On Tailoring Wine & Spirit Recommendations Through Browsing & Shopping History

Thomas Marmo
College of Computer, Mathematical, & Natural Sciences
University of Maryland, College Park
College Park, MD, United States
tmarmo@umd.edu

***Abstract*—A national wine & spirits retailer can apply machine learning techniques to leverage years of customer data for the purpose of enhancing product recommendations. Online alcohol retail is growing every year as a percentage of total alcohol spending, and customers are naturally less exploratory without exposure to salespeople. Mimicking an in-store experience online through product recommendations that are highly personalized, highly regarded, and highly profitable can lead to a more satisfactory customer experience while simultaneously improving gross margin rates. The study delves into the integration of online browsing history and a combination of in-store and online shopping history to build a predictive model that can serve as the foundation of a product recommendation system that significantly improves key performance indicators such as add-to-cart rate, conversion rate, average transaction value (ATV), and units per transaction (UPT).**

Through the utilization of Spark, Machine learning models can be deployed and scaled up to accurately predict if a given user will purchase a particular wine or spirit. Browsing behavior can identify characteristics of products, such as departments or tasting notes to develop a customer's palette. After an initial testing cycle, additional filters can be applied to the model to make weighted adjustments. The retailer employs tactics to prioritize products belonging to the "Winery Direct" and "Spirits Direct" product lines. The two categories enjoy significantly higher profit margins, owing to the retailer circumventing traditional distributors.

Keywords*— *Spark, conversion rate, margin rate, wine, spirits, Winery Direct, Spirits Direct

I. Introduction

In the competitive landscape of the wine and spirits retail, providing personalized and relevant product recommendations to customers has become a critical aspect of maintaining customer engagement and increasing sales. After the newfound reliance of alcohol e-commerce that boomed during the COVID-19 pandemic, it has become even more vital for all wine & spirits retailers to optimize their online shopping experience. In addition to a traditional A/B testing strategy that tests front-end user experience (UX) changes on the website, a back-end algorithm adjustment can address the challenge of improving the product recommendation strategy through the application of machine learning techniques to leverage customer data.

The study focuses on harnessing three primary data sources: online browsing history, in-store shopping history, and online shopping history. Through these rich datasets, the aim is to create a robust machine learning model capable of predicting whether a customer is likely to purchase a product or not. With enough browsing and shopping history, a coherent story can be told that highlights a customer's interests and preferences within the two distinct categories of wine and spirits. Knowledge on a customer's willingness to purchase a given product can feed into a recommendation strategy that recommends products that a customer is likely to enjoy based off engagement from similar customers, and ideally increasing the following key performance indicators:

- Add-to-cart rate (ATC)
- Cart-to-checkout rate (C2C)
- Checkout complete rate (CCR)
- Conversion rate (CVR)
- Average transaction value (ATV)
- Units per transaction (UPT)

Generating product recommendations that cater to customer preferences on an individual basis can potentially create an entirely new problem. A user can be alienated into a niche pocket by solely relying on personalized recommendation systems. While recommendation systems can promote a more engaged user, overuse will be at the detriment to a retailer's bottom line, enforcing that the user should stick to the products they are familiar with and not try anything new. As D. Roy and M. Dutta found, hybrid approaches to recommender systems generally see increased performances over purely model-based recommendation systems [1]. Curated content should serve alongside more personalized recommendations to maximize effectiveness.

One unique feature of the that the prediction model takes into account is whether if a wine or spirits product is part of the retailers "Winery Direct" line or "Spirits Direct" line of products, respectively. These two product lines are purchased without a middleman, putting the retailer into similar shoes as a wholesaler. Products under these lines are sold at traditionally less than what they would be if a distributor was in the mix. Some of the money saved by cutting out the distributor is passed along to the customer, while some is kept by the retailer in the form of higher margin rates. Some customers are less willing to purchase these "Winery Direct" and "Spirits Direct" products due to brand loyalty with national brands, but implementing a weighted recommendation system can help give customers the chance to see the "Direct" line of products that are similar to what they have already seen on the website and purchased online and in stores.

Exclusively online customers are at a disadvantage to their in-store and omni-channel counterparts. It is difficult to replicate the in-store experience that a knowledgeable salesperson can provide. Salespeople can provide recommendations based on their tastes as well as in-person samples of wine and spirits. The retailer trains their staff to rarely decline a customers request to sample before purchasing, for all but the most expensive of bottles. Looking at the in-store purchasing behavior shows the effects of this with the basket rates of their “Winery Direct” and “Spirits Direct” products. Salespeople can break customers out of bubbles where they are purchasing the same products repeatedly. Customers who shop online tend to explore less, sticking to their tried and true, or purely just look at the bottles that are hundreds and hundreds of dollars without ever making a purchase. As with any omni-channel business, conversion rates are much higher in-store than online.

In addition to analyzing what customers purchase, understanding online browsing behavior is essential in tailoring effective recommendations. Online browsing history provides insights into products a customer may show interest in but does not necessarily buy, akin to traditional window shopping. This aspect of customer behavior, often overlooked in traditional retail settings, can be a valuable source of information for an omni-channel retailer. Recommending a high-ticket item that a customer has browsed multiple times, especially during strategic times like the holiday season, can serve as a catalyst for conversion. By leveraging the predictive capabilities of the machine learning model developed in this study, the retailer can strategically present targeted recommendations that align with a customer's interests, driving not only engagement but also conversion rates. Understanding the nuances of online browsing behavior complements the focus on historical purchasing data, providing a more comprehensive understanding of customer preferences and enhancing the overall effectiveness of the recommendation system.

II. METHODOLOGY

A. *Background & Reasoning*

My current role as a data analyst for a national wine & spirits retailer grants me the opportunity to A/B test changes to the front-end of their website and app. Although, a back-end upgrade to the product recommendation engine will require a greater level of research to justify an investment from my employer. With a constrained development team and a packed roadmap, and deviations from quick wins and fill-ins on the action priority matrix must forego more scrutiny.

The retailer currently holds all their data between Adobe Analytics and an internal relational database, in which I have been granted permission to use for research purposes. The stipulation being that I must refrain from using their name and I must subset the data in some fashion to conceal full sales and web traffic information.

I planned to test multiple different classical machine learning algorithms with the dataset and anticipated that a starting point would be logistic regression. When training the model, the distributed computing framework, Apache Spark, helped process the large dataset. A positive to Spark implementation is that it is scalable and can eventually enable real-time product recommendations. A locally run version of the prediction model will suffice as a proof of concept, although if a variation of the system is fully implemented to a website and mobile app, cloud services such as Amazon Web Services Elastic Compute Cloud may ensure more a reliable recommendation system.

B. *Wine & Spirits Query*

The data used for this study was pulled directly from the relational database of the national wine & spirits retailer, although the data was modified to mask some confidential information, such as sales dollars, sales units, and the true gross margin rate. A sample of five thousand wines and spirits was chosen to be representative of the entire catalog of products that the retailer offers. A restraint of this size, cutting out thousands of products, was requested by the retailer to safeguard any financial information that they did not want available to outside parties.

The final query that was used to garner a list of three thousand wines and two thousand spirits that was used in the final study extracted the following:

- Product ID
- Product name
- Sales Strategy
- High-level department
- Medium-level department
- Low-level department
- Taste characteristics
- Number of units sold
- Sales dollars
- Gross margin dollars
- Unit price

Limiting the analysis to wine and spirits meant that other categories were not represented in this study. Beer, cigars, glassware, and any other general merchandise were excluded entirely. Additionally, only standard 750ml bottles were represented in the five thousand bottle dataset, leaving out piccolo, demi, liter, magnum, and jeroboam sized bottles as well as minis and handles of liquor.

Not all products have their taste profiles listed within the retailer’s internal database, so when querying the list of products some were dropped from the results. The prediction model was intended to index taste profiles to help identify the likelihood of a given customer purchasing a bottle, so this further constrains the dataset. The website has taste profiles available as well as product descriptions that could have been extracted to help expand the list of products to more popular items that were not well documented internally, but any form of web scraping was forbidden.

C. Customer Visits Query

With product information established, the query was also reliant upon a customer's website history as well as their online and in-store purchase history. Starting with the customer journey on the website, any visit data was intended to track what product pages (PDPs) that a user touched throughout all of their visits to the retailer's website. While, eventually tracking customers via third party cookies, such as an Adobe Visitor ID may provide relevant advertising information on a traditionally non-transactional shopper, an internal loyalty ID helped us track known customers. Customers with a loyalty ID have willingly signed up for the retailer's loyalty program either in-store or online. That is to say, they are typically more fruitful customers than someone who chooses to stay "un-loyal" to the retailer despite any of the perks associated with signing up, such as digital and physical coupons, other discounts, and class vouchers for wine and spirit courses. Using this group as a sample should not be representative of the entire customer base, as they are the most involved customers. They are willing to visit more often, spend more money, and purchase more "Winery Direct" and "Spirits Direct" products than a traditional "un-loyal" customer. Customers with loyalty IDs had their loyalty ID, date of website visit, and product ID viewed extracted from the retailer's relational database from a time period of the last 12 fiscal months.

D. Customer Transaction Query

Transactional data of the customers described in the previous paragraph was also pulled in a similar manner. Customers with loyalty IDs had their loyalty ID, date of purchase, and product ID purchased extracted from the retailer's relational database within the same time period. The visits dataset, online transaction dataset, and in-store transaction dataset were all merged into one dataset with a binary indicator that represented if a customer purchased the 750ml wine or spirit bottle on a given day.

E. Joining Datasets

The following information/features were used to construct a dataset with enough information for machine learning algorithms to make informed predictions on. The information found in F, G, and H, online browsing history, shopping history, and product information, respectively were pulled from a database using SQL queries. The information found in I, calculated information, was achieved by obtaining the gross margin rate of each product as well as determining how popular a product was in their low-level department bubble. For instance, how popular, unit sales wise, is a particular Kentucky straight bourbon whiskey when compared to all other Kentucky straight bourbon whiskies within the five thousand bottle sample.

F. Online Browsing History

- CUSTOMER_ID: long
- DATE: string
- PRODUCT: integer

G. Shopping History

- CUSTOMER_ID: long
- DATE: string
- PURCHASE: integer
- PRODUCT: integer

H. Product Information

- PRODUCT_NAME: string
- DIRECT: integer
- DEPT_1: string
- DEPT_2: string
- DEPT_3: string
- CHAR_1: string
- CHAR_2: string
- UNIT_PRICE: double

I. Calculated Information

- MARGIN_RATE: double
- POPULARITY: double

J. Data Importation with Python and Spark

The final combined query was performed, taking several hours to pull from Google BigQuery and produced a dataset of over five hundred fifty thousand rows that followed the comma separated value (CSV) formatting.

Google Colab offered free access to the Python 3 Google Compute Engine backend, granting access to a runtime with 12.7 GB of system random access memory and 107 GB of disk space for the project. Natural integration with the PySpark library allowed for a spark session to start natively within Google Colab's Python notebook under the app name "WineSpiritsRecommendation". Spark was able to read the CSV file and infer the schema.

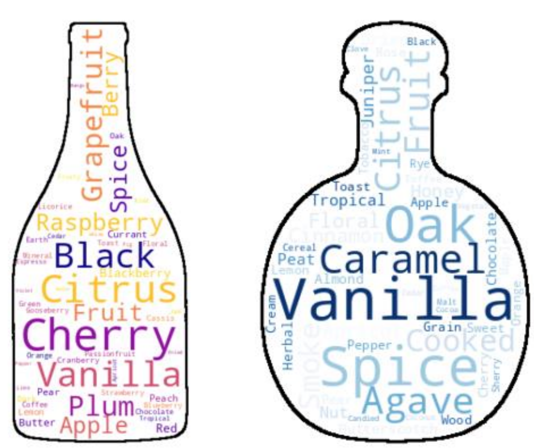
K. Data Validation

To validate the data after it was imported, several measures were taken. Printing the schema confirmed that Spark's schema inference was made correctly and is pictured below this paragraph. Counting the number of rows ensured that no rows of data were lost, while showing the first ten rows confirmed that all 14 columns were present. Lastly, checking for any missing values was a necessity, especially for the departments and characteristics of products which were not all available in the retailer's database. If there were any missing rows, several machine learning prediction models would struggle to make predictions accurately or run into errors that stop it from running at all.

```
root
 |-- CUSTOMER_ID: long (nullable = true)
 |-- DATE: string (nullable = true)
 |-- PURCHASE: integer (nullable = true)
 |-- PRODUCT: integer (nullable = true)
 |-- PRODUCT_NAME: string (nullable = true)
 |-- DIRECT: integer (nullable = true)
 |-- DEPT_1: string (nullable = true)
 |-- DEPT_2: string (nullable = true)
 |-- DEPT_3: string (nullable = true)
 |-- CHAR_1: string (nullable = true)
 |-- CHAR_2: string (nullable = true)
 |-- UNIT_PRICE: double (nullable = true)
 |-- MARGIN_RATE: double (nullable = true)
 |-- POPULARITY: double (nullable = true)

Number of rows: 551076
```

Another exercise to validate the two problematic characteristics columns was to generate word clouds using the wordcloud Python library. This was accomplished by separating the complete data frame into two, one for wine, and one for spirits. Joining the text from the two characteristics columns and splitting the text allowed me to count the word frequency and create an array with Python's NumPy library. The library is open source and enables arrays and matrices within Python, alongside mathematical functions, and applications to process data in many areas of data science and machine learning. I plotted the wine and spirits arrays with masks of a magnum bottle of pinot noir and a standard seven hundred fifty ml bottle of tequila, respectively.



L. Spark Pipeline

Through the combination of online browsing history, in-store transactions, and online transactions, any attempted model was intended to predict if a customer is willing to purchase a product online with a binary "yes" or "no". With a large array of products that differ from one another to various degrees, not all products that are granted a "yes" can be displayed on the user's screen at once. Instead, a probability score between zero and one was assigned to products so that only the products with the highest likelihood of being added to a user's cart would be recommended.

To orchestrate data preprocessing with Spark, a pipeline was an effective measure to organize the various steps needed to prepare the data, engineer features, and train a machine learning model within a single, coherent structure. Pipelines were particularly advantageous in Spark due to their ability to automate repetitive tasks and seamlessly apply the same set of operations consistently to both training and test datasets. This helped mitigate the risk of data leakage and ensured the integrity of the machine learning process. Spark also offered a built-in parallel processing function that enhanced the overall efficiency when processing big data using various techniques of data science.

The three high, medium, and low-level department columns as well as the two taste characteristics columns were all categorical features within the data frame. Effective handling of these categorical features, or any categorical features, was imperative for accurate predictions within a logistic regression model. String indexing played a crucial role in the process of numerical encoding by assigning unique numerical identifiers to the distinct categories. This facilitated the translation of qualitative information into a format compatible with the computations that logistic regression, and other classical machine learning models performs. PySpark offers its "StringIndexer" class to accomplish this task.

After implementing a method to complete string indexing, a method to assemble the indexes into a vector was necessary to aid in the implementation of logistic regression using PySpark. Within the framework of a prediction model, the assembly of feature columns into a vector seamlessly combined individual feature columns into a single vectorized format. The various columns were transformed into a dense vector representation that could be natively engaged as the input for the logistic regression model. This ensured that the logistic regression algorithm could effectively process and learn from the input features during the model training phase. The utilization of PySpark's "VectorAssembler" helped facilitate this process into the machine learning pipeline, and included not only the three high, medium, and low-level department columns and the two taste characteristics columns, but the unit price, margin rate, and popularity of each product.

Next in the pipeline was to feed the features into a logistic regression model, with the primary goal of predicting which binary output the purchase column was for each product a customer saw. After splitting the data frame with a random 80/20 split, the predictions generated relatively quickly when considering the amount of data processed. This is because Spark is well known for it's efficient solutions in terms of execution time, albeit with performance heavily based on the amount of memory available [2].

To assess the model's performance, several of Spark's out-of-the-box evaluation metrics were used. This included the area under the receiver operating curve, the accuracy, the precision, the recall, and the F1 score.

Due to the pipeline that Spark has access to, it was relatively straightforward to implement other machine learning methods without writing a lot of new code. One of the other pipelines I created was based on a gradient boosted trees (GBT) classifier. The classifier could reuse the features that were created for the logistic regression pipeline. I chose this specific approach because the GBT classifier sees success in solving for class imbalances by selecting the most influential features from a given data frame. GBT is also relatively quick to perform, similar to classical logistic regression, so an increase in performance may warrant a slightly longer run time.

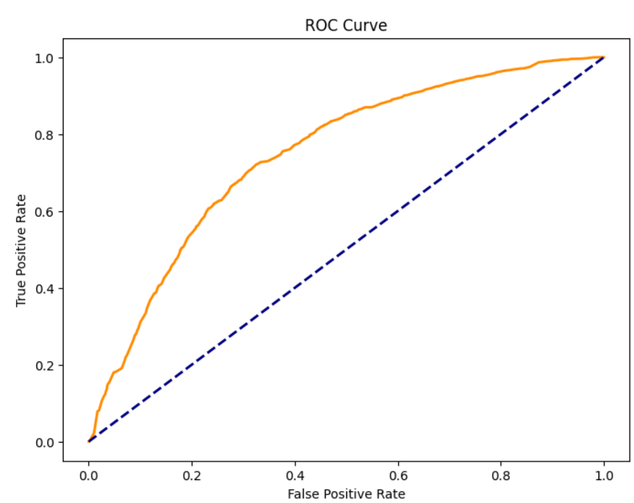
III. Results & Discussion

The logistic regression and gradient boosted tree classifier pipelines both performed well, given the constraints of the dataset. Ultimately, the highest performer in all measures of performance, the area under the receiver operating curve, the accuracy, the precision, the recall, and the F1 score, was the gradient boosted tree model.

While the best method to evaluate the performance is argued [4], many believe that the area under the receiver operating curve is the best option to reflect performance when making predictions for data frames with significant class imbalances, such as our combination data frame that combined online browsing history, online shopping history, and in-store shopping history of the retailer's loyal customers.

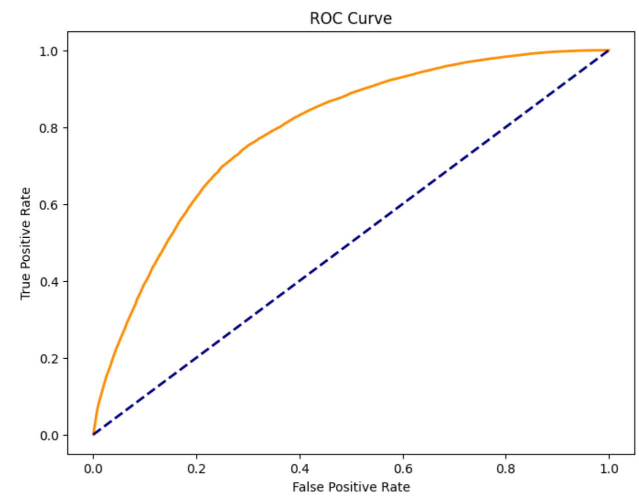
The logistic regression pipeline resulted in a 75% probability of correctly distinguishing between a purchased item and non-purchased item, which is an acceptable starting point with limited data. A more robust data set that more accurately represents the entire population of the retailer's customers should improve this score to achieve at least an 80% probability.

```
Logistic Regression Pipeline
|-- Area Under ROC: 0.75
|-- Accuracy:       0.73
|-- Precision:      0.72
|-- Recall:         0.73
|-- F1 Score:       0.71
```



The gradient boosted tree pipeline resulted in a better probability of 79%. As you can see in the ROC curve charted below, the curve is much smoother and less jagged than the logistic regression model. The smooth curve indicates that the predictions are much better calibrated across the different threshold values and should provide more stable performance than a more bumpy or jagged model, such as the logistic regression ROC curve.

```
Gradient Boosted Tree Pipeline
|-- Area Under ROC: 0.79
|-- Accuracy:       0.76
|-- Precision:      0.75
|-- Recall:         0.76
|-- F1 Score:       0.75
```



IV. Next Steps

In addition to the predictions based on the subset of data, the added functionality of building a custom customer profile is available at the bottom of the code to simulate how the model would make predictions for a brand-new user with only online browsing history and no shopping history available. This simulation provides an solution to not being able to use third party customer data to identify non-transactional un-loyal customers who were excluded from the data frame at the request of my employer.

For instance, with little to no data on the simulated customer seen in the code who theoretically visited the product pages for three red wines, Shannon Ridge Cabernet Sauvignon, Iter Californian Pinot Noir, and Robert Mondavi's Private Selection Bourbon Barrel Aged Cabernet Sauvignon, the model is largely based on the popularity and price of each product. When properly implemented, the departments and characteristics of the bottles of wine and spirits should have a higher weight added to them to hone in on each customer's taste in products. The more data on a customer's taste, the better the algorithm should perform.

V. Conclusion

In conclusion, machine learning algorithms, such as logistic regression and gradient boosted tree classifiers can make predictions on what a loyal customer will purchase when given information on their previous online browsing history in addition to their in-store and online shopping history.

With the dust settled from the alcohol industry's boom in 2020, the time to take a step back and realize the opportunities to improve performance is now. The customer journey has forever changed to be omni-channel where a customer shops both in-store at brick-and-mortar locations as well as online storefronts.

Attempting to replicate the in-store experience is difficult to achieve due to the high bar that a salesperson can set when they cater to the customer. A salesperson can provide personalized product recommendations on a customer-by-customer basis, upselling and adding on to their purchase while making sure that they are bringing products to the table that the customer will actually like. A salesperson can provide a tasting of wine and spirits and make suggestions on how a spirit can be mixed into a cocktail. Trying to replicate this online is reliant on curated content which requires a team of content producers to manage. Even so, many may look past the curated content and simply search and browse on the website, opting out of using any articles, listicles, or tutorials on the website.

By utilizing a prediction model as the groundwork for a full recommendation system for a national chain of wine and spirits retailers, customers can be served what they want to purchase based off of their tastes. The products are the heart of the website, so a customer will not have to go out of their way to find them, as is needed with curated content designed by the retailer's content team.

The gradient boosted tree classifier seems to be an excellent starting point to deploy with an uncompromised version of the data frame. The uncompromised version should include non-transactional customers, the entire catalog of wine, spirits, and beer, and should be less concerned with privacy due to its intended internal-only access.

The gradient boosted tree classifier held an edge offer logistic regression because it automatically starts to learn feature importance, resulting in a high-performance ROC curve with a smooth and efficient flow that enables quick computation. This is a benefit that should be seen when Spark enables the recommendation model to be easily scaled up onto the retailer's website, learning more and more about the customer's tastes with each visit, even if it is non-transactional. Additionally, manual weights can be added to prioritize products from the "Winery Direct" or "Spirits Direct" line of products, as well as gross margin. Sacrificing the most efficient model in terms of conversion rate may be beneficial if it boosts these products with higher margin rates over low margin rate national brand wine and spirits. Beer has a very limited "Brewery Direct" line, so it will likely be left out of that process.

A specific audience that should see the most benefit from the implementation will be the online "window-shoppers" who frequently visit the website to look at high-ticket items such as Clase Azul Reposado Tequila, or Blanton's Single Barrel Kentucky Bourbon. While these shoppers may not purchase the expensive or frequently out of stock items, the recommendation model can suggest items of a lower price-point with similar appeal and encourage a conversion. I anticipate these historically non-converting customers will be the hardest to appeal to.

VI. References

[1] D. Roy and M. Dutta, "A systematic review and research perspective on recommender systems," Journal of Big Data, vol. 9, no. 1, May 2022

[2] N. Ahmed, A. L. C. Barczak, T. Susnjak, and M. A. Rashid, "A comprehensive performance analysis of Apache Hadoop and Apache Spark for large scale data sets using HiBench," *Journal of Big Data*, vol. 7, no. 1, Dec. 2020

[3] T. Hasanin, T. M. Khoshgoftaar, J. L. Leevy, and N. Seliya, "Examining characteristics of predictive models with imbalanced big data," *Journal of Big Data*, vol. 6, no. 1, Jul. 2019

[4] M. J. Pencina, R. B. D' Agostino, R. B. D' Agostino, and R. S. Vasan, "Evaluating the added predictive ability of a new marker: From area under the ROC curve to reclassification and beyond," *Statistics in Medicine*, vol. 27, no. 2, pp. 157–172, 2007