Leo Toueg

16lsit@queensu.ca

20062982

251 Report: Some Results from Online Shopper's Intention Data Analytics

---

Outline:

I. Introduction:

As data analysts, we usually don't have control over how data are measured or collected. For this, sometimes we need to manipulate input data to make the data analytics more effective. More specifically, we learned in class about the Cross Industry Standard Process for Data Mining (CRISP), which usually goes as follows: Business understanding, Data understanding, Data preparation, Modeling, Evaluation, Deployment. In other words, understand the question, investigate the data, model, evaluate, and bring results forward. We were given data describing behaviors of users visiting an online shopping site. In this report, we will look at the results of trying to make the most accurate prediction of who will buy something, and the identification of properties that are associated with buyer likelihood.

II. Properties of the dataset/Observations:

First and foremost, we need to make sure that we understand the data we are going to be feeding into our models. In order to make the most informed decision on how to form a good model, we need to look at the data itself, and make sure a couple things check out. Initially, this might be done by looking at things like potential missing data, wrong data, sampling issues, and other considerations that affect model bias and our general ability to feed the data to algorithms of choice. In compliment to this, in removing as much as we can from our data, we might find it easier to clean, or optimize our data beforehand. This can be due to a variety of reasons, such as two attributes having overlap or redundance (correlation). However, before we dive into the considerations, let's clarify the 18 attributes classifying the 13330 entries and make some initial observations:

The 18 attributes are:
- **Administrative:** This is the number of pages of this type (administrative) that the user visited.
- **Administrative_Duration:** This is the amount of time spent in this category of pages.
- **Informational:** This is the number of pages of this type (informational) that the user visited.
- **Informational_Duration:** This is the amount of time spent in this category of pages.
- **ProductRelated:** This is the number of pages of this type (product related) that the user visited.
- **ProductRelated_Duration:** This is the amount of time spent in this category of pages.
- **BounceRates:** The percentage of visitors who enter the website through that page and exit without triggering any additional tasks.
- **ExitRates:** The percentage of pageviews on the website that end at that specific page.
- **PageValues:** The average value of the page averaged over the value of the target page and/or the completion of an eCommerce
- **SpecialDay:** This value represents the closeness of the browsing date to special days or holidays (ex Mother's Day or Valentine's Day) in which the transaction is more likely to be finalized. More information about how this value is calculated below.
- **Month:** Contains the month the pageview occurred, in string form.
- **OperatingSystems:** An integer value representing the operating system that the user was on when viewing the page.
- **Browser:** An integer value representing the browser that the user was using to view the page.
- **Region:** An integer value representing which region the user is located in.
- **TrafficType:** An integer value representing what type of traffic the user is categorized into.
- **VisitorType:** A string representing whether a visitor is New Visitor, Returning Visitor, or Other.
- **Weekend:** A Boolean representing whether the session is on a weekend.
- **Revenue:** A Boolean representing whether the user completed the purchase.

Source: *Online Shoppers Purchasing Intention Data Set, Sakar, C.O., Polat, S.O., Katircioglu, M. et al. Neural Comput & Applic , 2018*

As we can see, we have a very extensive list of attributes to look at. At first glance, this seems to be extremely valuable information, that if sampled correctly, may be valuable in determining purchasing tendencies in customers. For that reason, it is now important to consider if the data can be 'cleaned'. In practice, this is referring to the means by which we can considerably optimize input data in order to better extrapolate data from it by means of transforming or simplifying it. However, it is important to note that in attempting to reduce bias (which we hopefully are), we are introducing new biases in the equation.

First, I wanted to make sure that there were no empty data entries, so with the use of Python, I iterated through the data. This was done using the *Pandas* open-source Python Data Analysis Library, through which I was able to import the data and make a couple of observations:

Output: `Total number of null values in dataset: 0`

Reassuringly, there are no null values to deal with, which spares us from a lot of thinking and problem solving. Next, I looked at the number of unique values for each attribute, to get a sense of the data we are dealing with, and to familiarize myself with the data within each attribute.

Here were the results:

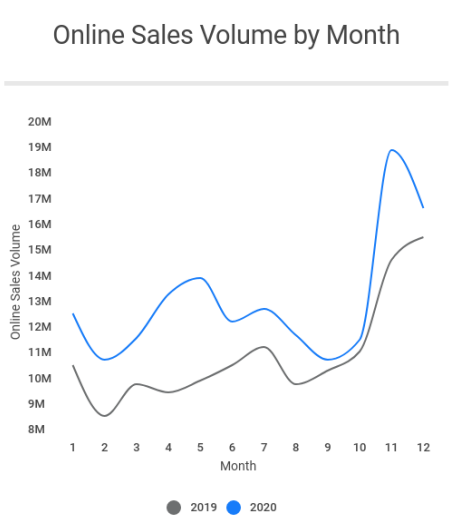
- Administrative:	27	**- Browser:**	13
- Administrative_Duration:	335	**- Region:**	9
- Informational:	17	**- TrafficType:**	20
- Informational_Duration:	1258	**- VisitorType:**	3
- ProductRelated:	311	**- Weekend:**	2
- ProductRelated_Duration:	9551	**- Revenue:**	2
- BounceRates:	1872	**- OperatingSystems:**	8
- ExitRates:	4777		
- PageValues:	2704		
- SpecialDay:	6		
- Month:	10		

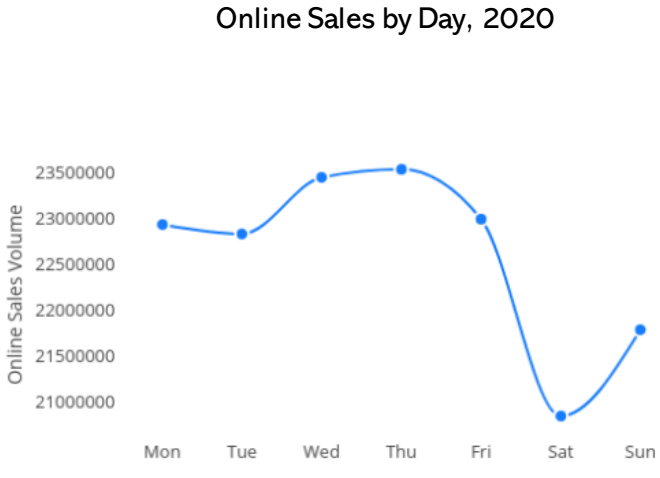
(Done with EXCEL function COUNT)

Looking at some of these numbers, a couple things stand out. For example, having data from only 10 out of 12 months under the Month attribute (itself being sampled unevenly, ex: 184 'Feb' records vs 3364 'May' records) indicates there may be some inherent data sampling biases. It is important to be aware of these subtleties, as they may point to some data cleaning techniques which will help us identity potential buyers through a better modeling process. For this reason, I chose to disregard the month attribute, and remove it completely. It is important to note that I am altering the original data, in hopes that I'll be able to better identify which attributes correlate to purchases, as the Month attribute seems redundant and will add a layer of difficulty to any model determining what inherently inclines a person to make a purchase in their internet session. Another reason this might be a good decision is because the notion of time-sensitivity may already be captured by the SpecialDay attribute.

Another two questionable attributes pop out to me at first. These are Operating System and Browser. Not only does Browser have more unique answers, but it is more unevenly sampled (ex: <1000 browser 13 users vs. >7500 browser 2 users). Intuitively, it seems rather unlikely that someone's Operating System influences their likelihood to purchase something in one browser session. Even if this attribute's information gain was significant, we wouldn't have enough data from other browsers for any results to have statistical significance. For that reason, those two attributes were also omitted from the original data.

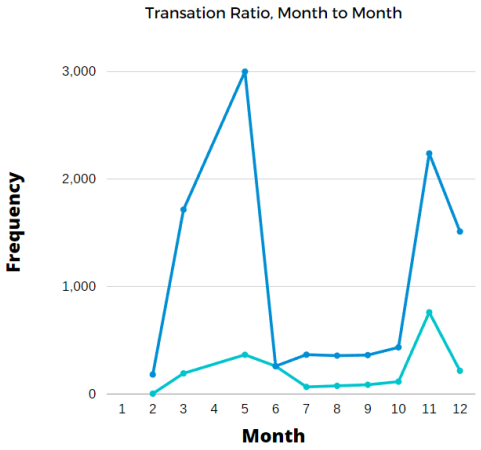
I also chose to not include TrafficType, or Weekend. I could not find very much information on TrafficType, and from what I was able to gather, traffic sources are not quite useful for classifying if a user will make a purchase. As for the Weekend attribute, I did some research on online shopping patterns and the correlation to days, months, etc... In 2019 Mondays and Thursdays were the most popular days of the week for online shopping and the peak time of day for ecommerce occurred between 8pm and 9pm. In 2020, Wednesdays and Thursdays were the best days for ecommerce sales and the peak hour for ecommerce was between 10am and 11am with a second peak at 8pm and 9pm. As we can see, this attribute seems rather volatile, and I think that attributes determining inclination to buy are more deep-rooted than that. In short, while it is true that for example there are consumer patterns, the essence of the timeliness is better captured by the SpecialDay attribute:





Source: https://www.salecycle.com/blog/stats/when-are-people-most-likely-to-buy-online/

If the Weekend attribute was a good indicator, it wouldn't give us a lot of data to look at. For example, there are only 1908 records in our data of a transaction being completed, versus 10422 sessions without a transaction completed. In trying to get a feel for any potential buying periods, I looked at the distribution of the ratio of purchase, month to month. So, for example: in February, there were 3 purchases and 181 non-purchases. This is a ratio of approx. 0.016. The two peaks over the whole year are in May (0.12) and November (0.339). The graph of the transaction ratio almost matches our research curve indicating online sales volume by month:



Dark blue curve: Non purchases

Cyan curve: Purchases

The dataset provided to us seems to confirm the trend that people are more likely to spend during those two peak periods. This might be relevant information for product and business owners relying on marketing for sales, and serves as good insight for us, in conceiving a model. In conclusion, not only does the data point towards online shopping being done on weekdays, but also that being a weekend/weekday doesn't affect purchases so much as does the trend of meaningful day in the year.

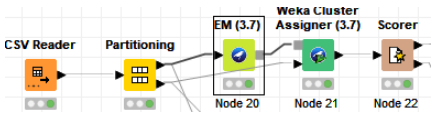
So far, we have removed columns from out input data, based on our understanding of their utility. We will now look at the sampling, this time from a numerical and statistical perspective. We have removed 'bad' attributes but are now looking to make our model as effective as possible. In order to remove any form of implied hierarchy, I will convert string labels to integers, then to 1-Hot encoding of the integer. This is a technique we discussed in class where we are changing categorical attributes to numeric ones. In our case, we will map the VisitorType to their equivalent 1-Hot values. For example, these values are x: [0 0 1], [1 0 0], [0 1 0] for n = 3. I did this using EXCEL, by reformatting my data, using integrated formulas.

III. Clustering/Predictors:

Now that we have our data in a more compact, mathematically readable manner, we can move on to trying to cluster it. From this, we can now attack the problem of trying to figure out which attributes contribute to the likelihood of a purchase within an internet session. Alternatively, we can gauge the statistical importance of each attribute and chose to remodel, leaving some out if it seems as though they do not provide sufficient information gain. I'm not sure to what extent to explain every concept, however we will remind that information gain is an indicator of how much Entropy is removed by forming a certain clustering decision. For example, a good clustering decision (rule) in a decision tree will greatly improve the purity (ratio of class labels) of the children of the node if it has good information gain. As a whole, the information we can get can be very useful, in

many ways. With this information, we can for example stratify our input, because it could be advantageous to a company to sample each subpopulation independently. This might have been an interesting path to take with the assignment, to see if you can build accurate models within each region, optimizing marketing for all subpopulations. First, I wanted to see just how accurate I could get my model, before chasing something that might be right between my eyes. It might however have been an interesting result, nonetheless. An important thing to note is that while the webpage outlining the report states that Clustering should be its own part, I will add it to this part of the discussion, as it didn't account for nearly as noteworthy results, and I will briefly break down what I did in terms of Clustering.

There are three main types of clustering: distance based, density based, and distribution based. I tried a distribution-based clustering first, using the following KNIME workflow, and in particular the Expectation Maximization node:

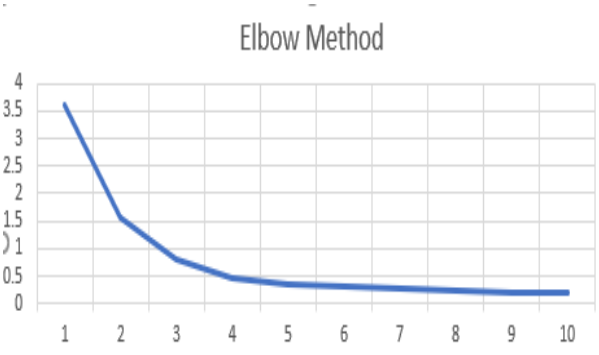


KNIME workflow

Revenue \ ...	FALSE	TRUE	1	0
FALSE	0	0	1505	592
TRUE	0	0	36	333
1	0	0	0	0
0	0	0	0	0

Confusion matrix (0 = predicted true)

The algorithm repeatedly found 2 clusters and has an accuracy of 74%. However, the precision and recall were not good. However, the two clusters were seemingly not related to purchasing probability, as they divided the data into a 60/40 split. This means about half the data was in one cluster and half was in the other. This was the general trend of every clustering algorithm I tried, whether it was distance, density or distribution based. I tried things such as imputing the number of clusters, changing the number of folds, etc.… It will be hard to find both: an interesting clustering result and a utility of the clustering to determine buying tendencies. I tried K-means, and throughout all my experimentation, each algorithm clustered the data in a different way, and it was hard to notice any patterns or useful results. For K-means, I used the Elbow method to determine the number of clusters to use. This is when you vary k (I did 1 to 10), and for each k you calculate the total sum of squares. The location of a bend (elbow) is usually considered a good number of clusters to use:

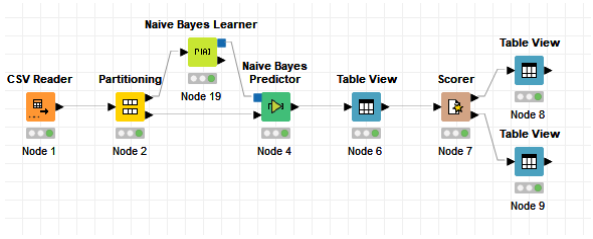


As we can see, the best number of clusters to use is 2. For reference, here are my K-means results:

```
kMeans
======

Number of iterations: 10
Within cluster sum of squared errors: 7406.379480500534
Missing values globally replaced with mean/mode

Cluster centroids:
                                        Cluster#
Attribute                 Full Data           0           1
                            (9864)      (7008)      (2856)
=============================================================
Administrative               2.3026      2.5271      1.7518
Administrative_Duration     80.3203     89.5795     57.6001
Informational                  0.5      0.5574      0.3592
Informational_Duration      33.3989     37.5404     23.2367
ProductRelated              31.8217     34.5078     25.2304
ProductRelated_Duration   1200.4479   1309.2405    933.4948
BounceRates                  0.0219      0.0141       0.041
ExitRates                    0.0428      0.0337      0.0653
PageValues                   6.0428      7.0061      3.6791
SpecialDay                   0.0605      0.0391      0.1131
Region                    100000000   100000000   001000000
Revenue                       FALSE       FALSE       FALSE


Clustered Instances

0      1729 ( 70%)
1       737 ( 30%)
```

As we can see, this one gave us a 70/30% split. All in all, I didn't manage to engineer a good clustering method, and no significant results came of it. Another reason this was the case is because clusters might not be as relevant to our specific question as they could be. This is because we already 'know' what we are looking at and what we are looking for.

When it comes to predictors, the first thing I tried was a naïve Bayes' classification. It is important to distinguish classification and clustering. Although both techniques have certain similarities, the difference lies in the fact that classification uses predefined classes in which objects are assigned, while clustering identifies similarities between objects, which it groups according to those characteristics in common and which differentiate them from other groups of objects. These groups are known as "clusters". In our case, we are really only interested in whether the person is likely to make a sale or not, so no other classifications or clustering pop out at me initially. This is not to say there don't exist any underlying relationships in the data that we aren't aware of. I'm not sure whether to focus more on the implementation or the results, but this is something we did in prior weeks on KNIME. Out of caution, I will briefly go over the theory and integration of the Naïve Bayes' algorithm. It essentially works on Bayes' theorem of probability to predict the class of unknown data sets, by assigning conditional probabilities to each attribute. It is called the naïve Bayes' classifier because it makes the 'naïve' assumption that all attributes are independent of each other, in order to make statistical inferences. There are three types of Naïve Bayes Classifiers: Multinomial, Bernoulli and Gaussian Naïve Bayes. I highly expect this algorithm to be outperformed by others such as random forests, but I wanted to get a benchmark. I did this using KNIME, and the following workflow in particular:



KNIME Workflow for Naïve Bayes'

I couldn't figure out how to use the cross-validation nodes in KNIME, so I just ran each test that I will present many times over, and I will discuss the results*. For the (Bernoulli) Naïve Bayes' algorithm, the results were not great, as expected. I used a standard 80/20 training and testing data split, then ran the model 100 times. I noted an average accuracy of 83%, with the highest in 100 models being 85%. As I was more interested in finding a model with a better accuracy, I won't go into too much unnecessary detail on the results, for which I didn't achieve a very good result. In general, it is important to look not only at a model's classification accuracy, but also consider things such as the confusion matrix, because you can understand where the classification model is and isn' right, and what type of errors it is making. For example, in the medical world, we want to make sure our model isn't making for a lot of Type 2 (False Negative) errors. This is when you predict or guess something to be negative (such as cancer) while it is positive. For this reason, I looked at the confusion table, which looks normal:

Revenue \ ...	FALSE	TRUE
FALSE	1800	263
TRUE	164	239

Concentration matrix for (Multinomial) Naïve Bayes'

The results are not very interesting, but essentially the model is somewhat accurate, and we don't have a sense of what attributes are important. Before moving on, as part of the workflow, KNIME displayed the gaussian distribution for each attribute. I decided to learn a little bit more about Gaussian naïve Bayes' and see if I could try implementing it. The idea behind this one is that instead of thinking of training data as discrete, we look at it continuously, and apply probabilistic models on data. As such, the assumption is still that the continuous values associated with each attribute are distributed according to a normal (or Gaussian) distribution. Essentially, we calculate two heuristics, and take the larger one:

P (purchase= true| evidence) = P (purchase = true) * L (evidence(1) | purchase = true) *...* L (evidence(n) | purchase = true)

P (purchase= false| evidence) = P (purchase = false) * L (evidence(1) | purchase = false) *...* L (evidence(n) | purchase = false)

*where L is the likelihood vs P is the probability

The way one estimates P (purchase = true) is up to the individual, but I chose to set it to the initial proportion of data entries where a purchase was made because it seems to make the most sense in this situation, as we have collected a lot of data. Here are the results I got from trying out Gaussian Naïve Bayes', notably a model accuracy of 84%, and the following concentration matrix:

Revenue \ ...	FALSE	TRUE
FALSE	1808	288
TRUE	148	222

Concentration matrix for Gaussian Naïve Bayes'

Let's see how this one relates to the previous model. As we can see, the data is generally more properly classified, however there are a little more Type 2 errors than before. We however notice the slight difference in Type 1 errors. The recall is of 54%, which isn't the greatest. This means that out of all the positive classes, we predicted about 54% of them correctly. The precision is about 70%, which means that from all the classes we have predicted as positive, about 70% are indeed positive. While these results were interesting and compelling, I felt as though they didn't paint the whole picture, and there was more to be discovered. One thing I tried doing at this point of the assignment was to retest both algorithms with stratified training data. Since the training data is so heavily skewed in the 'no purchase' class, an idea could be to stratify the training data so that the ratio of records that have a purchase is equal to the ratio of non-purchases. Unfortunately, I had a lot of trouble implementing this, and have no definite results to report.

The last algorithm I successfully implemented was the Random Forest Classifier. This is because out of all the ones we learned, I really enjoyed seeing how this one worked, as well as its general performance. They are made from decision trees, where each tree in the random forest spits out a class prediction and the class with the most votes become our model's prediction. This works because many relatively uncorrelated models (trees) operating as a committee will generally outperform any of the individual constituent models. We basically make a lot of decision trees from bootstrapped training data with a preselected number of attributes and look at the consensus amongst all the trees in the forest. For the purposes of this assignment, I made my forest 1000 trees large. I ended up using GINI Index for the Random Forest instead of Information Gain or Information Gain Ratio, since this performed slightly better. Here are the results from my Random Forest Model, ran with an 80/20 split, as well as 10-fold cross validation:

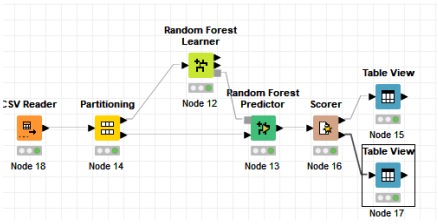


Figure 1: (basic) KNIME Workflow

Revenue \ ...	FALSE	TRUE
FALSE	2018	82
TRUE	165	201

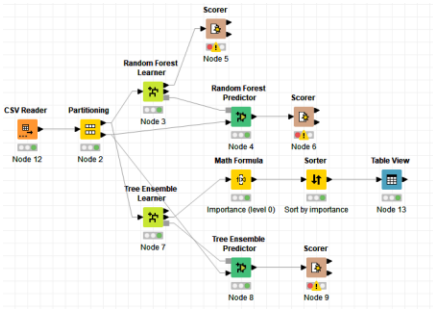
Figure 2: Confusion matrix

As we can see, these results immediately look more promising. For starters, the model has an accuracy of 90%, which is honestly awesome. It is important to keep in mind that this number is the result of us 'tweaking' and extracting information from our original data set, and to remain cognisant of the biases that may have been introduced into the model. Surprisingly, the recall and precision remained almost equal (about 54% and 72%, respectively). This is because in our data, simply did not have a good ratio of 'true' and 'false' classification, and our testing data reflects this. This is why I tried to implement stratified sampling of bootstrapped data, with a stronger 'purchase made' ratio. This could have hopefully improved the precision. Unfortunately, I couldn't get it to work, but I project it would have done slightly better than the current one. Another thing I'd like to retroactively add is that I would've liked to try 1R on each attribute, and a little more of the 'naïve' solutions, but I didn't want to waste too much time.

IV. Attribute selection and ranking:

Looking at everything we've done, I think it was helpful to remove redundant attributes, as they can make for a lot of noise in complicated data. Since the Random Forest predictor was the most accurate out of everything I did, I assessed the relevance of attributes predominantly by using

the attribute importance method, and experimentally looking at Gini importance. This is something we learned in class (Particularly Slide34 material) and that I implemented using KNIME:

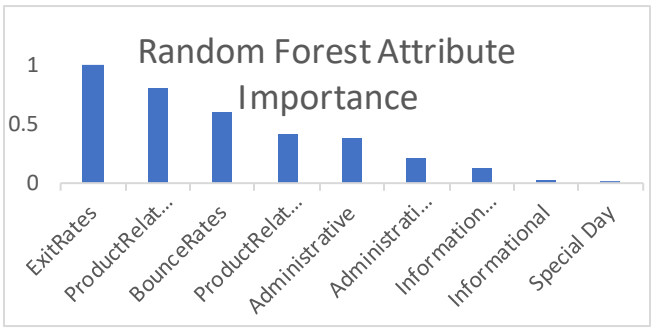


KNIME Workflow

RowID	#splits (level 0)	#splits (level 1)	#splits (level 2)	#candidates (level 0)	#candidates (level 1)	#candidates (level 2)	importance (level 0)
PageValues	274	495	837	274	547	1093	1
ExitRates	217	335	582	268	579	1099	0.8097014925373134
ProductRelated_Duration	156	276	549	255	548	1154	0.611764705882353
BounceRates	124	239	420	295	563	1117	0.420338983050847 44
ProductRelated	112	229	453	291	561	1074	0.3848797250859107
Administrative	62	170	343	275	534	1065	0.22545454545454546
Administrative_Duration	35	126	286	267	528	1057	0.13108614232209737
Informational_Duration	11	60	170	298	515	1130	0.036912751677852 35
Informational	8	45	131	252	512	1102	0.031746031746031744
SpecialDay	1	23	88	282	556	1026	0.003546099290780 1418

Attribute statistics table

Here is how the statistics table works: "#splits (level x) as the number of models, which use the attribute as split on level x (with level 0 as root split); #candidates (level x) is the number of times an attribute was in the attribute sample for level x" (Quote from KNIME Node descriptor). Looking at this table, it seems extremely unlikely that an attribute has importance 1, but the results might be interesting to look at. For example, the attribute with the least importance is Region. This seems to make sense intuitively. However, the importance level of SpecialDay is very low, and this tells me that the results are to be taken with a grain of salt. Here is a graph version of what I found:



As we can see, the attributes having an importance of 5 or more % are: Page Values, Exit Rates, Product Related Duration, Bounce rates, Product Related, Administrative and Administrative_Duration. I tried seeing if this was consistent with the GINI importance. Essentially, when an attribute x is used to split within a tree, the Gini measure is smaller going down the tree than that of the node itself. You add up all the Gini increases for every use of x to split in any tree. The idea is that the greater the total increase, the greater the importance of this attribute. Unfortunately, I couldn't implement it successfully.

In conclusion, if I were to present this data to a website owner looking to better identify trends in his customers, I would tell them that the five most important attributes in descending order are PageValues, ExitRates, ProductRelated_Duration, BounceRates and ProductRelated. I would also justify my exclusion of attributes such as Month, as there is more data pertaining to timeliness of online purchases, and more evidence pointing to certain expenditure trends. If I were to keep trying to find a better model, I would start by exclusively looking at these five attributes, as it seems that

they account for most of the variance in the classification. For reference, here are the distributions of these attributes:

Gaussian distribution for PageValues per class value

	FALSE	TRUE
Count:	8355	1509
Mean:	1.99883	26.90122
Std. Deviation:	8.97391	35.80866
Rate:	85%	15%

Gaussian distribution for ExitRates per class value

	FALSE	TRUE
Count:	8355	1509
Mean:	0.0469	0.01956
Std. Deviation:	0.05069	0.01647
Rate:	85%	15%

Gaussian distribution for ProductRelated_Duration per class value

	FALSE	TRUE
Count:	8355	1509
Mean:	1068.3913	1811.78338
Std. Deviation:	1753.15428	2200.3615
Rate:	85%	15%

V. Actionable conclusions:

It is important to remember the assumptions and the groundwork we did, before arriving at any conclusions. First and foremost, it is important acknowledge the amount of bias that may have been introduced into the model. We did do our best to minimize bias, but every person working on this data will do different things with it, and it is important to remember that while you may observe interesting trends, the results you may find are not fact, but rather anecdotal. There are many things we can take away from our data. For example, having a high PageValue is tied to higher likelihood of purchase, however it is not clear what the cause/effect is, looking at what PageValue is. On the other hand, the data shows that a smaller ExitRate is more likely to relate to more purchasing, as the ExitRate mean is almost twice as large for non-purchases as it is for purchases. However, the variance is slightly larger so it a bit more nuanced. All in all, if you can influence PageValue then that is most likely the optimal marketing strategy to adopt for someone to be more likely to purchase something. It is unclear to me what exactly PageValues are but looking at the importance of each attribute can give an idea of what influences decision making, and I would tackle PageValue first.