will be a step-to-step guide on how I did it.

```
from sklearn.ensemble import RandomForestClassifier
classifier_rf = RandomForestClassifier(random_state=42)
```

The RandomForestClassifier library had to be imported as that is the method that will be used to predict Revenue value. Following that I set a variable as the classifier for later purpose and the random state is a default value widely used by data scientists.

```
classifier_rf.fit(X_train, y_train)
```

```
RandomForestClassifier(random_state=42)
```

```
y_pred = classifier_rf.predict(X_test)
y_pred
```

The variable model was fitted into the training dataset and then a predict variable was run which is a feature of our model that can use average values to find a mean.

```
accuracy = accuracy_score(y_test, y_pred)
classification_rep = classification_report(y_test, y_pred)

print(f'Accuracy: {accuracy}')
print(f'Classification Report:\n{classification_rep}')
```

```
Accuracy: 0.8912925905895078
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.95      0.94      1569
           1       0.68      0.54      0.60       280

    accuracy                           0.89      1849
   macro avg       0.80      0.75      0.77      1849
weighted avg       0.88      0.89      0.89      1849
```

This is a accuracy variable which we had to calculate to test the accuracy of this classification model. Following that it is put into a classification report allowing further analysis using evaluation techniques stated throughout this report.