Real-time prediction for customers' purchasing tendencies based on pageviews

Team: SharkyData

Professor: Mason Porter

Buyan Li

Yun Lin

Yu-Yuan Chang

Yafei Dong

1 Executive Summary

1.1 Context

Fingerhut is an established e-commerce retailer that offers a diverse range of products, including electronics, necessaries, health care, and more. The company has been in operation for over 70 years and has built a reputation for providing affordable products and convenient credit options to its customers. As a senior student majoring in data theory, Fingerhut's website presents a unique opportunity for analyzing customer behavior and predicting their purchases based on pageviews. By collecting data on the frequency of pageviews for a particular product, data analysts can gain insights into customer preferences and forecast which products are likely to be purchased in the future. Through data analysis techniques such as machine learning algorithms, the relationship between pageviews and purchases can be analyzed to generate accurate predictions. This approach can provide valuable information to Fingerhut's marketing and sales teams, allowing them to optimize their inventory management, pricing strategies, and promotional campaigns. By leveraging the wealth of data available on the website, data analysts can gain valuable insights into consumer behavior and improve the performance of the company's sales and marketing efforts.

1.2 Problem

E-commerce companies are constantly seeking ways to improve their sales performance and increase their revenue. One critical aspect of this is predicting the likelihood that a customer will make a purchase during a shopping session. This information is highly valuable, as it enables companies to target their customers more effectively with promotions and marketing resources that are most likely to drive sales.

In this project, our goal is to prototype a predictive model for our industry partner Bluestem Brands that can accurately forecast the probability of a customer making a purchase during a shopping session. To accomplish this, we will analyze a range of customer behavior data, such as page views, search history, and previous purchase history. Using machine learning techniques, we will develop a model that can predict the likelihood of a purchase based on these factors. By deploying this model, Bluestem Brands will be able to more accurately target their customers with promotions and marketing resources that are most likely to drive sales. This will improve their cost and return ratio, ultimately leading to increased profits. By providing a specific solution to this problem, we aim to deliver tangible value to our industry partner and contribute to the ongoing advancement of the e-commerce industry.

1.3 Methodology

Our methodology for this project involves a comprehensive approach to ensure that we deliver accurate and reliable results. Initially, we will conduct extensive research and gather relevant insights from stakeholders to establish a clear understanding of the business needs. This will include a preliminary review of the available data to gain insights into its size and structure. Once we have defined the problem, we will proceed to explore the data in detail. This will involve conducting exploratory data analysis using descriptive statistics and visualizations to uncover patterns and relationships between variables. Based on the insights obtained, we will perform data cleaning and feature engineering to prepare the data for modeling. The next step is to develop machine learning models to predict the likelihood of a customer making a purchase. This will involve evaluating various modeling techniques, including decision trees, regression analysis, and ensemble methods. We will use techniques such as hyperparameter tuning and cross-validation to ensure that our models are robust and perform well on the available data. To evaluate the performance of our models, we will use appropriate metrics such as accuracy, precision, and recall. We also explore other metrics such as AUC-ROC curves and F1 scores to gain a more comprehensive understanding of our model's performance. All in all, we reflect on our methodology and approach to identify areas for improvement and opportunities for future research. Our methodology will adhere to best practices and established standards in data science to ensure the highest quality of results.

1.4 Results

The study concluded that the random forest model is the best approach for predicting the target variable. Cross-validation and parameter tuning was performed to determine the optimal values for the criterion and max_depth. The results indicated that criterion = “entropy” and max_depth = 20 produced the highest F-1 score and accuracy. Mean Decrease in Impurity (MDI) was used to identify the feature importance, which showed that "session_time" and "promo-code" were the most informative variables. This aligns with the initial feature assumption and supports the accuracy of the model.

2 Introduction

In the internet era, e-commerce has become an integral part of everyone's lives. Unlike traditional business models, e-commerce generates enormous amounts of data about its customers constantly. Over the past decade, it has been proven that how well companies can make use of that data is the key to gaining a competitive advantage. If implemented well, solutions based on data science can help companies make improvements in every aspect of the business from cutting costs to increasing sales. A great example of this is using machine learning models to predict customers' purchasing tendencies. Through implementing this kind of system, companies can gain insights into how likely a customer is going to make a purchase and allocate their marketing resources accordingly.

We partnered with Bluestem Brands to build data sciences projects using data from Fingerhut, a Bluestem subsidiary. Fingerhut is an online retailer that has provided us with their pageview/clickstream data for a week.

For this project, our team will prototype a real-time purchase tendency prediction system based on a week of customer pageview data. Through our project, the ultimate goal is to prove the feasibility of building such a system on our industry partner's platform. More specifically, we want to train machine learning models to give predictions for the problem: **whether a customer would make a purchase at the end of a visit to the company's online store based on the pages viewed so far in this visit.**

To answer this question, we've divided the task into a few parts:

1. Using graphs and summary statistics to better understand the purchasing tendencies of the customers.

2. Test various machine learning models and find the one best suited for our

application.

3. Explore the limitations of our method and future challenges.

3 Methodology

3.1 Assumptions

3.1.1 General Assumptions

Before starting to build the model, we first need to clarify the set of assumptions under which the models are expected to operate. These assumptions are needed because ultimately this project is based on mathematical abstractions of the real world. The entire complexity of the real world is impossible to be represented in simple mathematical models, thus calling for making assumptions in order to allow us to abstract real-world problems into researchable mathematical problems.

First and foremost, it needs to be understood that the data we are able to explore were very limited and it is possible that there are other key variables that affect the problem “if the user will make a purchase”. If these variables are changed, the model will not perform as well and can even be useless. Acknowledging this issue, we assume that the result our models are trying to predict only depends on the data we are able to collect plus randomness. In other words, information that’s not in our dataset is assumed to have no impact on what we’re trying to predict. Thus, this underlying relationship between the label and the features can then be abstracted as $y = g(X) + \epsilon$, where y is the label we’re trying to predict, X are the features that decides y, and ϵ represents randomness.

Secondly, implementing machine learning models to make predictions real-time at scale has many infrastructure challenges such as computing power requirements. Depending on the availability of resources for these challenges, different models might be more suitable in production. Due to the scope of this course, we'll just focus on prototyping a model that works best for the amount of data we have on hand and assume that all the models will perform the same in production when the scale is much larger.

Last but not least, there are many non-technical concerns when it comes to implementing a new technology in a business. For instance, there might be legal and compliance concerns when a business implements this kind of system. Regulators and lawmakers in certain regions might have different rules for collecting user data and using it to improve sales. This can cause changes to both the data that our model is trained on and the construction of the model itself. Similarly, there can even be ethical concerns for implementing this system. Although we've carefully considered the ethics before starting this project, it is by no means comprehensive and there could always be new ethical concerns coming up as it is being used on more and more people. While always keeping ethics and the law in mind, we make the following assumption for this project: non-technical problems when implementing this system in real life are not considered.

Thus, we assume for each unique visit of a session, if the visitor spends time on the same page without further action than 10 minutes, we would consider the visit to be 'offline' then we dropped these pages.

3.1.2 Assumptions Due To the Dataset Given

Due to time and hardware constraints, we were only given one dataset containing clickstream data for a week and we were forced to make some assumptions due to this unavailability of data. A key category of variables we were interested in exploring is variables related to the prices of products in the shopping cart. However, there is no such variable available in the dataset given to us. The closest ones `Cart_Interstitial` and `cartRemove` server calls which track the products added to and removed from the shopping cart. Thus, the best solution we could come up with is to assume every visit starts with an empty shopping cart and use the sum of products added and removed to approximate what's in the cart. This is clearly unrealistic but still nonetheless provides information of the same category.

3.2 Terminology

There are a few key terminologies that will be appearing throughout this report and we'll clarify the definitions in the following:

- *Session*: Given the context of this project, each unique `visitid` is defined as a session.
- *Visit*: In this report, *visit* is used interchangeably with *session*.
- *Pageview*: A pageview refers to a line of data generated from each server call. In this report, pageview and server call are used interchangeably.

3.3 Model Architecture

Since we are building this project with the goal of real-time prediction in mind, it is necessary to create an architecture that fits this purpose. For a model to make predictions based on the current page and all the previous pages, it has to be able to store past information in some form of memory within the model. Most machine learning models do not have this property and can only make predictions based on one data point, which presents a big challenge. After reviewing some relevant literature online (Seippel, 2018), we've come up with the following architecture for our model.

First, to include information about previous pageviews, we are going to create new features in our dataset. Based on previous works on similar topics, summary statistics (max, min, mean, sum) of the price of products viewed and in cart play an important role in predicting if there will be a purchase at the end of the visit. This works as a form of memory for our dataset since at each pageview, key information about previous pageviews is stored in these new features.

Then, various machine learning models are going to be trained on the dataset with the new features. Each datapoint for the models is a single pageview, and the models will be trained to make predictions on each datapoint. In other words, every time a user has a new pageview, the model will give a binary prediction of whether the user will make a purchase.

4 The data

We'll be using user visit session data from Dec 10 2022 to Dec 16 2022, with 42730149 rows in total. Due to the size of this dataset, we were unable to read all the columns into our coding environment. For this project, we loaded 17 columns, namely `'visitid','pageurl','pagename', 'pageeventvar2','pagetype', 'visitdatetime', 'hit_time_gmt', 'productlist', 'searchterms', 'searchresults', 'newvisit', 'post_evar27', 'evar28', 'evar83', ’promocode’, ’devicetype’,` and `'ordernumber'`. Each row represents a pageview log and each column represents the attributes collected during the pageview.

4.1 Limitations

Since we only have customer shopping (browsing fingerhut site) data from Dec 10 2022 to Dec 16 2022, there will be a shortage in terms of recognizing the customer behavior. The project is limited by the assumption we made earlier.

5 Preliminary explorations

Fig 1. Number of Visits vs Time by date

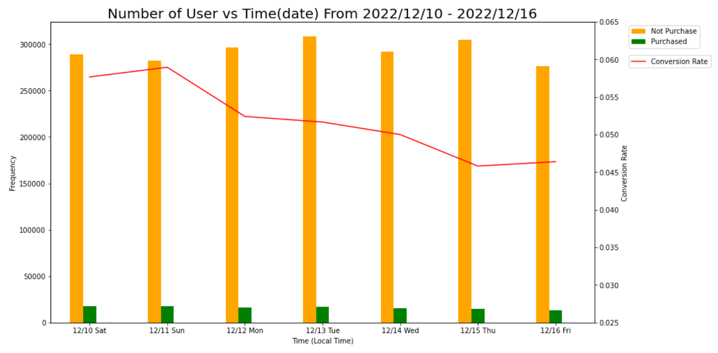
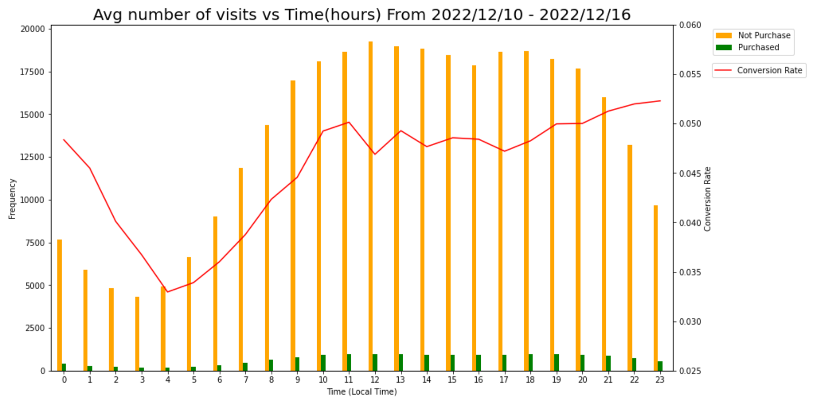


Fig.1 is showing the purchasing trend of fingerhut from Sat to Friday. Worth to mention that on Sunday had the highest conversion rate while it didn't have the most visit number.

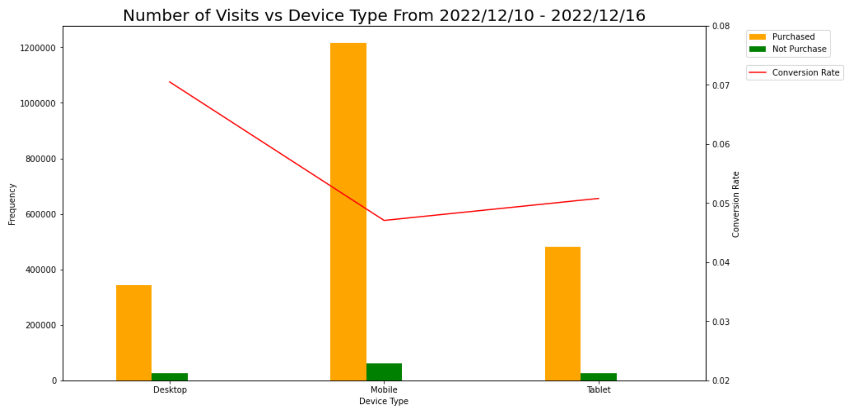
Fig 2. Avg Number of Visits vs Time by Hours



From fig.2 , we can see the average purchasing trend of fingerhut from 0 am to 23 pm. We noticed that after 10 pm to 1 am, it had the highest conversion rate while it only contained fewer visits.

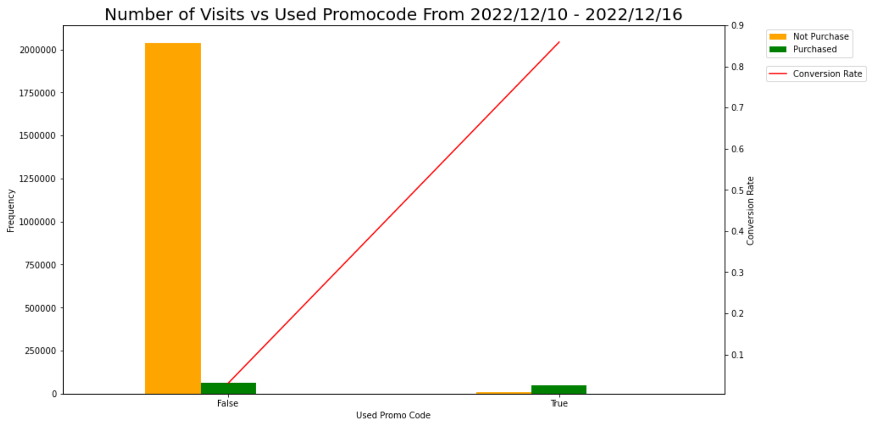
X-axis: 0 represents 00:00 - 00:59, 1 represents 01:00 - 01:59…,etc

Fig 3. Number of Visits vs Device Type



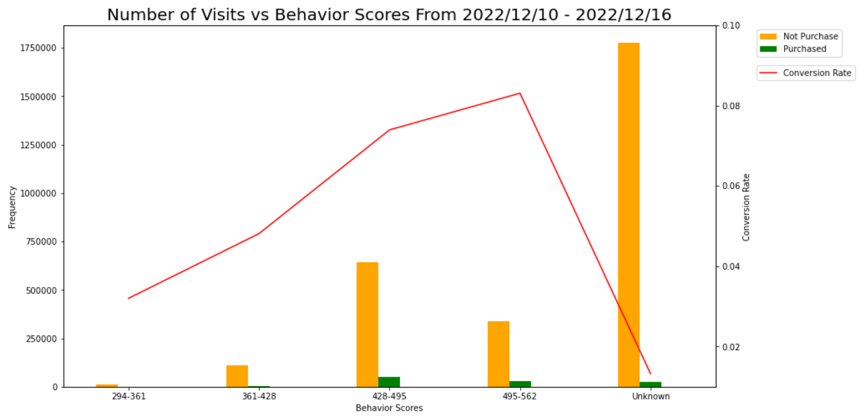
From fig. 3, we can see that although mobile is the most used device for users, desktop and tablet users have higher conversion rate.

Fig 4. Number of Visits vs Using Promo Code



From fig. 4, we can see that people who used promo code to purchase an item are lower than the people who did not use promo code. Also, the spike orange bar from not using promo codes are displaying people who just browse the site, which they did not purchase anything. Another fact from this plot is that a small number of people applied the promo code in their cart but didn't purchase the item at the end. Moreover, we were not sure about the promocode being applied before checkout or after checkout which can be a potential concern, so we have to assume it was applied before checkout in order to include it in our model.

Fig 5. Number of Visits vs Behavior Score (Equally Bin)



From Fig. 5, we equally bin the behavior score from 294 to 562 into 4 bin, and we can tell the rate from better score customers tend to have higher conversion rate. Also, the majority of the visits are unknown due to not providing enough information. which are likely just the user who is browsing the site, and they have the lowest conversion rate.

6 Preprocessing & Feature Engineering

6.1 Adding "Memory" Features

As mentioned in section 3.3, we're going to add features that contain information about previous pageviews within the same visit. Based on previous work done on this topic (Seippel 2018), the most useful such features are going to be summary statistics of prices of products in the cart and summary statistics of prices of products viewed (in the product details page). We're going to construct these features based on the price information given in `productlist`. Since the strings in `productlist` are the only places where price information is stored, we'll first have to parse the strings in `productlist` to create a dictionary object that maps the product code to its price.

Via this method, we have constructed a python dictionary called `prices` that contains 41684 product codes and their prices. With this information on hand, we can track the prices of products the user has viewed and in the shopping cart.

6.1.1 Prices of products in shopping cart

Although there isn't a variable that tracks what products a user has in the cart at each pageview, we can still manually construct such a variable using `pagename` and `pageeventvar2` from the original dataset plus the `prices` dictionary that we've just created. Based on observations, every time a product is added to the cart, a pageview with `pagename` equals `Cart_Interstitial` and `productlist` equals the product code is generated. Conversely, every time a product is removed from the cart, a pageview with `pageeventvar2` equals `cartRemove` and `productlist` equals the product code is generated. Being able to track the products added to and removed from the cart, we are able to create columns `min_price_cart`, `max_price_cart`, `avg_price_cart`, `sum_price_cart` (Variable documentation given below).

6.1.2 Prices of products viewed

Similar to the prices of products in the shopping cart, we want to track the prices of products viewed by the customer. We did this by extracting the price of the product viewed from productlist at every server call with pagename starting with pdp. Aggregating these prices gave us sum_prod, avg_prod, min_prod, max_prod.

6.2 Behavior Score

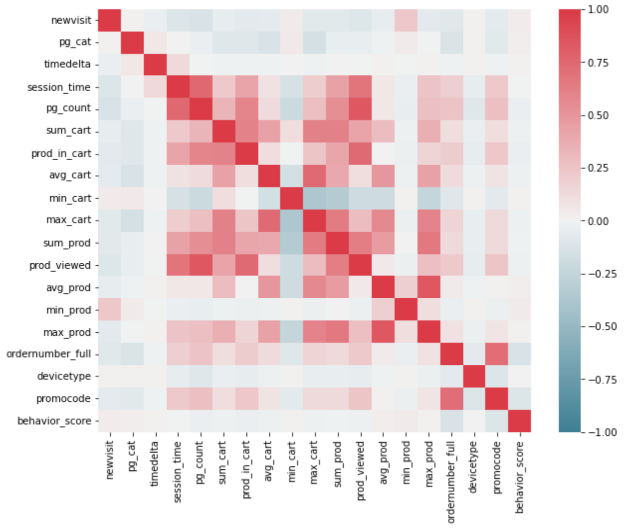
From section five Fig.5, we can see the big difference from conversion in different behavior score groups. Therefore, we decided to bin the evar_83 equally into four groups and unknown, which we created the behavior_score variable with behavior_score_294-361, behavior_score_361-428, behavior_score_428-495, behavior_score_495-562, and behavior_score_Unknown five categories.

6.3 Time Delta

The timedelta (in second) were calculated by the difference between each session. Based on the assumption we made, for the visitor to spend time on the same page without further action than 10 minutes, we would consider the visit to be 'offline' then we dropped these pages. Thus, we will remove all the rows in timedelta that are over 600 seconds.

6.4 Feature Selection

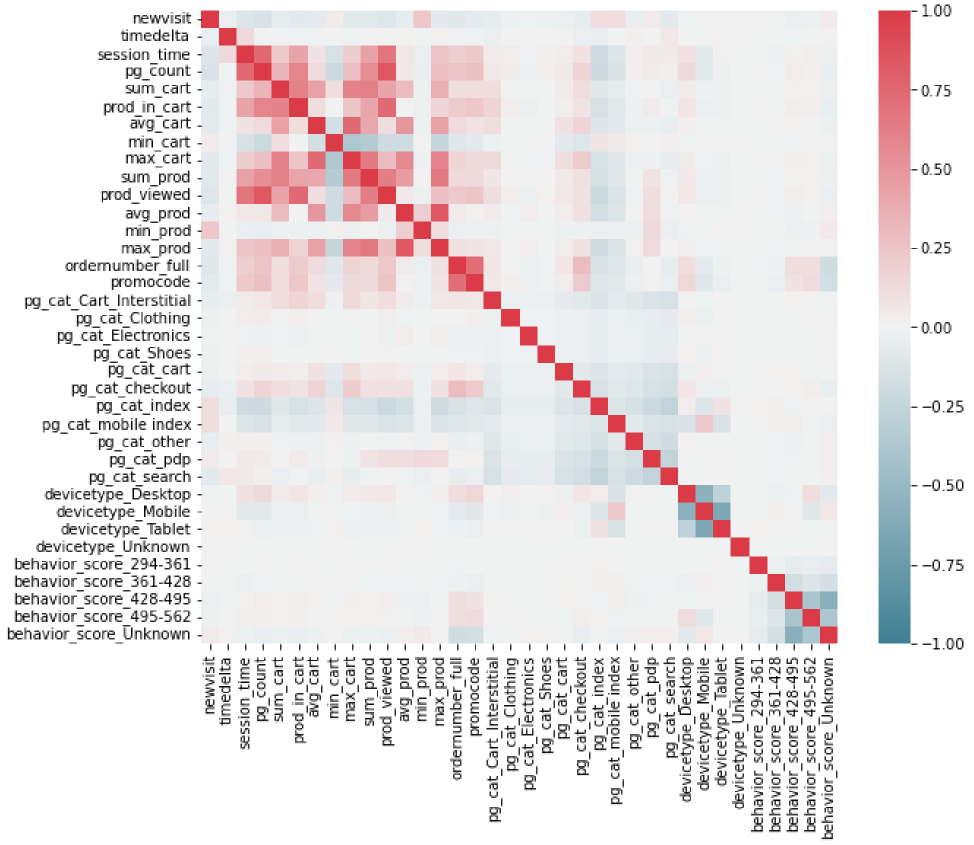
There are 256 features in the original dataset and only a few are going to be useful when building our model. By reviewing related literature, we've decided to keep the following features: 'visitid', 'newvisit', 'pg_cat', 'timedelta', 'session_time', 'pg_count', 'sum_cart', 'prod_in_cart', 'avg_cart', 'min_cart', 'max_cart', 'sum_prod', 'prod_viewed', 'avg_prod', 'min_prod', 'max_prod', 'ordernumber_full', 'devicetype', 'promocode', 'evar83', 'behavior_score'. These features will be subjected to more selection steps as we're building the models. The figure below shows the correlation of selected features, for this plot we preprocess encoding 0, 1, 2, … for `pg_cat`, `behavior_score` and `devicetype` to check the correlation for these two features with other features. We can observe that there is no high collinearity issue within the features we selected, and the feature that may have potential issues is just the promo code.



6.5 Feature Engineering

6.5.1 Categorical Variables

Since 'pg_cat' and 'devicetype' have more than 2 categories within them, we assume the relationship between these categories are not linear. Therefore, we applied one-hot-encoding on 'pg_cat', and 'devicetype', which produced the following feature: 'pg_cat_Cart_Interstitial', 'pg_cat_Clothing', 'pg_cat_Electronics', 'pg_cat_Shoes', 'pg_cat_cart', 'pg_cat_checkout', 'pg_cat_index', 'pg_cat_mobile index', 'pg_cat_other', 'pg_cat_pdp', 'pg_cat_search', 'devicetype_Desktop', 'devicetype_Mobile', 'devicetype_Tablet', 'devicetype_Unknown', 'behavior_score_294-361', 'behavior_score_361-428', 'behavior_score_428-495', 'behavior_score_495-562', and 'behavior_score_Unknown'. The fig below shows the correlation matrix of selected features with one-hot encoding features. It does not show any high multicollinearity issue, which the features we selected and generated looks good for the model training.



7 Model Building

7.1 List of Features

Feature Name	Type	Description
`newvisit`	Boolean	If this visit is the first visit made by the user
`pg_cat`	Categorical	Category of the current page
`timedelta`	Float	Time between this server call and the previous one
`session_time`	Float	Total time spent since the start of this visit
`pg_count`	Integer	Total pageviews (server calls) generated since the start of this visit
`sum_cart`	Float	Total value of all the products in the shopping cart
`prod_in_cart`	Integer	Number of products of all the products in the shopping cart
`avg_cart`	Float	Average value of all the products in the shopping cart
`min_cart`	Float	Minimum value of all the products in the shopping cart
`max_cart`	Float	Maximum value of all the products in the shopping cart
`sum_prod`	Float	Total value of all the products viewed
`prod_viewed`	Integer	Number of all the products viewed
`avg_prod`	Float	Average value of all the products viewed
`min_prod`	Float	Minimum value of of all the products viewed
`max_prod`	Float	Maximum value of all the products viewed
`devicetype`	Categorical	Device type of this visit used
`promocode`	Boolean	If this visit applied promo code or not
`Behavior_score`	Categorical	The score is binned by ‘evar83’

7.2 Training/Testing Data and Cross-Validation:

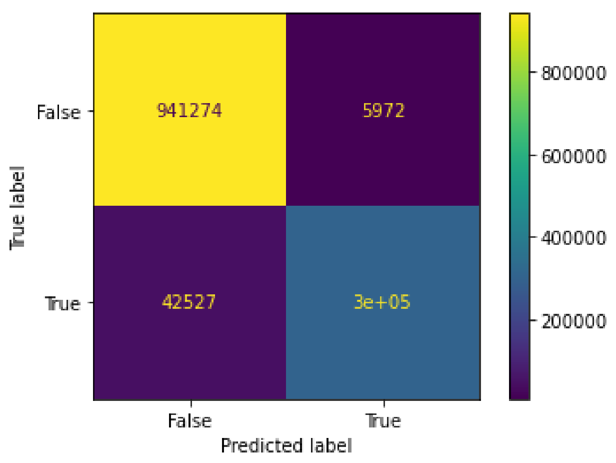
After feature selection and engineering, we had a complete data set with desired features. Due to the complexity of SVM and KNN, we have to sample out the dataset for those two models. We tried PCA to reduce the dimension of the data set, but after testing with different dimensions. We figured out that keeping all the columns but randomly sampling out 50% of rows is actually the best way to reduce computational time but keep information as much as possible. Also, we have utilized cross validation and parameter tuning to avoid overfitting and optimize parameter selection.

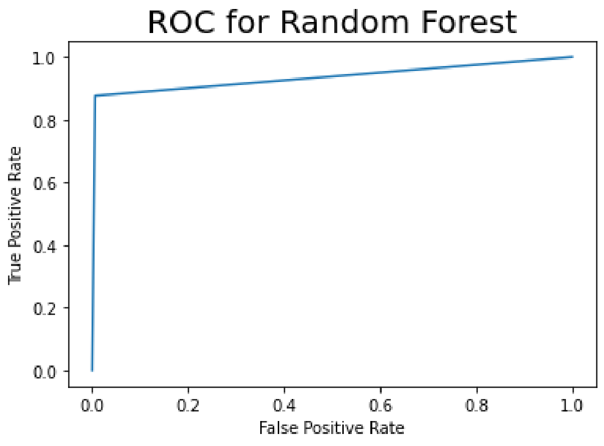
8 Result

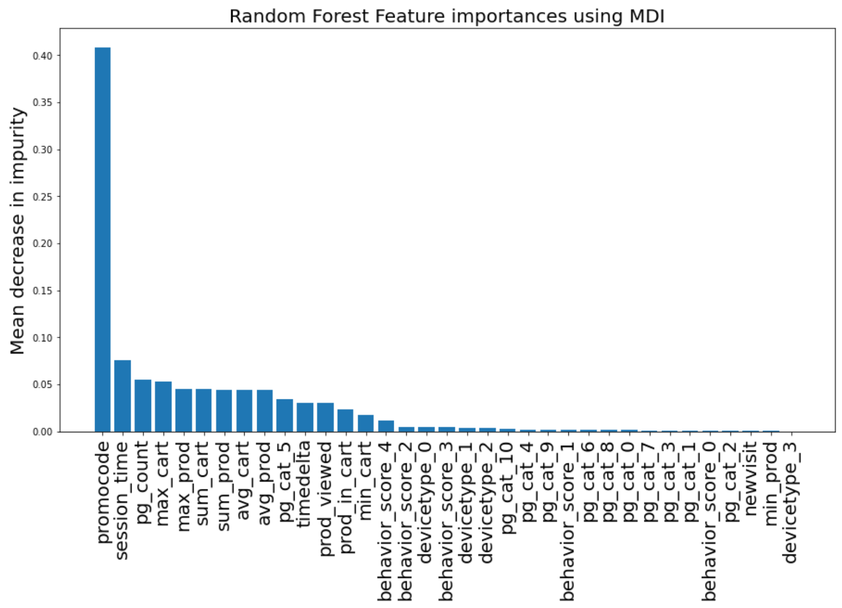
8.1.1 Random Forest

We use cross validation and parameter tuning to figure out which criterion and depth of the random forest is best for this model. For criterion, we tried "entropy" and "gini". For max_depth, we tried 10, 20 and 30. Based on the result, we determine criterion = "entropy" and max_depth = 20 will produce the best result for our selected features. We have the result:

Accuracy	0.9624
Precision	0.9806
Recall	0.8765
F1-Score	0.9256



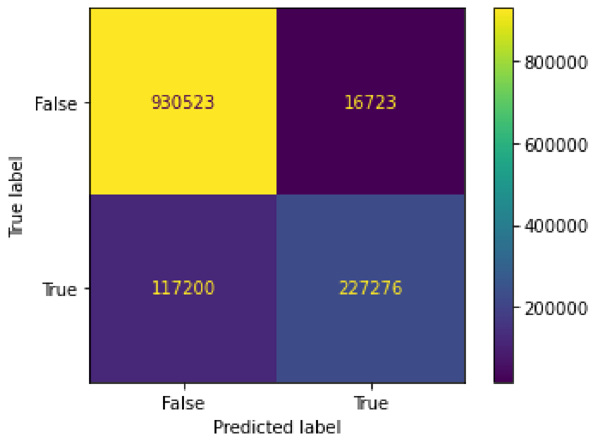


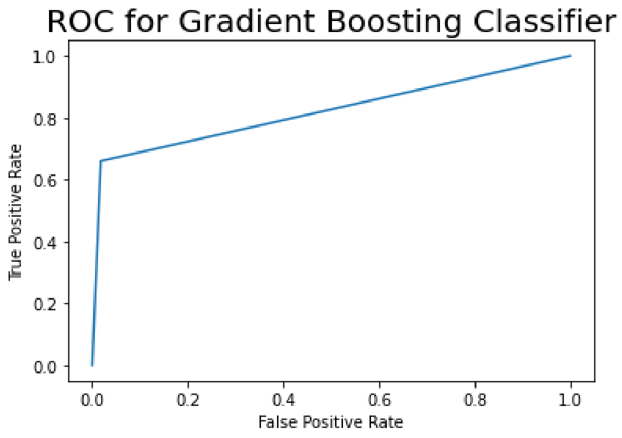


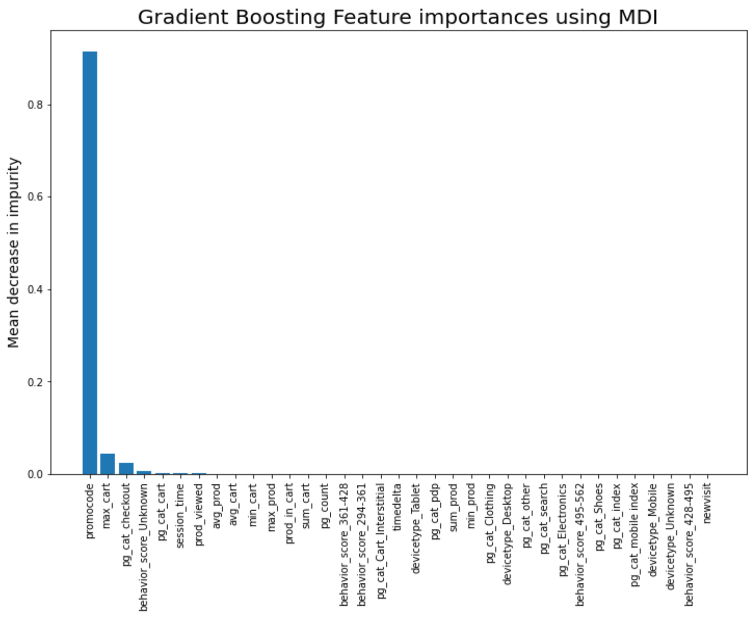
As we can see from the result, F-1 score and accuracy is high enough. Also, we list feature importance based on Mean Decrease in Impurity (MDI). The blue bars are the feature importances of the forest, along with their inter-trees variability represented by the error bars. As expected, the plot suggests that “session_time” and "promocode’ are significantly informative, which is reasonable for our feature assumption.

8.1.2 Gradient Boosting Classifier

Accuracy	0.8963
Precision	0.9314
Recall	0.6597
F1-Score	0.7724



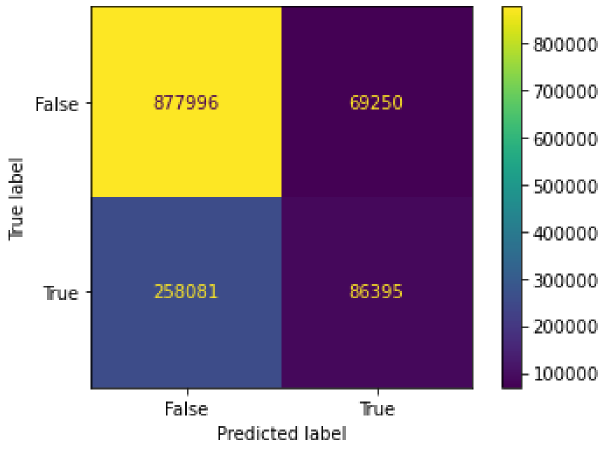


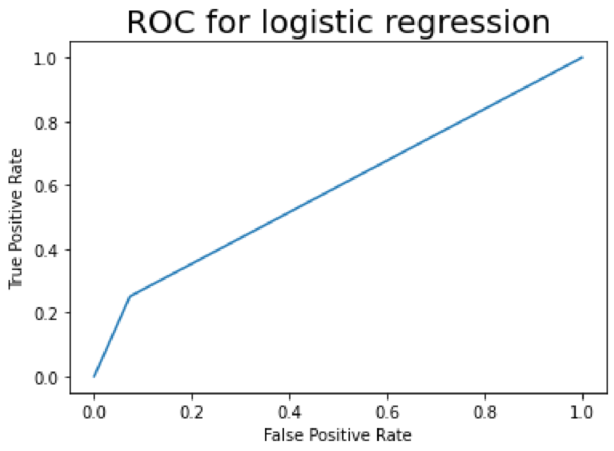


As we can see from the result, F-1 score and accuracy from the gradient boost classifier also perform well. The reason why the gradient boosting algorithm works well is because higher weight is given to the minority class at each successive iteration, which is ideal for imbalanced datasets. However, from the feature importance of the gradient boosting model, we observe the model is highly reliant on one feature ‘promocode’, which does have some limitations due to the timeframe.

8.1.3 Logistic Regression

Accuracy	0.7465
Precision	0.5550
Recall	0.2508
F1-Score	0.3454



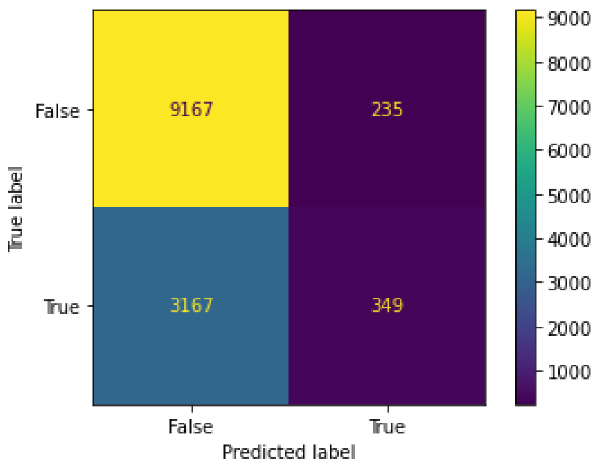


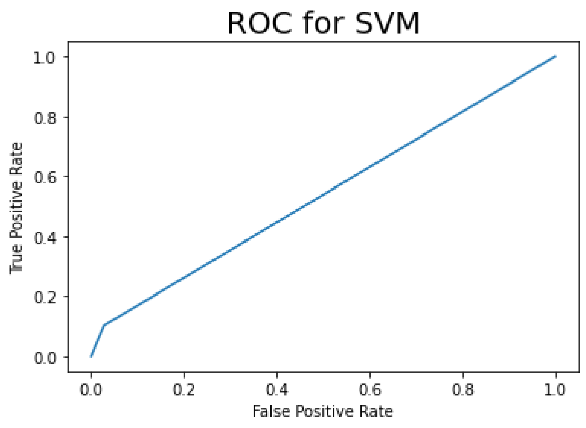
Based on the result, the performance is not desired. Logistic Regression is not appropriate for this particular data set. We suspect that some classes are highly correlated or highly nonlinear, the coefficients of your logistic regression will not correctly predict the gain/loss from each individual feature. In such cases, logistic regression cannot predict targets with good accuracy (even on the training data).

8.1.4 SVM

SVM has computation complexity O(NR) where R is the number of iterations, which causes the computational time to be extremely slow. We use a sample data set to run SVM to reduce computation time to a reasonable range. We tried “linear”, “poly” and “rbf” kernels. Even selecting the best kernel “linear”, the performance still did not meet our expectations.

Accuracy	0.7366
Precision	0.5976
Recall	0.0992
F1-Score	0.1702



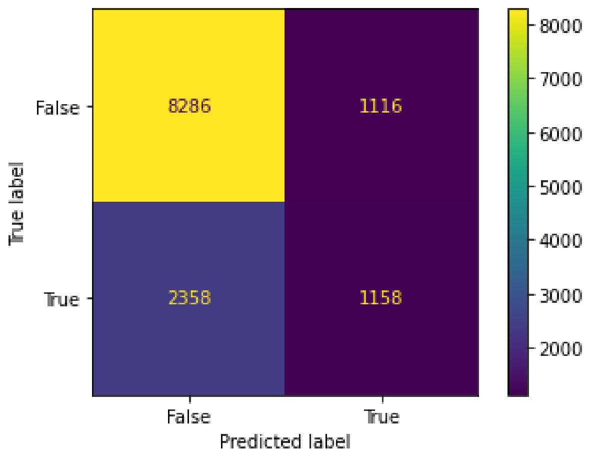


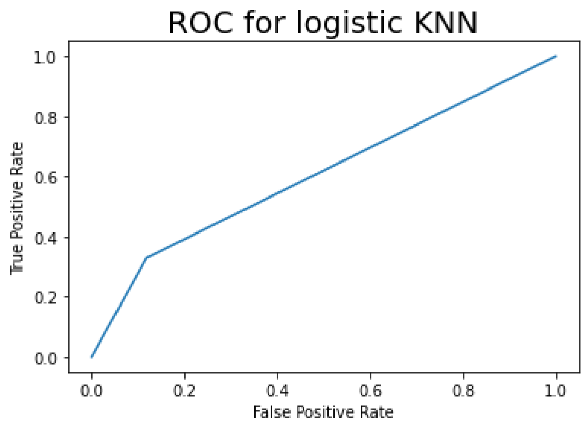
Based on the result, we can see that performing is not as good as random forest and Gradient Boosting. We checked the overfitting issue but it seems like SVM just does not perform well on our model. We can interpret too much information from this model.

8.1.5 KNN

KNN also has an extremely long computation time and is computationally expensive when the dataset has many features. We used sample data to train KNN models. We use parameter tuning on weights and n_neghbor parameters. We find out with default value n = 7 and weights = "uniform" performance the best compared with other parameter combinations.

Accuracy	0.7310
Precision	0.5092
Recall	0.3293
F1-Score	0.4





Knn did not meet our exception on F1-score, we used cross-validation to prevent over-fitting problem but the performance still has not increased. KNN is not helpful for our model, the computation time and performance were not good enough to represent the model we need.

8.2 Conclusion

Based on the result of random forest and Gradient Boosting with given data. We can tell by the feature importance graph that promo code and session_time have affected purchasing the most, which is reasonable under our model assumption. Beside promo code, our team would suggest fingerhut tries to keep customers in the webpage as long as possible. For example, a better recommendation/UI system that keeps customers browsing the page longer.

Also, our team understands that distributing promo code will cause decrease in revenue, but based on the result, customers prefer to make purchases with promo code. Our team would suggest fingerhut to distribute coupons as much as possible without greatly affecting profits

9 Future work

Analyzing the frequency of pageviews can provide insights into customer preferences and predict future purchases. However, the limited data available only scratches the surface of the website's potential. Firstly, to further develop the prediction model, a larger time frame dataset needs to be collected. This will enable more robust analysis of customer behavior and allow for the development of personalized recommendation systems for individual customers. Moreover, factors external to the website, such as economic conditions, weather, and social trends, can significantly impact consumer behavior. The data we were not sure was all done by user interaction,so we would suggest a better bot detection system to eliminate the noise data. Incorporating these external factors into the prediction model can further improve its accuracy and provide a more comprehensive understanding of customer behavior.

10 Citations

Customer purchase prediction through machine learning, 2018, Hannah Sophia Seippel.

https://essay.utwente.nl/74808/1/seippel_MA_eemcs.pdf Accessed Feb. 2023

30 Ecommerce Conversion Rate Optimization Steps to Help Boost Sales,

https://www.bigcommerce.com/articles/ecommerce/conversion-rate-optimization/ Accessed Feb.

2023