Predicting LTV and Repeat Customers for Shopify Stores with Unsupervised Machine Learning

FEBRUARY 10

Created for Shopify Store platforms
Authored by: Benjamin N. Bellman

Predicting LTV and Customer Repeat

By Benjamin Bellman

Table of Contents:

Special Thanks to Lou Graniou & Ben Bell

"There is only one boss. The customer. And he can fire everybody in the company from the chairman on down, simply by spending his money somewhere else." – Sam Walton

1. Introduction :

E-commerce as of 2021 is worldwide, a $4.89 trillion industry. One of the leading platforms in the marketplace is Shopify which currently holds a 23% market share in the United States with over 1.7 million live stores. For perspective, that is about 1 store for every 200 Americans! With e-commerce expected to keep growing, it is valuable for Shopify storeowners to get the most out of the available data the platform provides. This project will seek to determine customer behavior of a fitness apparel store that runs on the Shopify platform. We will first aim to predict *post-first week customer spending* of a one-year Customer Lifetime Value (LTV) period, then turn our attention to predict whether customers *repeat orders past their first order week*, using supervised Machine Learning Algorithms.

We'll seek to understand which factors influence these two target variables and use the information collected to deliver data driven recommendations on what actions can be taken to increase Lifetime Value (LTV) and decrease Customer Acquisition Costs (CAC). This Report will walk you through the steps taken. In the case you are a Shopify store owner, you can replicate this for your business! The entire code can be found on the GitHub Repository here.

What is LTV ?

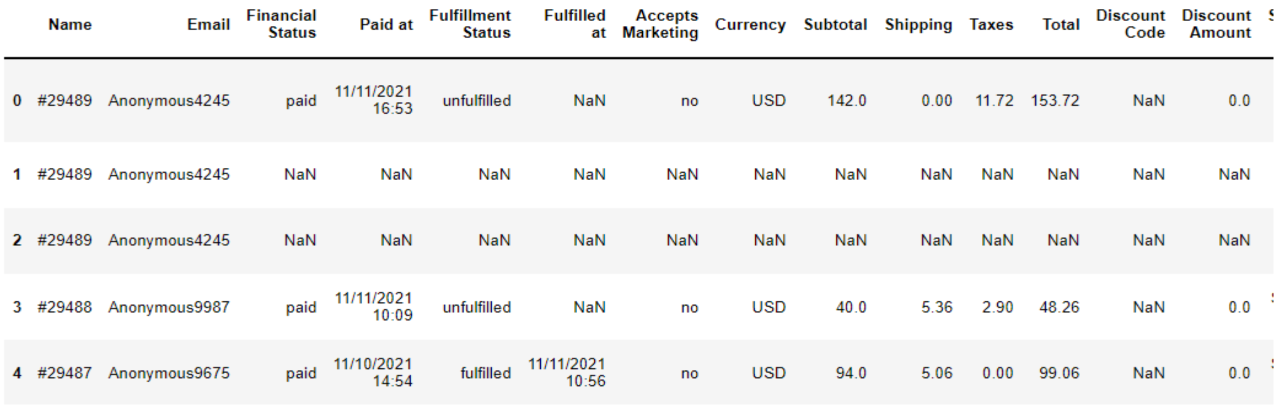
Qualtrics defines CLTV as "the customer's total worth to a business over the whole period of their relationship." It is a measurement of unit economics that allows for better decision-making on CAC, improved forecasting, and scaling. In theory, the lifetime value would be over the entire course of the relationship the customer has with the store. In practice, LTVs are generally calculated in intervals. Depending on our business strategies, we may want to calculate a short term 3-month CLTV or 1-year CLTV. For more information, I recommend getting more information with this article.

Important Heads-Up

Naturally, LTV for any business can be very difficult predict. The results we obtained for the post first-week customer spending were not conclusive. We proceeded to switch to a classification problem of whether customers made an order past the first week which yielded better results. This will be explained in greater detail in the Modelling and Conclusion sections. With that said, there were many pertinent findings that helped this business which will be discussed!

2. Data Exploration & Data Wrangling

Shopify data is usually standardized when orders are exported in csv format. In this dataset, the original raw export is organized with each row representing an item sold and 73 columns which provide information on the transaction. Most of these will not be useful for this analysis and will be dropped early on. Here is a quick preview of the first couple of rows and columns:



	Name	Email	Financial Status	Paid at	Fulfillment Status	Fulfilled at	Accepts Marketing	Currency	Subtotal	Shipping	Taxes	Total	Discount Code	Discount Amount
0	#29489	Anonymous4245	paid	11/11/2021 16:53	unfulfilled	NaN	no	USD	142.0	0.00	11.72	153.72	NaN	0.0
1	#29489	Anonymous4245	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	#29489	Anonymous4245	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	#29488	Anonymous9987	paid	11/11/2021 10:09	unfulfilled	NaN	no	USD	40.0	5.36	2.90	48.26	NaN	0.0
4	#29487	Anonymous9675	paid	11/10/2021 14:54	fulfilled	11/11/2021 10:56	no	USD	94.0	5.06	0.00	99.06	NaN	0.0

The data used for this analysis comes from a real company which generated $1.8 million in YTD revenue and has been in operation since 2018. For our store, our dataset contained **50,418** records each representing an item sold.

Featured columns include:

- *Name*: Refers to the Order Id. It is not unique, as there can be multiple items in one order. This column should not contain any missing values.
- *Email*: The email of the customer, which is used to identify them (*Anonymized here)*
- *Paid at*: Refers to the date the purchase occurred. Only one date for each order, missing values are expected. Other columns follow this logic.
- *Subtotal*: Refers to the total for each order before taxes.
- *Line-item name*: Name of the product item. Typically, a store sets up the product name where we can get information such as the size, type, and color of the product in question. This column should never contain missing values
- *Line-Item Quantity and Line-Item Price*: How many items purchased and their price. These shouldn't contain missing values.

Some of the columns are blank. Anonymous 4245 purchased 3 items, 2 the rows below the first will contain blank values except for the item purchased column, since they would contain the same information in the first row. If a Shopify store is not set up properly, it may result in possible missing values or incomplete transactions (as was the case for this company). The analysis will clean some data but assumes that there are no missing values in the *Created at*, *Line-item name*, and *Line-Item Price* columns as all of these refer to each individual item sold. If there is missing data in any of these rows, they need to be dropped and discarded. For a more detailed guide on all 73 columns, please check out Shopify's documentation here. The data has been modified from its original version to protect the anonymity of the company as well as the information of all its customers.

The raw data file which we will start with has gone through the following alterations:

- *Billing Name, Billing Street, Billing Address1, Billing Address2, Billing Company, Billing Phone, Shipping Name, Shipping Street, Shipping Address1, Shipping Address2, Shipping*

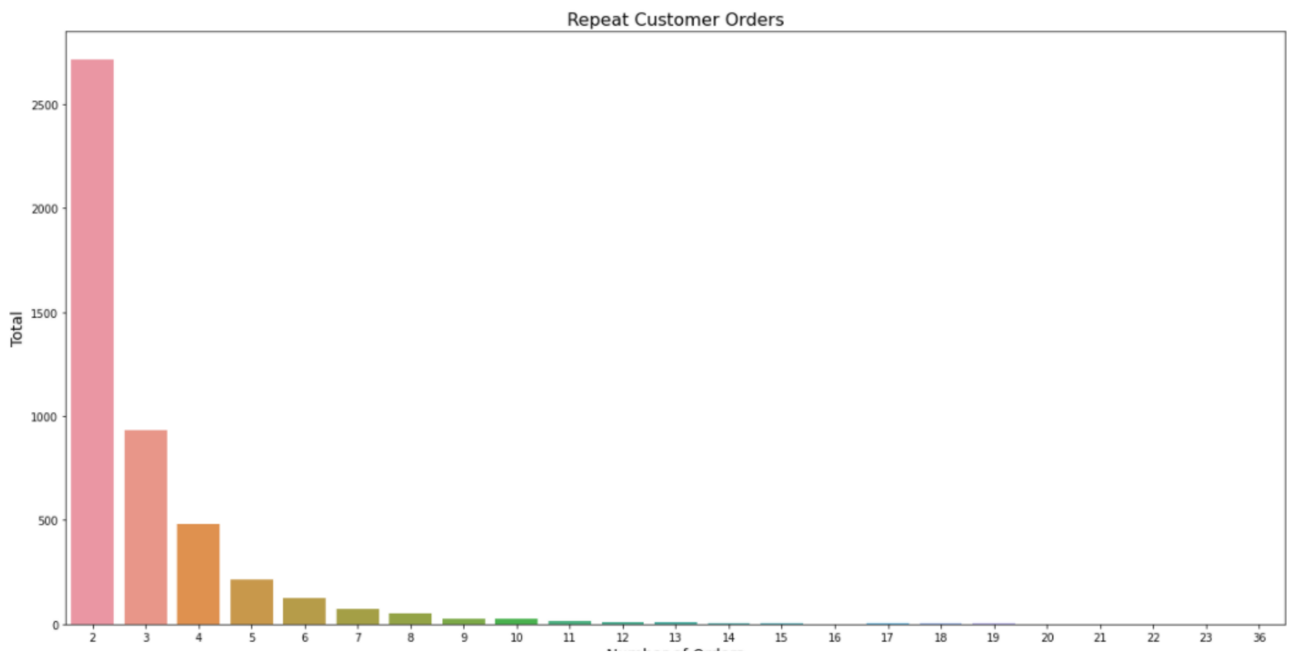
Company, Shipping Phone have all been replaced with null values. These columns will not be used in the analysis and dropped early on shortly after the introduction
- Values in the *Email* column have been replaced with a unique id with Anonymous + number for each customer instead of their actual emails.
- Some of the *Line-item name* items have substituted product names.

Let us explore our data a little bit. After performing a couple of operations to clean up our data, we created 3 separate dataframe, organized by unique customer, order, and item. Let's look at some key statistics:

- There are **16,180** unique customers.
- There are **25,188** unique orders.
- There are **560** unique items.
- Out of all our customers, ***4556*** were repeat (*28%*)
- Out of all our orders, ***13,565*** were made by repeat customers (*53%*)
- Our cheapest item was **$12.50**, our most expensive was **$200**.
- Our most popular item was *Black/Mint Contour Leggings–M*, with **1042** purchases.

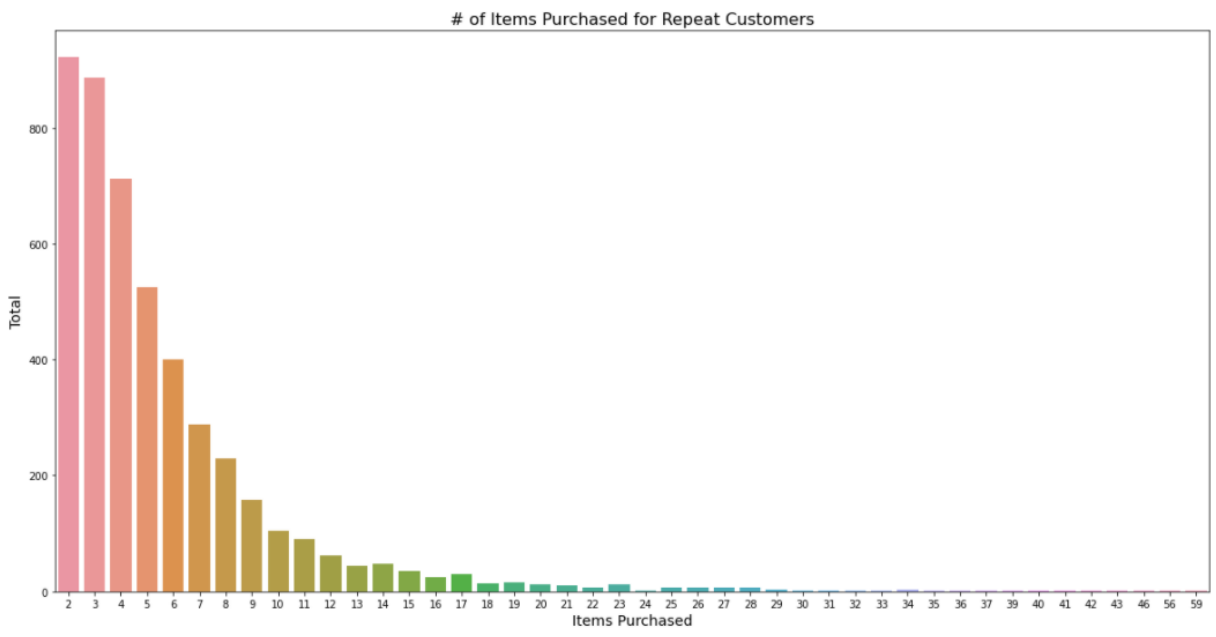
This Shopify store specializes in women's workout fashion. The majority of our orders come from repeat customers. A little more than a quarter of our customers are repeat, which is on par with the e-commerce industry standard of 20%-40% depending on the nature of the business. While a decent repeat rate, we will see if our analysis can gain any information to try and increase this KPI. A business can always benefit from understanding what drives their customers to repeat purchases as to minimize CAC and increase LTV. In this respect, we want to try and understand the behavior of the repeat customers Let's take a look at the distribution of orders for repeat customers:

Number of Orders for Repeat Customers:



It appears over half of repeat customers are making only 2 orders, with a slight minority making 3 and 4. Understanding what drives a customer from going from 2 to 3 orders would be extremely beneficial to increase, however it is outside the scope of this analysis. In addition, the limited datapoints for this dataset could be problematic. So far, we know that our repeat customer is likely to make 2 orders. Now let’s look at the number of items purchased for repeat customers.

Number of Items Purchased for Repeat Customers

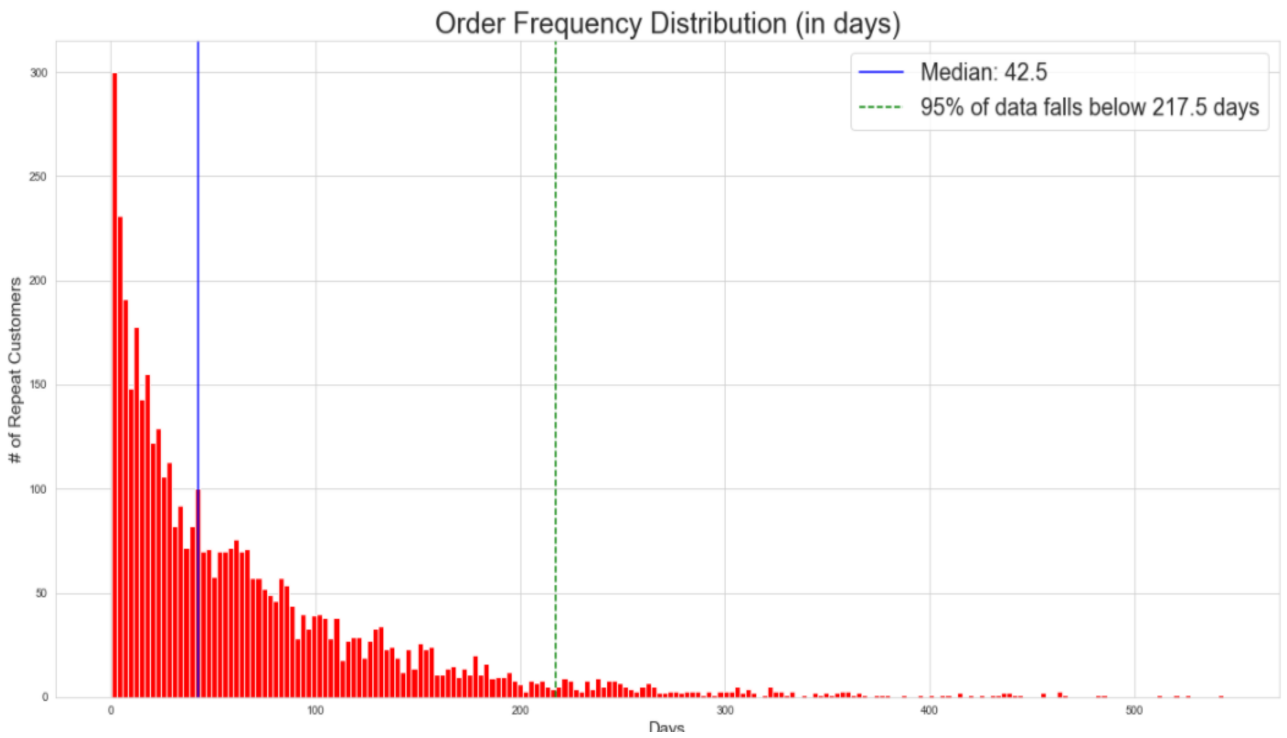


The decline in number items purchased is more progressive than what we observed with the orders. Repeat customers are virtually just as likely to purchase two or three items. Most of them are purchasing between **2-6** items. It even appears we have couple of super-fans who are purchasing 6+ products, which is amazing! However, one hypothesis is that a good portion of them might be resellers. While we do not have data available on this, it is likely that this explains some of the 15+ items purchased.

Now that we know a bit more about the number of items a repeat customer purchases and how many orders they make, it would be interesting to study how often they make an order by getting their order frequency. This is calculated by getting the amount of time has elapsed between a customer’s first order and their most recent order then dividing the number of total orders they made by that time.

While this would not exactly measure how much time elapsed between a customer’s first order and their second order, it can give us an idea of the distribution and give us an idea of a point are they likely not going to order from our store again. We create a visualization which shows the order frequency distribution in days, with number of days on the x axis.

Distribution of Order Frequency



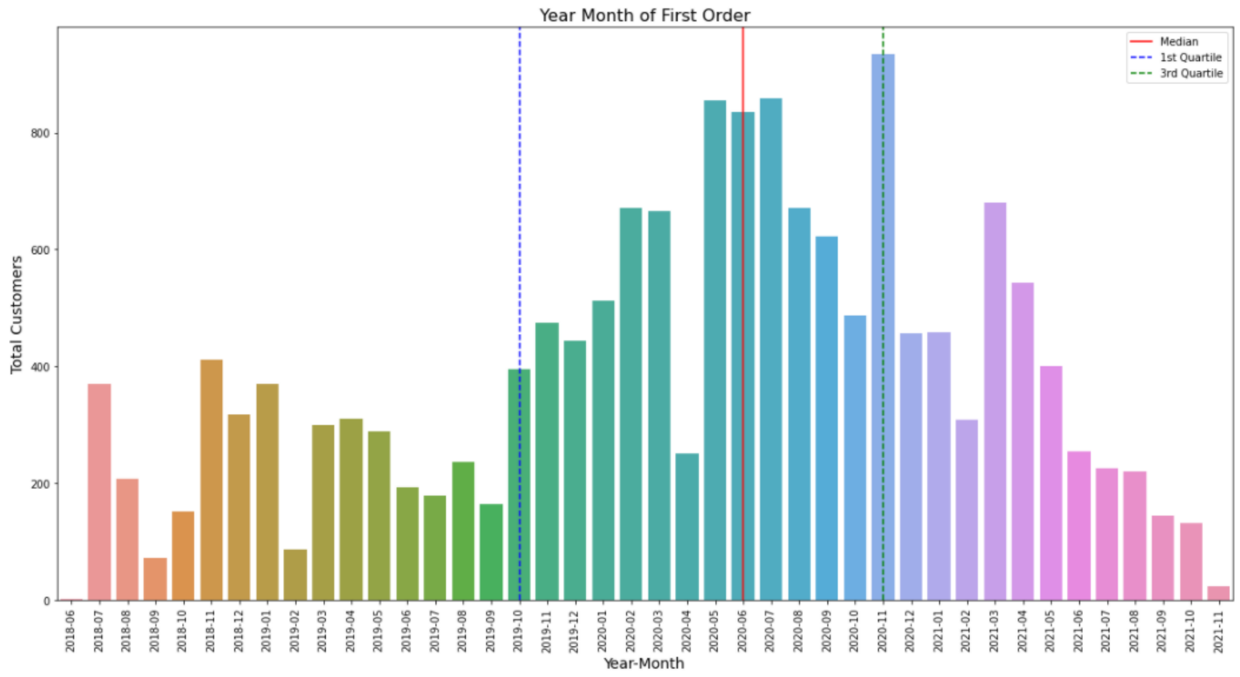
From what we can see, the median order frequency for repeat customers is **43** days. Only 5% of repeat customers made another order after **218** days, indicating that if they don't purchase before then, then it is very unlikely that they will purchase again from our store.

This visualization is very useful as it is showing us that most of our customers will make an order within a year. If they don't, we likely will lose them forever and will have to spend more in CAC to generate a new customer. Given that it is more expensive to acquire a new customer than simply attempting to retain one, if we know they likely will not make a new order after 218 days it makes sense to offer them a large 50% off coupon after that time to entice them to make another purchase (which is one of the recommendations our team made to the founder).

Why is this relevant to LTV? Well, in addition to giving us an indication on when these customers "churn", we need to define a period for which we want to conduct our LTV analysis, but we still don't know when our store started getting traction and when it got most of its first customers. Picking an interval where we would have the best data is critical, especially for our Machine Learning algorithms where we don't want to throw too much data away.

So, let's look at when customers made their first order since the inception of this Shopify store.

Distribution of First Orders by month since June 2018 .



We observe that **75%** of first-time orders were made between inception (June 2018) and November 2020. While most orders , there seems. Interestingly, for the time we have available data, for this data, the distribution of new customers approaches a normal distribution, with fewer customers at the tails and higher, at least for this interval. It is interesting as we can see the 4-product lifecycle and its phases illustrated here: *Introduction* from 2018-06 to 2019-09, *Growth* until 2020-06, *Maturity* until 2020-12 and *Decline* until 2021-10 (November was incomplete). The reason first time customers have been steadily decreasing since mid-2021 is that the owner decided to focus on a new venture during this time. Since we already know that most customer will not make an order after a yea, we can use the 1-year interval as the period threshold for our analysis and we would only be losing ~ 25% of our data.

Here are the steps to take:

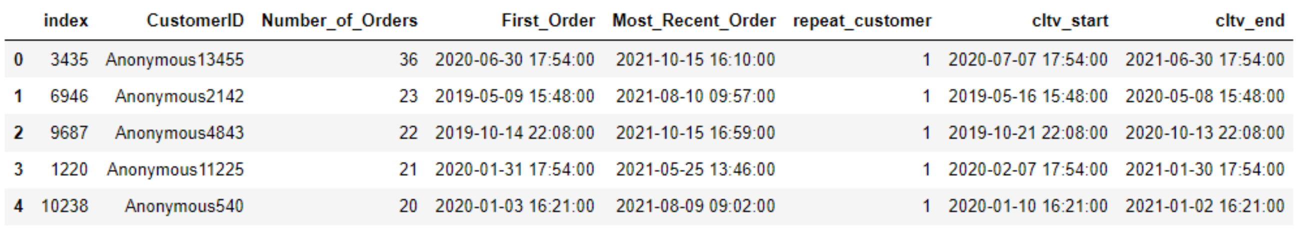
- We will need to create a dataframe organized by customer with their CLTV period.
- Our last order date in the dataset is ***2021-11-11 16:53***
- Customers need to have made their first order at least 1 year before that date.
- Customers who made their first order after ***2020-11-11 16:53*** will be discarded.

We could just as well have done a 3-or 6-month interval and kept more data, however, using the 1-year threshold, we are discarding only 25% of our original data. If we were losing significantly more, this would not be possible as Machine Learning Models do best with more data. There are tradeoffs to choosing shorter vs longer intervals. For SaaS companies on subscription models, it may make more sense for shorter periods when tracking churn, but for a fitness apparel store where a customer may not purchase monthly, a one-year interval makes more sense.

Now that we defined our LTV period, we continue with additional data wrangling and reshape the dataframe. We create a unique customer data frame where for each customer, we get their *First Order* date, *Most Recent Order*, *LTV start* and *End* dates.

We define *LTV start* as one week after their *First Order* date. **Therefore, we will be trying to predict customer behavior for the next 358 days following the week of their first order**. If they made an order beyond the 358 days, for the purposes of our 1-year LTV definition, they will not be considered repeat.

Customer Dataframe with LTV Start and End Dates



	index	CustomerID	Number_of_Orders	First_Order	Most_Recent_Order	repeat_customer	cltv_start	cltv_end
0	3435	Anonymous13455	36	2020-06-30 17:54:00	2021-10-15 16:10:00	1	2020-07-07 17:54:00	2021-06-30 17:54:00
1	6946	Anonymous2142	23	2019-05-09 15:48:00	2021-08-10 09:57:00	1	2019-05-16 15:48:00	2020-05-08 15:48:00
2	9687	Anonymous4843	22	2019-10-14 22:08:00	2021-10-15 16:59:00	1	2019-10-21 22:08:00	2020-10-13 22:08:00
3	1220	Anonymous11225	21	2020-01-31 17:54:00	2021-05-25 13:46:00	1	2020-02-07 17:54:00	2021-01-30 17:54:00
4	10238	Anonymous540	20	2020-01-03 16:21:00	2021-08-09 09:02:00	1	2020-01-10 16:21:00	2021-01-02 16:21:00

The last section of the data wrangling will be to create a dataframe that will contain independent variables and see if they play a role on our target variables, *post first week* spend and *order_pfw*.

First week features (Independent Variables) :

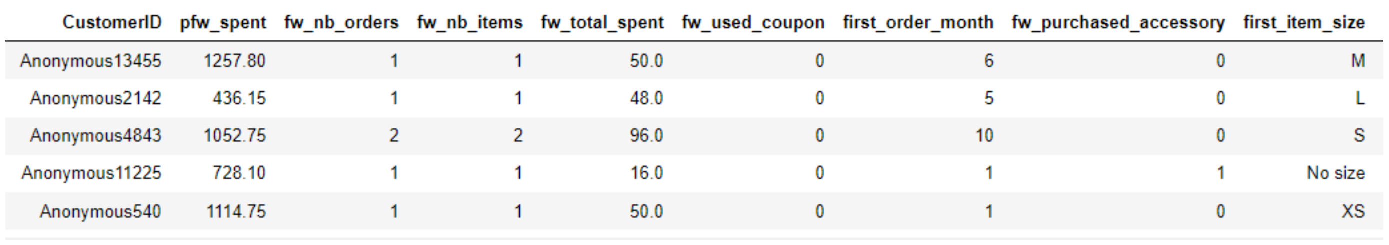
- **fw_nb_orders**: (int) total orders made by the customer in the first week (fw).
- **fw_nb_items**: (int) total items purchased in their first week.
- **fw_total_spent**: (float) total spent by the customer in their first week.
- **fw_used_coupon**: (bool) *True* if the customer used a coupon in the first week.
- **fw_purchased_accessory**: (bool) True if customer only purchased an accessory in fw.
- **location**: (string) within {'Northeast', 'Midwest', 'South', 'West'} - this is assuming all customers are from the US (not always the case) and assuming they did not move over the time spent on the platform, which was separately verified to be true for this dataset.
- **in_wealthiest_zipcode**: (bool) *True* if customer is in the top 1000 wealthiest zip codes.
- **first_item_size**: (string) within {'XS', 'S', 'M', 'L', 'XL', 'NA'} size of first item purchased. NA is for items that are not clothing. If items of multiple sizes are purchased in the first order, pick the most chosen size (for example if 2 'M' items and 1 'S' item are purchased, categorize that as 'M'). If there is a tie, pick randomly (very rare)
- **accepts_marketing**: (bool) *True* if customer opted into marketing emails.
- **first_order_month**: (string) {'1' to '12'} month of customer's first purchase.
- **first_purchase_price**: (string) {'low', 'medium', 'high'} based on the price of the most expensive item in the first order made by the customer. Prices are split in 3 buckets where 'low' is for a price within {< $35}, 'medium' is the {$35,$45} range, and 'high' is the {>$45} range.

Target features (Independent Variables) :

- **pfw_total_spent (For Regression Problem)**: (float) amount spent between the end of the first week and the end of the 1-year CLTV period by the customer. This is the quantity we are aiming to predict.
- **repeat customer (For Classification Problem)**: (int): Binary outcome, where 1 indicates the customer repeated an order and 0 indicates they were not a repeat order.

We proceed to do some more wrangling and build our final dataframe, with these features.

Final 1 Year LTV Customer Interval Dataframe with Features



CustomerID	pfw_spent	fw_nb_orders	fw_nb_items	fw_total_spent	fw_used_coupon	first_order_month	fw_purchased_accessory	first_item_size
Anonymous13455	1257.80	1	1	50.0	0	6	0	M
Anonymous2142	436.15	1	1	48.0	0	5	0	L
Anonymous4843	1052.75	2	2	96.0	0	10	0	S
Anonymous11225	728.10	1	1	16.0	0	1	1	No size
Anonymous540	1114.75	1	1	50.0	0	1	0	XS

We get the following valuable information:

- Median 1-year LTV (first week spend + post first week spend): **$69**
- Mean 1-year LTV: **$102.70**
- Median First Week Spend: **$50**
- Mean First Week Spend: **$66.42**

Now let's get some key information on our target variable:

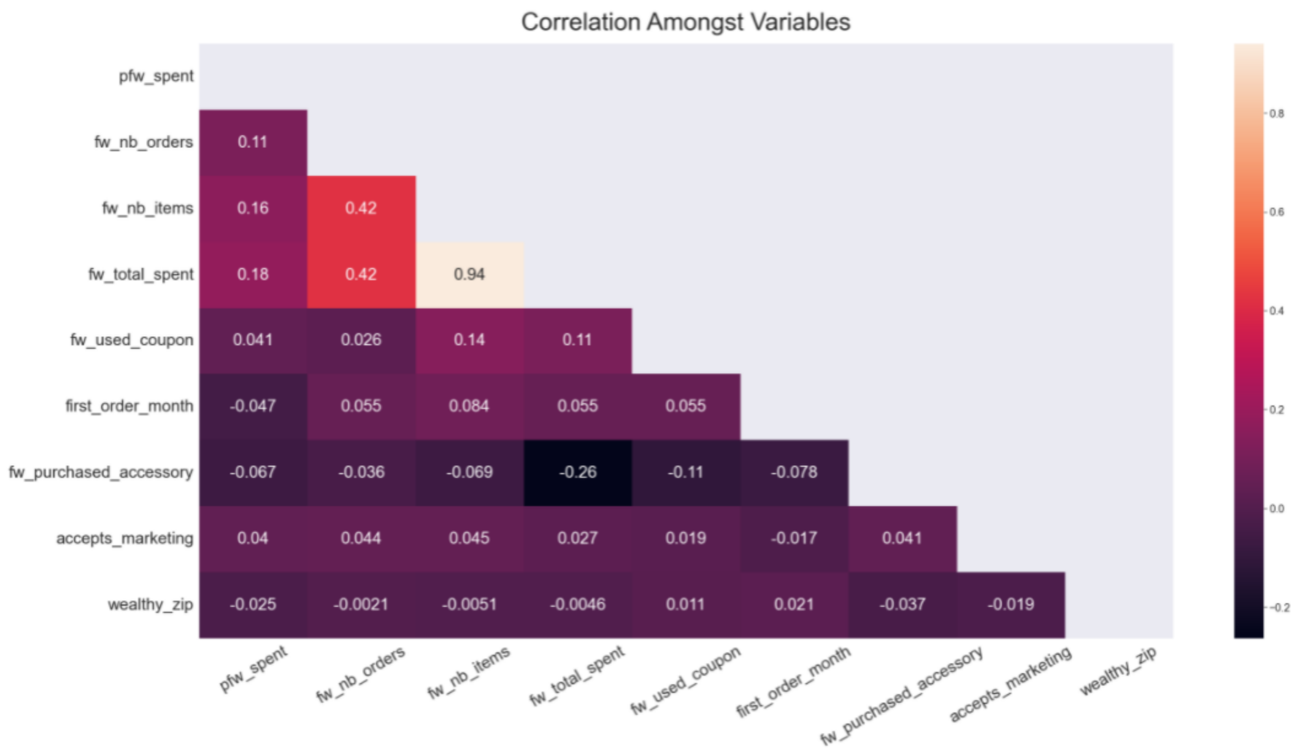
- Median *post first week* spend : **$0**
- Average *post first week* spend : **$36.26**

With respect to our target variable, the dataset is highly skewed towards the right (which makes sense, most customers are not repeat), something to keep in mind for the EDA.

3. Exploratory Data Analysis (EDA)

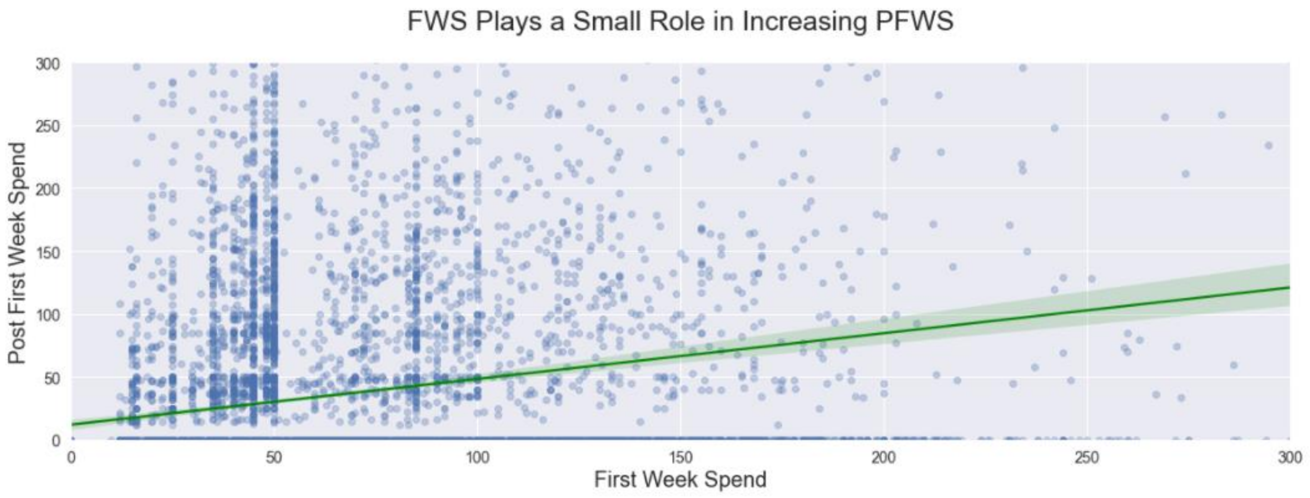
We are now ready to proceed to explore the relationships between these variables. We start the EDA portion by creating a heatmap that shows the correlation amongst the variables.

Pearson Correlation Amongst Variables:



Our target variable, *pfw_spend* is found in the first column. We can see that the two variables which seem to have the strongest R coefficient of correlation are *fw_total_spent* and *fw_nb_items* with 0.18 and 0.16 respectively. These two variables highly correlate with each other at 0.94, indicating that one of them should be dropped when we get to the modelling section. The *fw_nb_orders* variable comes in third and also has a slight positive role on the post first week spend. It can also be seen that *accepts_marketing* and *fw_used_coupon* both have a very small positive correlation, and *first_order_month* has very small negative correlation. This heatmap gives a pessimistic outlook for modelling as it does not seem there were any good variables that are very good predictors of post first week spend. Such are the limitations of the Shopify data. Now that we have a preview of the correlations, let's take a deeper look at our variables.

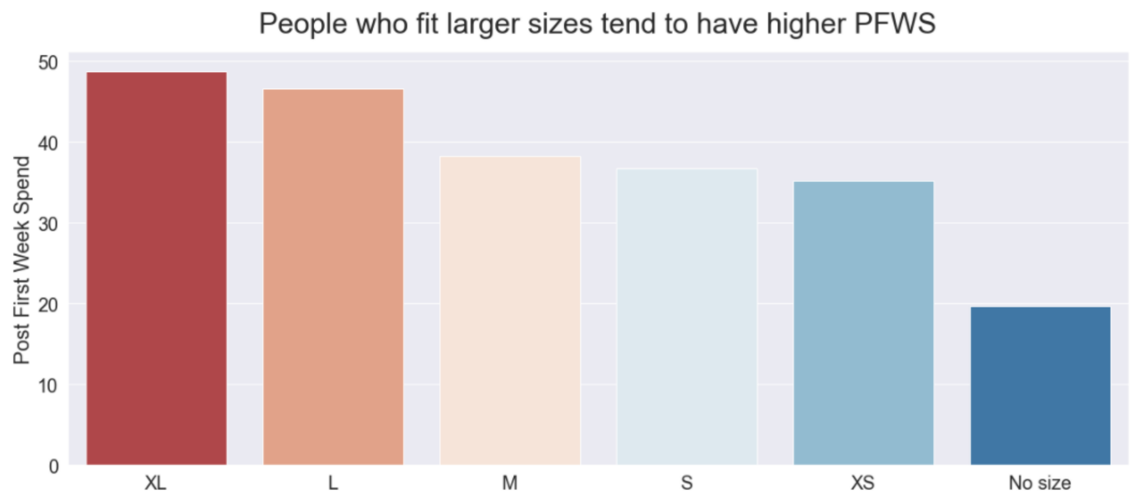
Post First Week Spend vs First Week Spend



According to our heatmap, our best predictive feature of *post first week spend* was the amount a customer spent in their first week on the platform. Using seaborn's regplot, we can observe a positive relationship between these two variables. Looking at the distribution, most customers are spending less than $50. At $50 of first week spend, we are seeing a higher number of customers making repeat purchases). There doesn't seem to be that many customers making repeat purchases between >$50.00-$79.00 spent but that between $80.00-100.00, this number increases again. This could be indicative of some popular products or product combinations which total to that sum.

One could have made the hypothesis that, if a customer has already spent a good amount on a product, particularly fitness apparel clothing, it is not likely they would have the need to spend again since they' already have bought a lot and won't have the need for additional ones. However, we can also formulate another hypothesis were customers who spent more were in the market for a good amount of fitness apparel and likely wanted to try out some of the items before they decided to purchase more if they were satisfied with their purchase. The data seems supports the latter hypothesis. However, as we discussed earlier, it is also possible that some resellers could be inflating these numbers, as they would simultaneously be spending more in both the first week and the 358 days after that. Thus, we want to carefully track our first week spenders, as it might be indicative, they will be more likely to make repeat purchases.

Post First Week Spend vs Size:

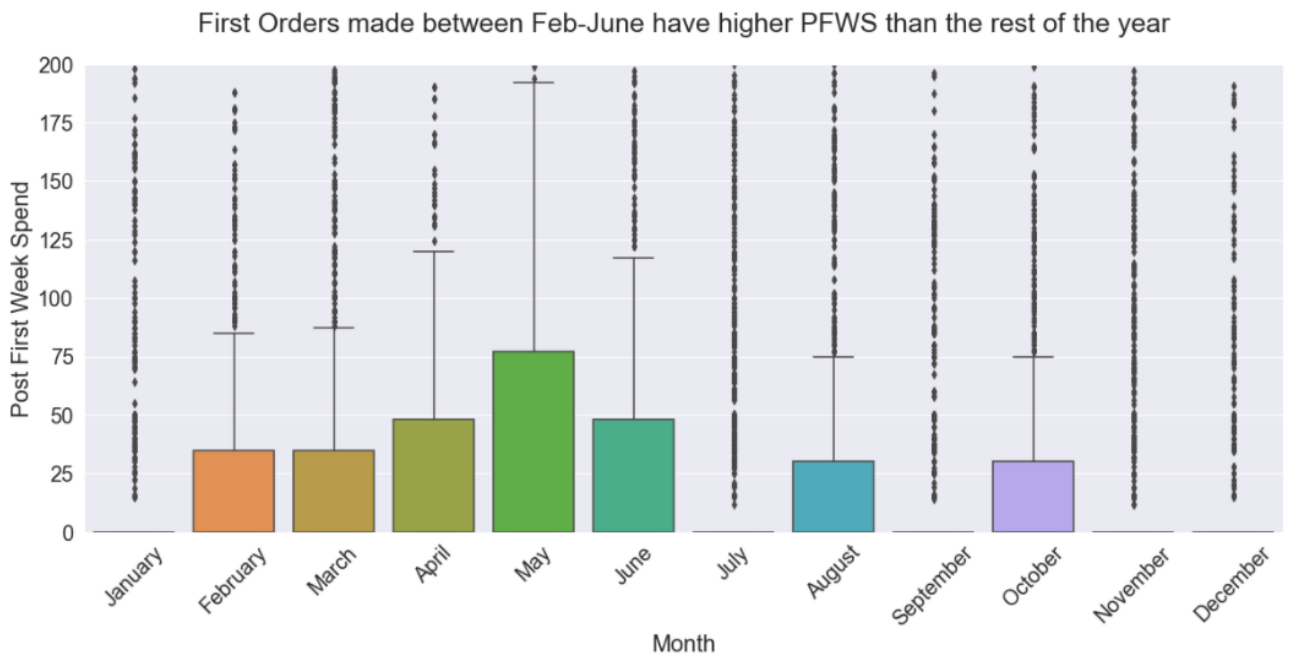


This is perhaps one of the most interesting findings which we cannot be discerned from the heatmap. As we go down in the sizing chart, our average post first week spends progressively gets worst from a high of $48 for XL to a low of $38 for XS. We do not count No size since these are just accessory purchases and are expected to be cheaper than clothing. There are two hypotheses which can explain this:

- *Poor Product Fit*: Our product fit is wrong, and sizing needs to be revisited. Customers who are buying smaller sizes are not as satisfied with the product, and therefore may not be purchasing be coming back to us as much.
- *Overly Optimistic Customers*: Customers do not know their size or "exaggerate" in thinking they are smaller sizes. Perhaps putting in place better sizing charts and dimensions before check out may change.

First Item Size therefore seems to play a role in post first week spend and could make for a good predictor variable. For modelling purposes, we will create a size variable that will be binary. The large size variable will be 1 if customer purchased an item that was XL or L or small if they purchased an article from another size, distinguishing the large sizes from the rest of the other product offerings.

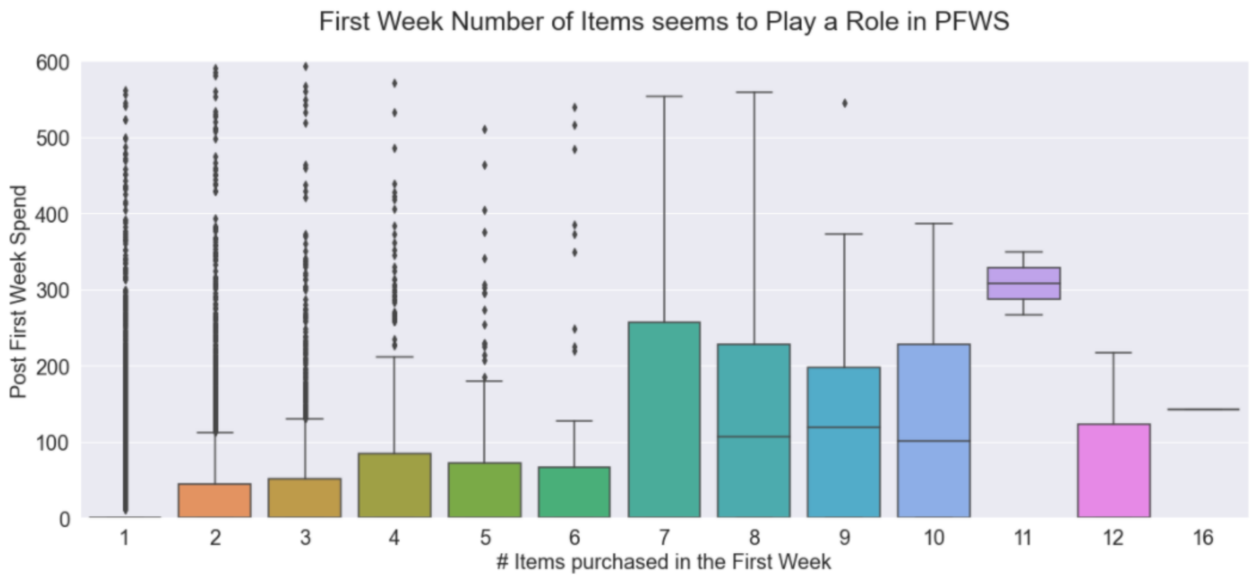
Post First Week spend vs Month of First Purchase



Post First Week spend is highest for months in the *February-June* . However, like most companies, this company does have most sales during the holiday periods. May has the highest post first week spend. This may be due to the nature of the most popular product which are principally leggings. As we saw, the median repurchase rate is *42* days, so a possible hypothesis is that they repurchase one in the summer months of June / July. Nonetheless, there does appear to be a trend.

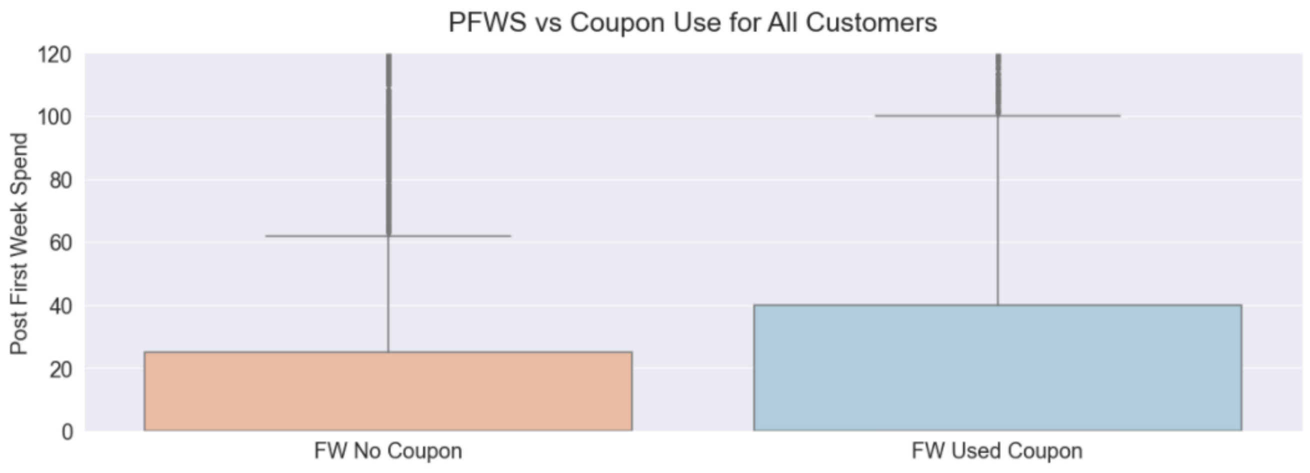
For the purposes of the modelling section, rather than have a column for each month, we will group Feb-June months in one category as “*hot months*” and all others in another.

Post First Week spend vs First Week Number of Items Purchased



It seems that the more a customer orders in the first week, the more likely they are to spend more in the following 51 weeks. This is a healthy sign for the business and seems to be a good indicator to incorporate into our model, however since we saw that it highly correlated with pfw_spend and that seemed to be a better predictor, we will have to skip this variable.

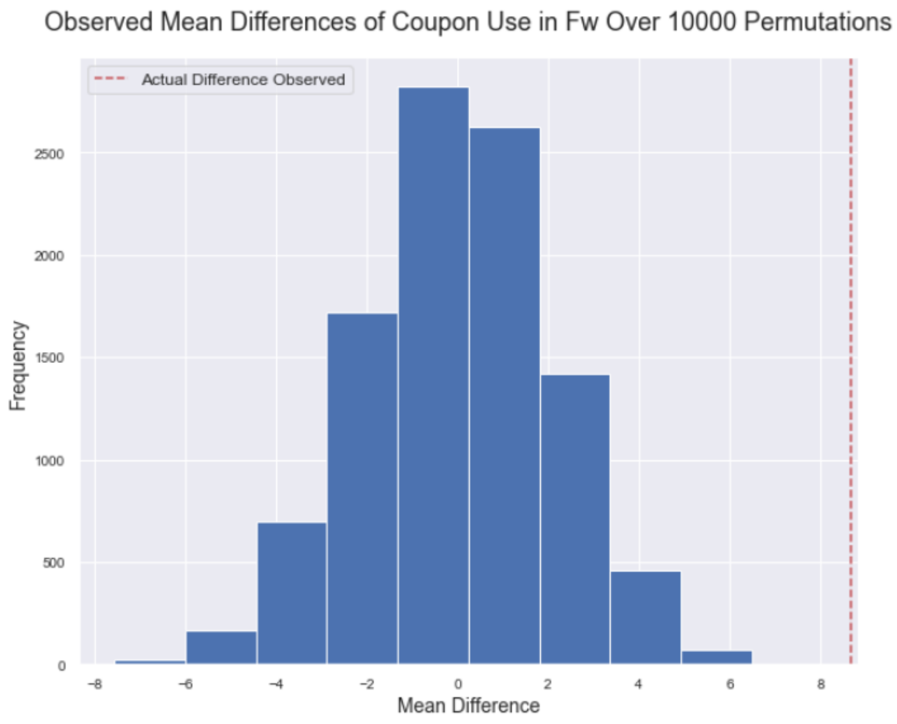
Post First Week spend vs Coupon Use in the First Week



It appears that customers who used a coupon in the first week spent more in the following 51 weeks than the customers who did not use a coupon. We observed a difference in means of **$8.69**.

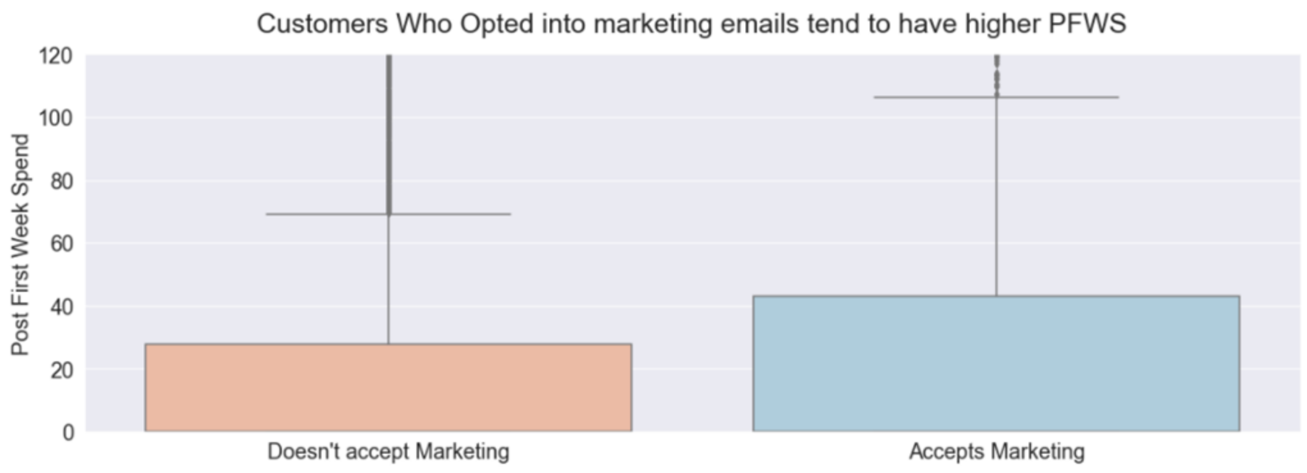
To determine if there was any statistical significance, we conducted hypothesis testing where our null hypothesis was that coupon use does not play a role in post first week spend and our alternative hypothesis was that it did play a role in increasing post first week spend. Our t-test confirmed this with a t statistic of 4.2 and a p-value of 0.003.

We also took 10000 different permutations and saw if our observed difference could be found in any of them, which was not the case.



Therefore, we reject the null hypothesis that coupon use does not play a role in post first week spend in order of the alternative hypothesis that it does play a role in increasing post first week spend. One possible explanation behind this is that if a coupon is used, a customer may feel like they are "getting a good deal" and are more likely to become a repeat customer, which increases the average post first week spend as a result. It could also be indicative of a customer being more engaged with the brand.

Post First Week spend vs Opting into Marketing Emails



We observe that customers who opted into marketing emails seem to have a higher spend. We proceeded to follow the same methodology as above with the coupons. This time our null hypothesis is that opting into email marketing does not play a role in pfw spend and the alternative is that it does. The observed mean difference was **$10.69**. The results from the t-test show a t statistic of **4.11** and a p-value of **0.03**, which means that at the 95% confidence interval, we can reject the null hypothesis in favor of the alternative hypothesis that marketing emails seem to play a role in. As a business, we want to try and incentivize people to opt in.

Other

There were additional variables which we observed in the notebook. We looked if the four different *regions* played a role, but they all had virtually identical post first week spent means, indicating that this was not a good variable to consider. We also looked if the customer was in one of the *100 wealthiest zip codes* in the United States. We hypothesized that if they were in wealthier zip codes, we could target more of our advertising to those customers, but the findings were the opposite: Customers outside the wealthiest zip codes would spend more.

4. Pre-Processing & Modelling :

Based on our findings in the EDA section, we proceed to prepare the data for modelling with some feature engineering. We start by dropping certain columns: *fw_nb_items* will be dropped as the heatmap indicated it highly correlated with *fw_total_spend*, which seemed to be the better predictor, *fw_nb_orders* is very unbalanced, *region* was found to be insignificant and *fw_purchased_accessory* is logically redundant with first item size.

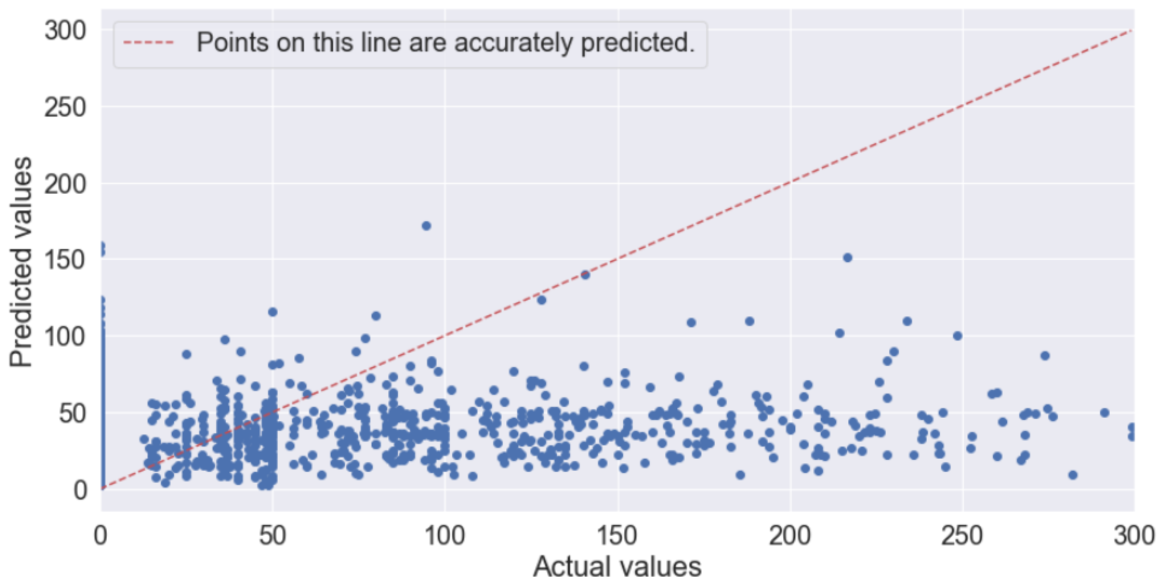
We then proceed to modify *first_order_month* to be 2 categories: February-June vs rest of the year. We do the same for *first_item_size*, splitting them in 2 categories: Large Size and Smaller size. We then use one-hot encoding to create dummy variables for our categorical variables. Lastly, we scale our data using StandardScaler, since *fw_spent* will contain larger values in magnitude and this could disrupt the mode.

Using train-test-split, we train our data *75%* for training and *25%* for testing. Our training data contains **7942** data points, and our testing data contains **2648**. Our data is now ready to be modelled.

Part I. Regression Models : Predicting Post First Week LTV:

Our first Model was the *Multiple Linear Regression*. Using GridSearchCV, we search for the best parameters for this model and fit our model with these parameters. The result is an R squared of **0.046** and **0.036** on the training and testing data respectively which is very poor, since 0 indicates no relationship. Let's see how our predictions compare with our actual results.

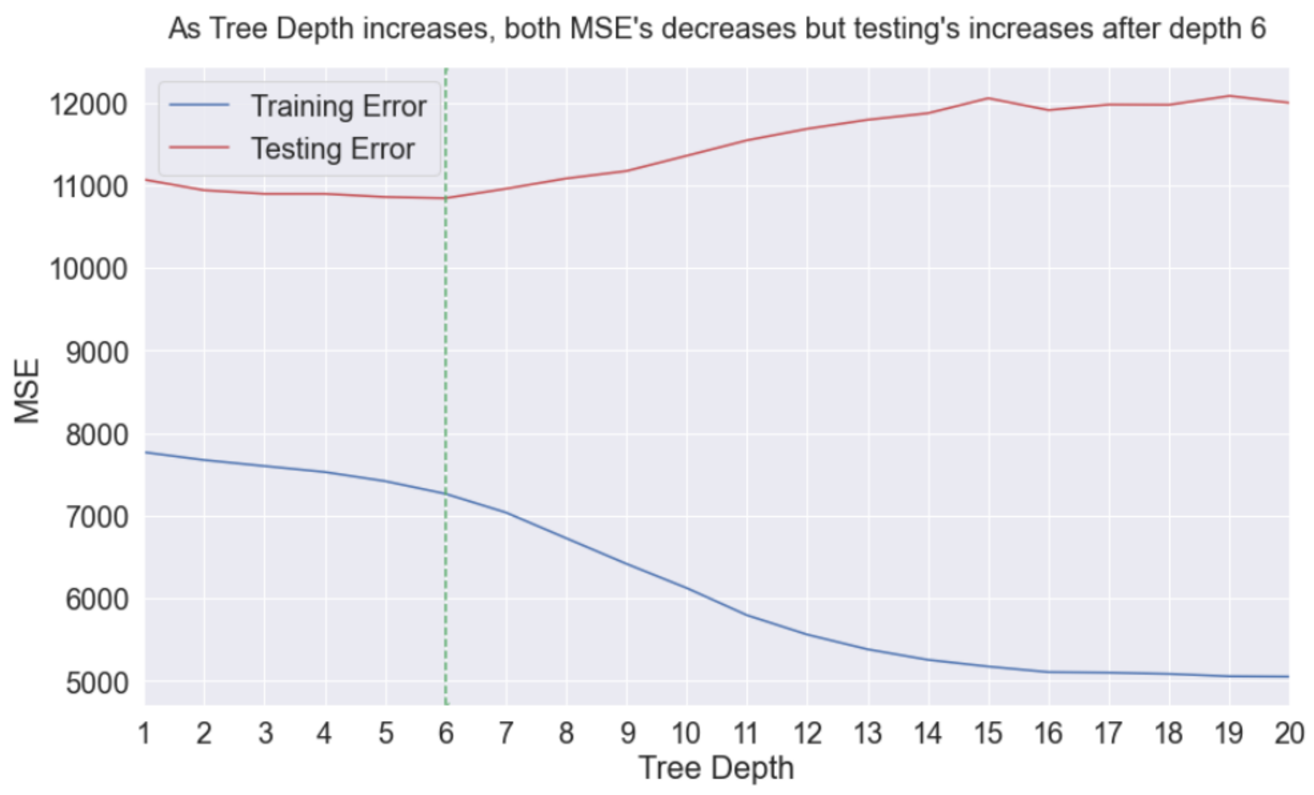
Linear Regression Model Results:



This is a terrible result! Clearly the relationship is not best explained with a linear model.

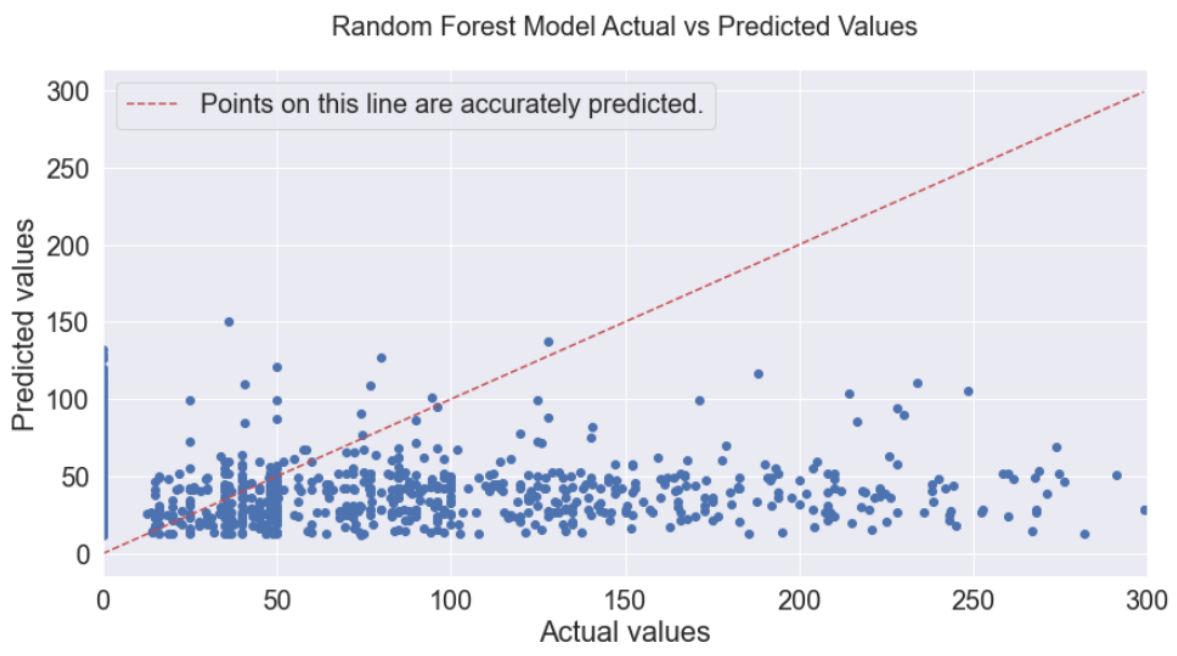
Given the poor findings of the linear model, we decided to turn to the *Random Forest Regressor* to see if a tree-based model would generate better predictions. Once again, we fit the model with the training data and tuned the hyperparameters using GridSearchCV. One of the most important parameters in the Random Forest Regressor is the depth of the tree. To avoid overfitting and tailoring the model to close to our training data, we want to find the optimal depth for the testing data.

Mean Squared Error for Training and Testing Data vs Tree Depth



As we can see, the optimal tree depth in this case is 6. Looking at this result, the Mean Squared Error the model generates decreases up to that depth on the testing data but increases after. Unfortunately, it seems the mean squared error is very elevated. Our model results give us an R squared of 0.03 for the testing data and a RMSE of 104.335, which is worse than the linear regression.

This was the predictions the model generated which was not much better than our previous model or than random.



Interpreting our Regression Metrics, Shortcoming and Transition to Classification:

Human behavior is after all difficult to predict and how much a customer. At the end of the day, the best our models were able to do was to show that the Shopify data was able to predict about a little more than **3%** of the variance in LTV for customers. As a comparison, this analysis was replicated for 2 different shopify datasets, and the range of R squared was between **3%-5%** for each store. If we had a new customer come in and had to guess how much there total LTV would be after the first week, we are probably better off guessing the mean post first week spend. There are additional degrees of freedom which can explain how much a customer will spend and we do not have access to the data which could explain these better predictors. Examples of variables which could have helped and been better predictors include, but are not limited to:

- Follows influencer / CEO on IG
- Where are they in their fitness journey?
- Days a week they work out?
- Height / Weight
- Race / Ethnicity
- Source of ad they were exposed too (Tik-Tok, Instagram)
- Were they Referred ?

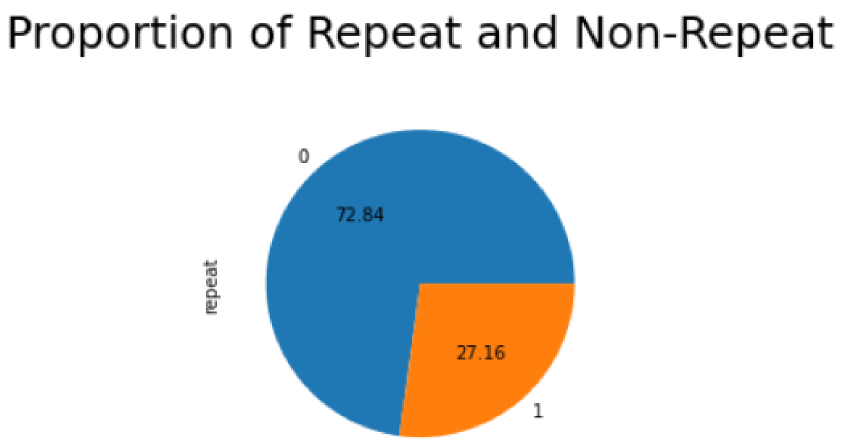
- How would they describe their diet?
- Level of Interest in working out.
- Mood on the Day of Purchase ? (not trackable!)
- Product pricing to the competition
- Satisfaction of the Purchase
- Etc.

There are other factors which explain these poor results. Unfortunately, our company was very limited in resources and was not able to follow up with customers with marketing emails to increase repeat purchases. Lastly, when we trained our data, we were training it with 73% of customers having not made a second purchase. The regression and random forest models therefore underpredicted the actual values as a result. We did correct for this by testing for only the repeat customers, but by doing so, our results marginally improved, and we discarded a lot of data. As a result of this reason, we decided to see if we could predict whether or not a customer was going to purchase again after the first week.

Part II. Classification : Predicting Repeat Customers:

Feature Engineering:

The Classification portion required reformulating our question "can we predict the post first week spend?" to "can we predict if a customer will purchase again past the first week? ". We create a new target column 'repeat' which will have a value of 1 if our customer's *pfw_spend* is greater than zero (indicating they made a purchase after the first week LTV period), and we drop our original *pfw_spend* target column. Our target variable becomes *repeat* with the positive class 1 denoted for repeat customers and 0 for non-repeat.

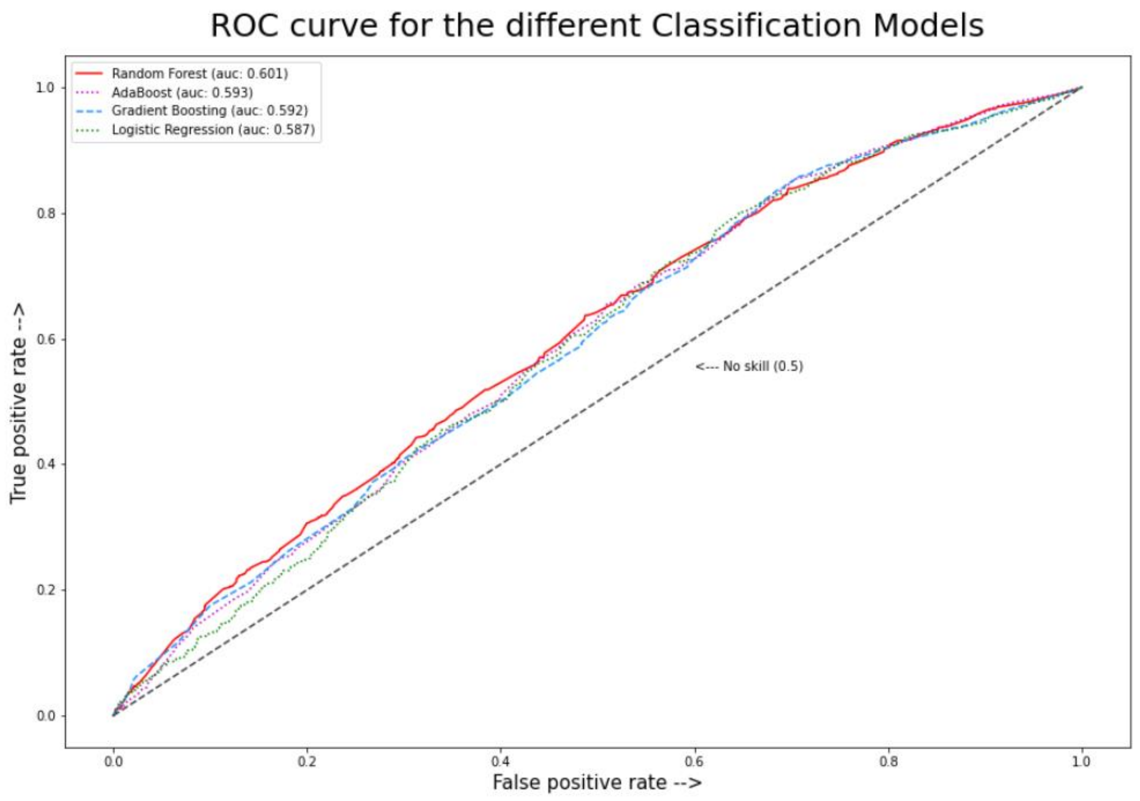


There is the presence of class imbalance in our dataset. Several methods exist to address class imbalance for classification problems, we opted to use the resampling technique, specifically using under sampling. This technique will keep all records of the repeat customers (the minority class) but will only sample a portion of the non-repeats (the majority class) so that we have an equal 50/50 split amongst the two classes to train our model on. We want our model to learn how to distinguish between the repeats and non-repeats. In the absence of this technique, a machine would favor the majority class and potentially ignore the minority class altogether, which is what happened when we didn't use this technique oversampling as it classified all the customers as being non-repeat. If we don't resample and give us an *accuracy score* (the number of correctly predicted repeat and non-repeat customers over all predictions) that mostly favors the majority will not be the best metric but now it will become relevant again since we are training the machine equally, it should be able to better distinguish between repeats and non-repeats. What we will be most interested in observing will be the model's *precision* (the percentage of all the correctly classified predicted repeat customers) and *recall* (the percentage of the actual repeat customers who were correctly classified).

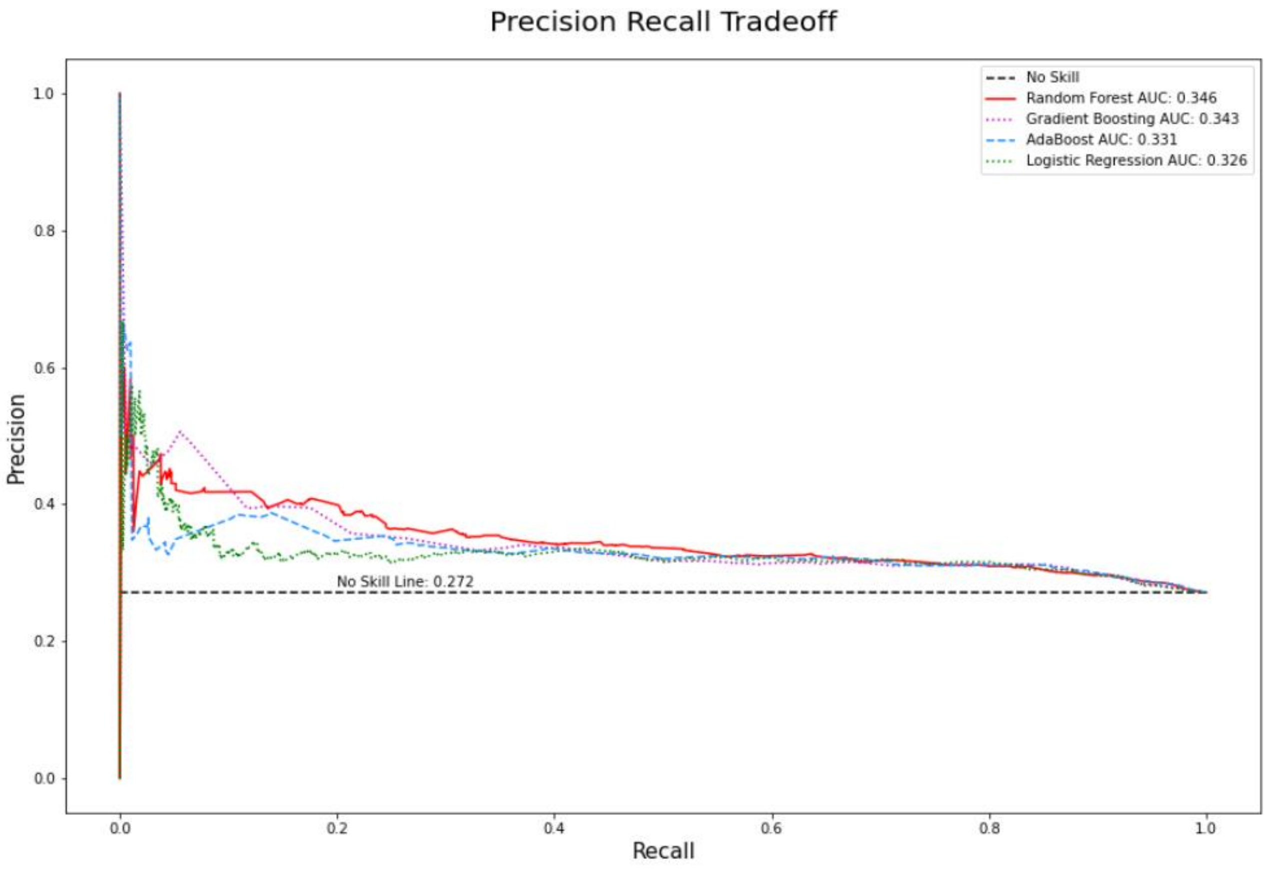
Model Selection and Feature Engineering:

Four models were compared: Logistic Regression Classifier, Random Forest Classifier, Gradient Boosting Classifier and AdaBoost. To the exception of Logistic Regression, where we used GridSearchCV, we proceeded to use Randomized Search CV with 5-fold cross validation for the other three models, using ROC_AUC as our scoring metric. We performed hyperparameter tuning this way and trained our models with the best parameters.

Results:

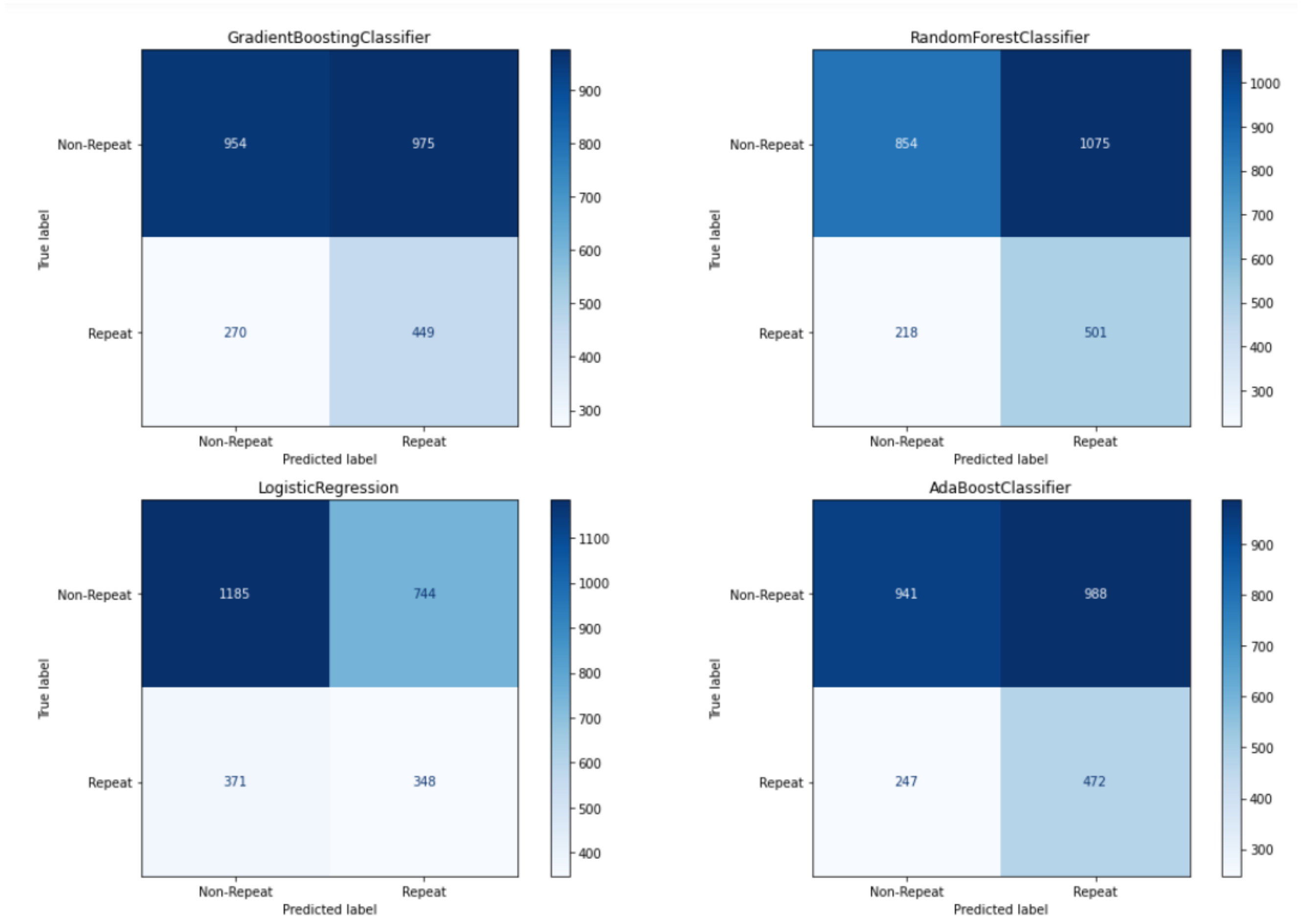


In a perfect scenario, we would want a curve that has a very high True Positive Rate for a very low False Positive Rate. At a very high level, this would indicate that the model is able to correctly identify the number of repeat customers while minimizing the number of incorrectly classified ones. Depending on the threshold of incorrectly classified repeat customers we are willing to accept, we can get a certain percentage of correctly classified ones, and vice versa. In these results, it seems that the Random Forest Classifier is our best model since it has the highest area under the curve. It is closely followed by AdaBoost, Gradient Boosting and Linear Regression. The results indicate the algorithms are performing better than random, but are far from being perfect classifiers, which was to be expected from what was discussed in the previous sections. We can also take a look at the Precision Recall curve.



In this scenario, if a classifier was powerful, it would be able to simultaneously be able to have high precision as well as high recall. The no skill line of **0.272** refers to the actual proportion of the minority class. We compare our models to that line. As can be seen here again, Random Forest has the highest AUC at **0.346**, this time closely followed by Gradient Boosting at **0.343**.

Confusion Matrices of Our Different Models:



Interpreting our Model Results:

There are many ways to interpret our results. To better understand them, we need to think of how these models would perform relative to a dummy classifier that were to guess at random. We will convert all results to percentages for simplicity:

Actual

Repeat	0.27
Non Repeat	0.73

No-Skill Classifier -- Model Trained on 50/50 data and predicts randomly with 0.5 probability with test data 0.27 / 0.73

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.365	0.365
	Repeat	0.135	0.135

Perfect Model - All customers correctly classified, precision, recall and accuracy are all 100%.

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.73	0
	Repeat	0	0.27

Logistic Regression Model - Best Model to distinguish the *Non-repeat Customers*

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.45	0.28
	Repeat	0.14	0.13

AdaBoost Model Results - Best Model to distinguish the *Repeat Customers*.

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.36	0.37
	Repeat	0.09	0.18

Gradient Boosting Model

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.36	0.37
	Repeat	0.10	0.17

Random Forest Model

True Label		Predicted Label: Non-Repeat	Predicted Label: Repeat
	Non -Repeat	0.32	0.41
	Repeat	0.08	0.19

For our data, the dummy classifier will classify **50%** of the non-repeats as repeat and non-repeat, and vice versa for the repeats. This means that when classifying at random, the default dummy classifier has the following metrics:

- *Accuracy* : **0.5** → (0.365 + 0.135)
- *Precision* : **0.27** → (0.135 / (0.135 + 0.365)) ; Rate of the positive class
- *Recall* : **0.5** → (0.135 / (0.135 + 0.135))
- *F1-Score (weighted)* : **0.35** → ((0.5 * 0.27) + (0.27 * 0.5)) / 0.77

So which model is better ?

Model	Best Hyperparameters (scoring = ROC_AUC)	ROC_AUC	F1_Score (weighted)	Precision	Recall	Accuracy
Logistic Regression	{'C': 0.001, 'l1_ratio': 0, 'penalty': 'l2'}	**0.587**	**0.384**	0.319	**0.484**	**0.579**
Random Forest Classifier	{'n_estimators': 10, 'max_features': 'sqrt', 'max_depth': 4, ' criterion': 'gini', 'bootstrap': 'False'}	**0.601**	**0.437**	0.318	**0.697**	**0.512**
AdaBoost Classifier	{'n_estimators': 250, 'learning_rate': 0.01, 'min_samples_leaf': 10, be_max_depth': 2}	0.593	0.433	**0.323**	0.656	0.534
Gradient Boosting Classifier	{'n_estimators': 5, 'max_depth': 3, 'loss': 'deviance', 'criterion': 'mae'}	0.592	0.419	**0.315**	0.624	0.530
No Skill Classifier	**NA**	**0.50**	**0.35**	**0.27**	**0.5**	**0.5**

Ultimately, it depends on the end goal we are trying to achieve. If the objective is to maximize the *F1-Score,* then the Random Forest model with **0.437** is the best. It represents a **24%** increase from the Dummy Classifier. Additionally, it has the best *recall* score at **0.7**, so out of all the repeat customers, it is the one that can classify most of them correctly.

The Adaboost Model nonetheless shows us something interesting. It outputs the same results as the dummy classifier for non-repeat customers. but does a good job at discerning our repeat customers, with a *recall* of **0.66**, representing an increase of **40%** as opposed to the dummy.

While the Random Forest does have a higher number of correctly classified repeat customers, it does so at the expense of incorrectly classifying more non repeats as being repeats. This is reflected with AdaBoost's *precision* score of **0.323** compared to the Random Forest's with **0.318**. The *precision* score for the AdaBoost model was the highest and represents an improvement of **20%** from a random guess. Therefore, if we care about making better predictions, AdaBoost is our best model to pick.

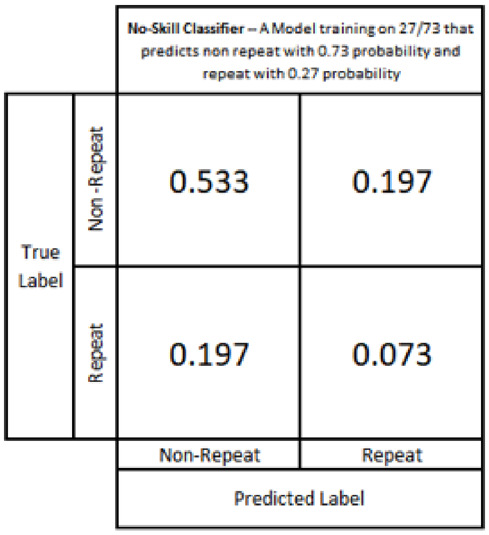
Interestingly, the opposite is the case for the Logistic Regression: It predicts repeat customers poorly (having close to the same predictions as the dummy) but is much better at discerning the non-repeat customers. If we were looking to understand what makes a customer likely to not repeat, then in that case, the Logistic Regression would be the best model to use, and we can use it to send coupons to all of the customers it identified as not being repeat after a week to try and get them to purchase again.

If we want to place a certain weight on precision and recall, we can use the f-beta scoring, so depending on the weights importance, our model can also change. The conclusion here is that there is not one model that there is not one dominant model that dominates.

Thresholding for Decision-Making.

As an alternative to resampling, we can also perform thresholding. In this scenario, we did not resample our data but changed our threshold for classification. By default, the threshold to be classified in the positive class means that an algorithm needs to return a probability higher than 0.5. However, since our data is trained on 73% / 27%, it puts higher weights in classifying into the negative class. To get out optimal *F1-Score* using this method, we created some code which outputs the best scire for each of the different thresholds of classification between 0 and 0.5. We found that Gradient Boosting had the best weighted *F1-Score* at **0.45** with a threshold of **0.27**. This model had a *recall* of **0.69**, a *precision* of **0.33**

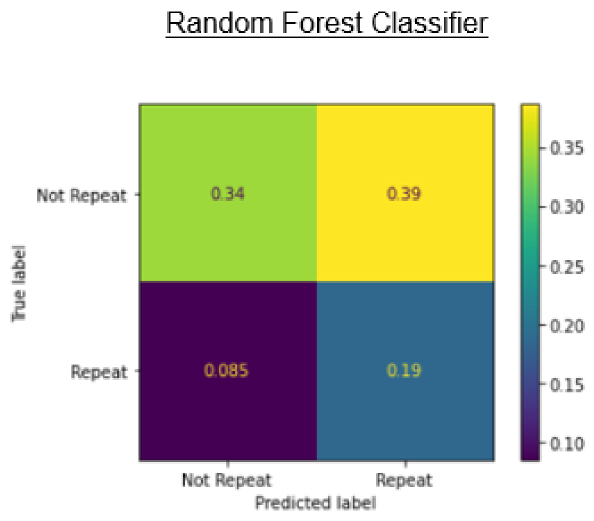
In this instance, since the data has not been split 50/50, we compare the result to a different No-Skill Classifier showed below:



No-Skill Classifier – A Model training on 27/73 that predicts non repeat with 0.73 probability and repeat with 0.27 probability

True Label		Non-Repeat	Repeat
	Non -Repeat	0.533	0.197
	Repeat	0.197	0.073

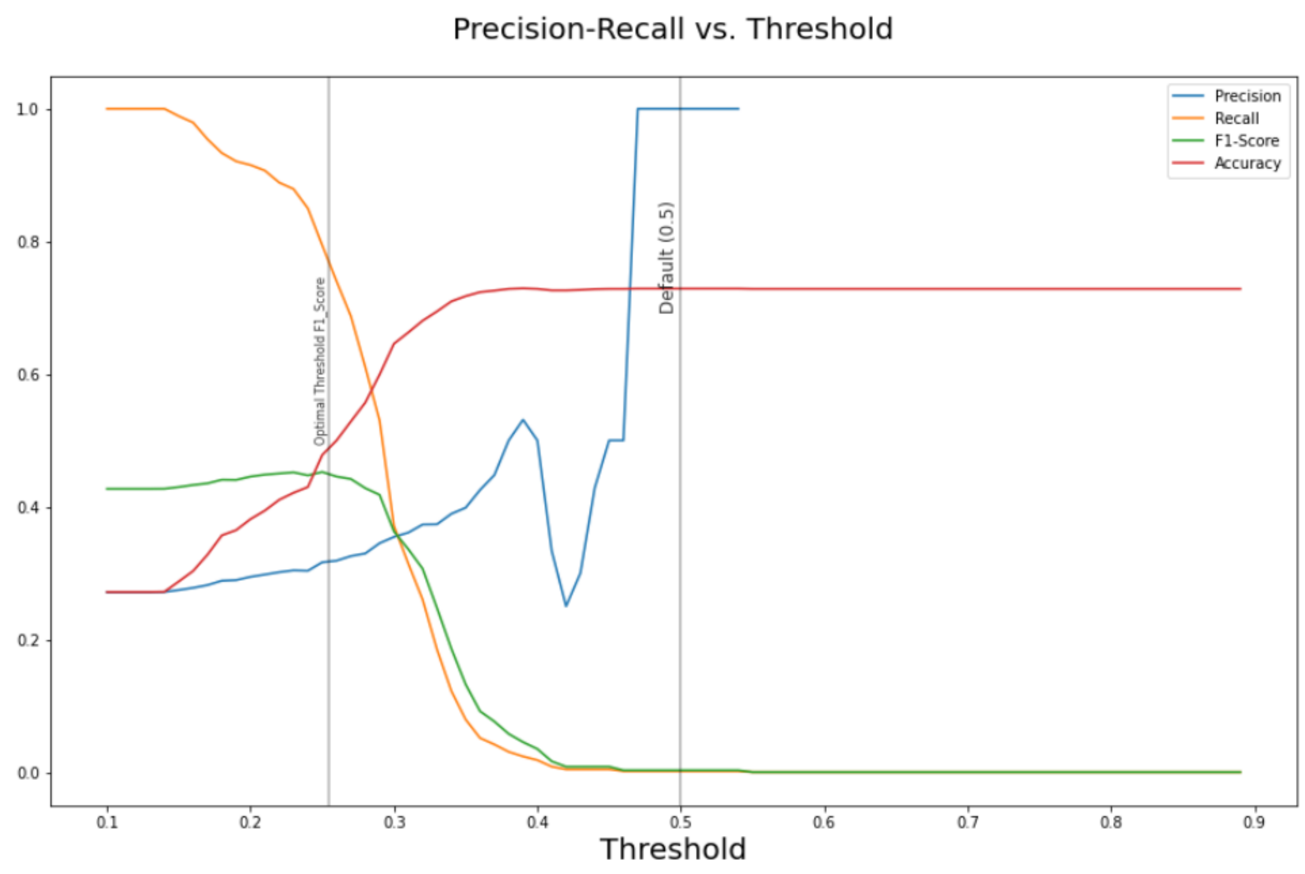
Predicted Label



F1-Score: **0.27** ; Recall : **0.27,** Precision: **0.27**.

When optimizing for F1-Score, we get an improvement of *66%*, *155%* and *22%* respectively for these metrics as opposed to a no skill classifier.

Lastly, with thresholding, we can use this graph to pick a precision recall tradeoff.



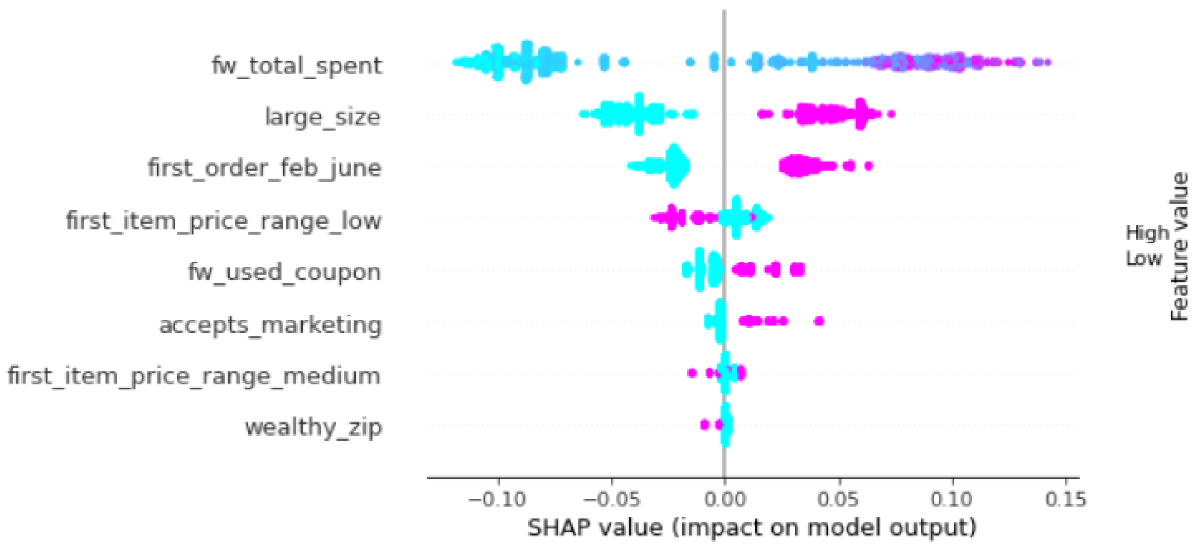
As we can see, recall experiences a significant decline from threshold 0.23 – 0.32 plummets and precision does not increase significantly in that interval. If the was able to make better predictions, we would see a much more progressive decline for recall and steeper increase for precision, which is not the case. An important factor to note is that while we picked a threshold of 0.27 to maximize the F-score, it looks like that with different data, we may get different results as it could be overfitting to this data. Therefore, the optimal range of the threshold might 0.25-0.28 depending on the new data coming in.

When using thresholding we separately tried ti find the *best precision score* and got 100% but this is because it only classified one customer as repeat that was repeat. As a result, it was not useful. Unfortunately, none of the models were able to get us very high precision but were able to have decent recall, thus when using these models, you will get close to identifying which ones of your customers will repeat.

5. Business Recommendations and Conclusions:

Exploring Variable influences on the Model with SHAP

We can look at how the model . The SHAP package in Python allows for this. Let's take an example of what the Gradient Boosting Model considered to make its decision :



Recommendations:

As a result, from this analysis, we recommended the following changes to the company:

- **Revisit Product Sizing**. As seen, sizing needs to be revisited as smaller sizes are generating smaller post first week spend, thus affecting LTV.
- **Optimizing Marketing Campaigns**: Make better use of our marketing and re-marketing campaigns to get customers to repeat more.
- **Coupon Optimization**: Optimizing Coupon Distribution and using the Logistic Regression Model to target the customers who end up in the True Negatives as they were likely not to repeat again and could benefit from discounts to persuade them to shop again.
- **Increase first week spend:** Find ways to get customers to spend more in the first week.

Considering these recommendations as well as well as closely monitoring first week spend we would be able to forecast the change in the customer repeat rate and potentially decrease the cost to acquire these new customers.

Future Work:

In the future, additional work could be used to get better results and to deliver additional and deliver better insights:

- Integrating Google Analytics to track more information about the purchases.
- Getting more data and thinking of additional ways to integrate it with our CLTV table.
- Finding the right customers for promotions
- Adding a column for each item and test which individual products generated higher post first week spend. This company was not a good example, since this works better on a larger scale and with more product diversity (80% of products were leggings).
- Using unsupervised machine learning techniques to cluster our customers.

Thank you very much for taking the time to read this report! We If you would have any feedback or have any questions or would me to help conduct these kinds of analyses for your business, please feel free to reach out to me at bbellman95@gmail.com. Best of luck in using getting the most of Shopify’s data for your business!