RETAIL CLICKSTREAM ANALYSIS AND PREDICTION

PROJECT REPORT for CS-GY 6513 BIG DATA

at

NYU

Submitted by

NIHARIKA KRISHNAN (nk2982)

VAIBHAV SINGH (vs2410)

CONTENTS

INTRODUCTION

Over the last few years, e-commerce has become an indispensable part of the global retail framework. Like many other industries, the retail landscape has undergone a substantial transformation following the advent of the internet, and thanks to the ongoing digitalization of modern life, consumers from virtually every country now profit from the perks of online transactions. According to statistics, in the year 2021, almost 2 Billion people purchased goods online worldwide resulting in USD 4.9 Trillion dollars in revenue. The key drivers of success over the next decade will be centered on building a deep understanding of and connection to the empowered consumer, and the only way to understand consumer behavior is to measure and analyze. Hence, retailers work towards gathering data from different areas that have an impact on their online store and use it to understand the trends and the shift in consumers' behavior to make data-driven decisions that will drive more online sales.

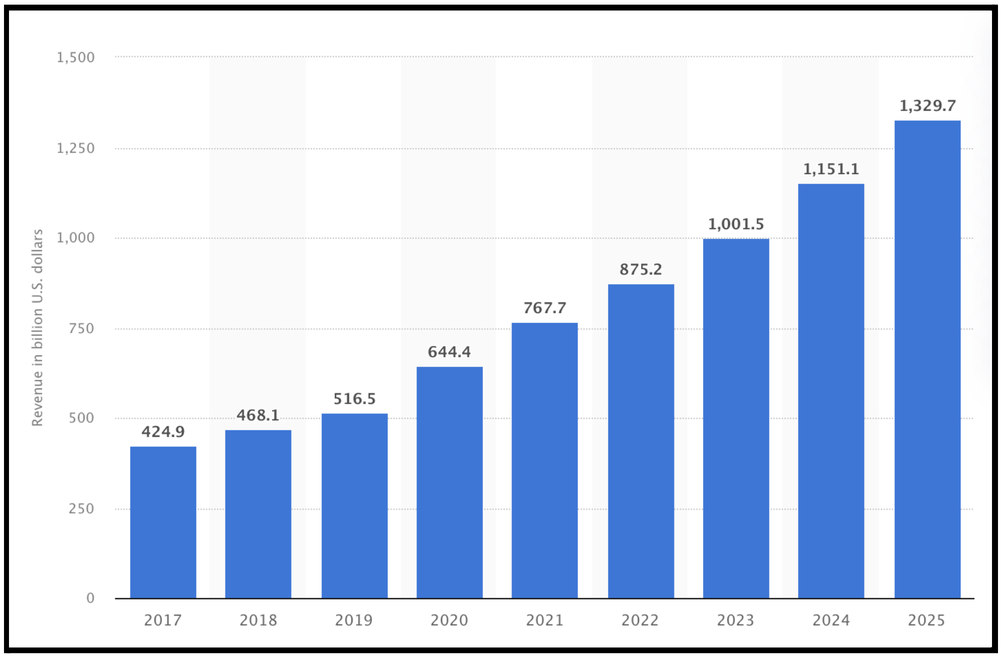


Figure-1: Retail e-commerce revenue in the US from 2017 to 2025 *(in billion U.S. dollars)*

Clickstream Data is the user's digital footprint left on a specific website during a browsing session. Clickstream analytics is the process of collecting, analyzing, and reporting aggregate data regarding pages a user visits on a website and the order in which they visit them. The path a user takes through a website is called the clickstream.

ARCHITECTURE

The data layer is currently a single CSV file, and this can be scaled with the help of any current big data storage solution like Amazon S3 or MongoDB. To perform big data analytics we have chosen Pyspark which is the Python API for Apache Spark, an open-source, distributed computing framework with a set of libraries for real-time, large-scale data processing.

For visualization, we have used Matplotlib and Plotly which is an interactive, open-source, and browser-based graphing library for Python. Finally, we used Streamlit, which is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. This is used in predictive analytics to simulate a real-time deployment.

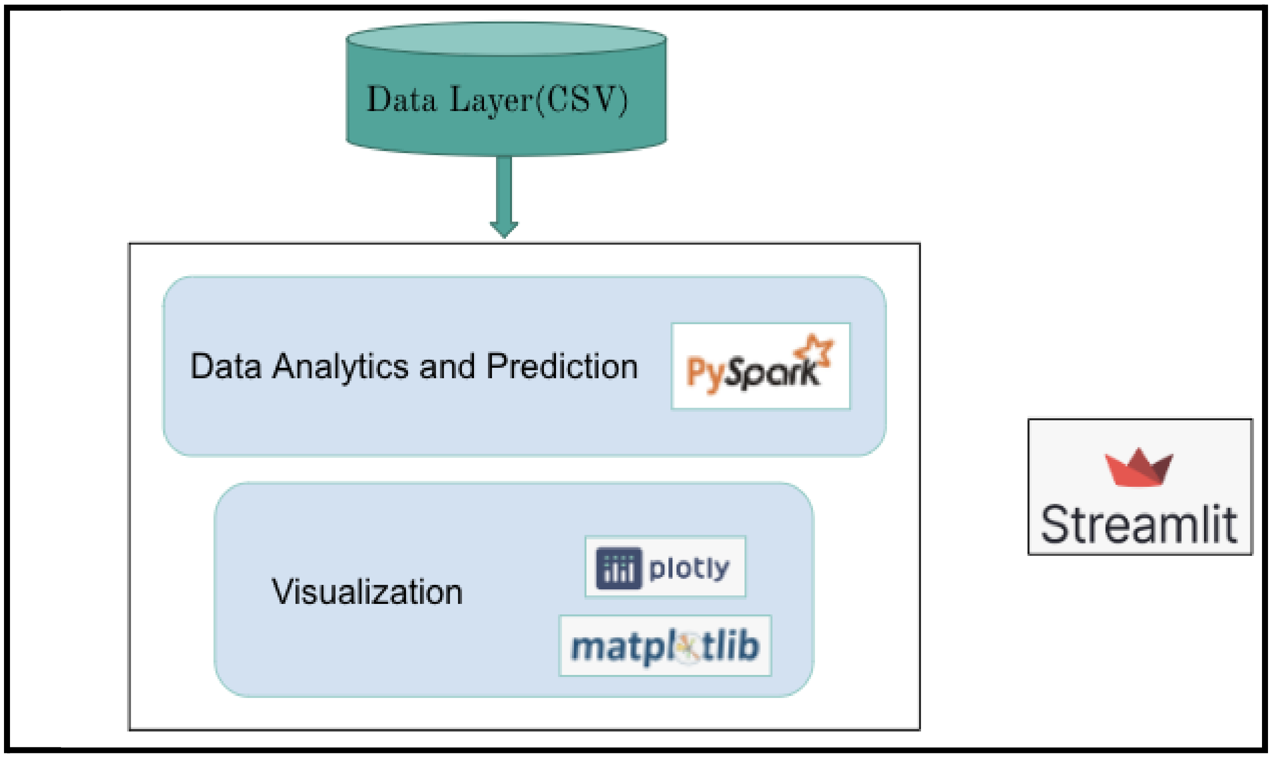


Figure-2: Architecture

DATASET

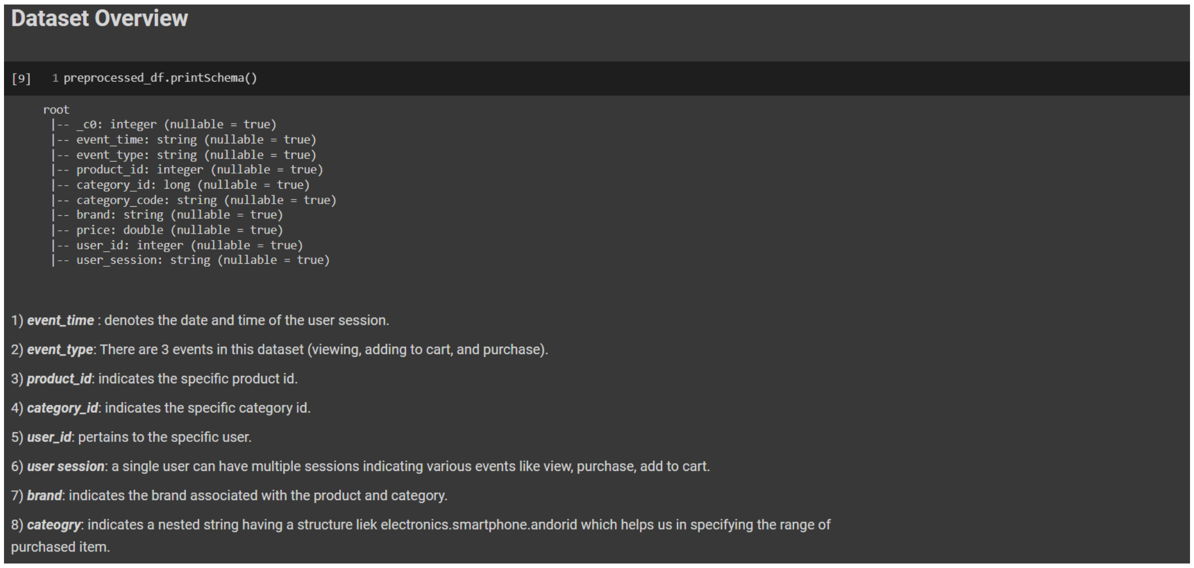
Dataset used for this Big Data project is clickstream user behavior data for October 2019 available from a large multi-category online store.

Dataset Source: It was collected by the Open CDP project

Dataset Size: 5.6 GB (Oct 2019)

If data is collected for one year, this would be an estimated 60GB+, making centralized computing futile in both offline and real-time inference. Hence, use of Big Data technologies is imperative to make analytics computationally fast leading to valuable insights and making critical decisions.

Dataset Overview:



Dataset Overview

```
[9]   preprocessed_df.printSchema()
```

```
root
 |-- _c0: integer (nullable = true)
 |-- event_time: string (nullable = true)
 |-- event_type: string (nullable = true)
 |-- product_id: integer (nullable = true)
 |-- category_id: long (nullable = true)
 |-- category_code: string (nullable = true)
 |-- brand: string (nullable = true)
 |-- price: double (nullable = true)
 |-- user_id: integer (nullable = true)
 |-- user_session: string (nullable = true)
```

1) ***event_time*** : denotes the date and time of the user session.

2) ***event_type***: There are 3 events in this dataset (viewing, adding to cart, and purchase).

3) ***product_id***: indicates the specific product id.

4) ***category_id***: indicates the specific category id.

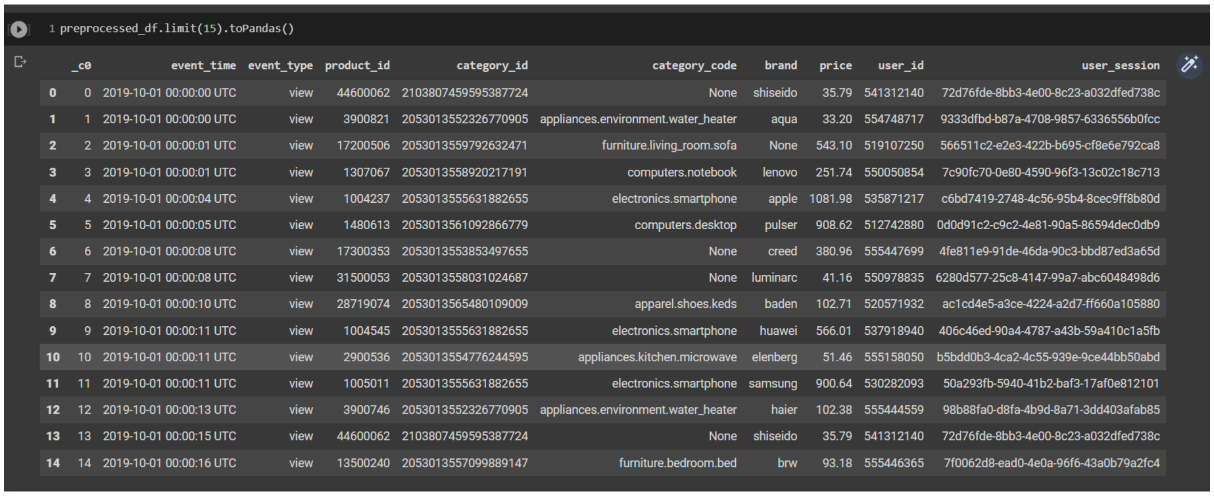
5) ***user_id***: pertains to the specific user.

6) ***user session***: a single user can have multiple sessions indicating various events like view, purchase, add to cart.

7) ***brand***: indicates the brand associated with the product and category.

8) ***cateogry***: indicates a nested string having a structure liek electronics.smartphone.andorid which helps us in specifying the range of purchased item.

Dataset without Pre-Processing:



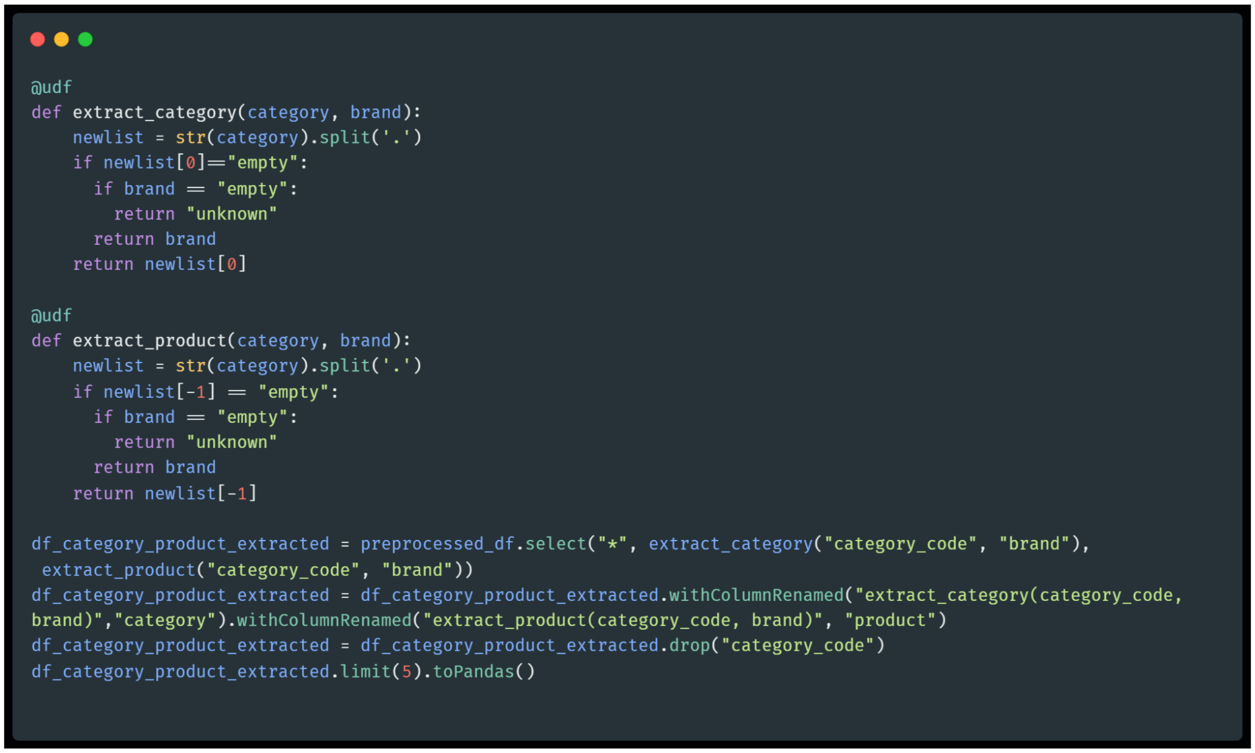
```
preprocessed_df.limit(15).toPandas()
```

	_c0	event_time	event_type	product_id	category_id	category_code	brand	price	user_id	user_session
0	0	2019-10-01 00:00:00 UTC	view	44600062	2103807459595387724	None	shiseido	35.79	541312140	72d76fde-8bb3-4e00-8c23-a032dfed738c
1	1	2019-10-01 00:00:00 UTC	view	3900821	2053013552326770905	appliances.environment.water_heater	aqua	33.20	554748717	9333dfbd-b87a-4708-9857-6336556b0fcc
2	2	2019-10-01 00:00:01 UTC	view	17200506	2053013559792632471	furniture.living_room.sofa	None	543.10	519107250	566511c2-e2e3-422b-b695-cf8e6e792ca8
3	3	2019-10-01 00:00:01 UTC	view	1307067	2053013558920217191	computers.notebook	lenovo	251.74	550050854	7c90fc70-0e80-4590-96f3-13c02c18c713
4	4	2019-10-01 00:00:04 UTC	view	1004237	2053013555631882655	electronics.smartphone	apple	1081.98	535871217	c6bd7419-2748-4c56-95b4-8cec9ff8b80d
5	5	2019-10-01 00:00:05 UTC	view	1480613	2053013561092866779	computers.desktop	pulser	908.62	512742880	0d0d91c2-c9c2-4e81-90a5-86594dec0db9
6	6	2019-10-01 00:00:08 UTC	view	17300353	2053013553853497655	None	creed	380.96	555447699	4fe811e9-91de-46da-90c3-bbd87ed3a65d
7	7	2019-10-01 00:00:08 UTC	view	31500053	2053013558031024687	None	luminarc	41.16	550978835	6280d577-25c8-4147-99a7-abc6048498d6
8	8	2019-10-01 00:00:10 UTC	view	28719074	2053013565480109009	apparel.shoes.keds	baden	102.71	520571932	ac1cd4e5-a3ce-4224-a2d7-ff660a105880
9	9	2019-10-01 00:00:11 UTC	view	1004545	2053013555631882655	electronics.smartphone	huawei	566.01	537918940	406c46ed-90a4-4787-a43b-59a410c1a5fb
10	10	2019-10-01 00:00:11 UTC	view	2900536	2053013554776244595	appliances.kitchen.microwave	elenberg	51.46	555158050	b5bdd0b3-4ca2-4c55-939e-9ce44bb50abd
11	11	2019-10-01 00:00:11 UTC	view	1005011	2053013555631882655	electronics.smartphone	samsung	900.64	530282093	50a293fb-5940-41b2-baf3-17af0e812101
12	12	2019-10-01 00:00:13 UTC	view	3900746	2053013552326770905	appliances.environment.water_heater	haier	102.38	555444559	98b88fa0-d8fa-4b9d-8a71-3dd403afab85
13	13	2019-10-01 00:00:15 UTC	view	44600062	2103807459595387724	None	shiseido	35.79	541312140	72d76fde-8bb3-4e00-8c23-a032dfed738c
14	14	2019-10-01 00:00:16 UTC	view	13500240	2053013557099889147	furniture.bedroom.bed	brw	93.18	555446365	7f0062d8-ead0-4e0a-96f6-43a0b79a2fc4

DATA PROCESSING AND IMPUTATION

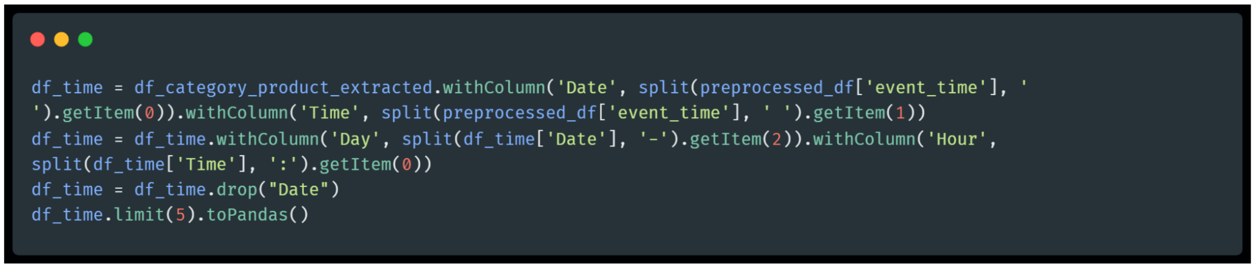
Data preprocessing and imputation are some of the most important steps in the analytics pipeline that help in providing clean, usable and efficient data. We extract informative features like Category and Product from Category_Code, Date and Time from Timestamp and handle Null values in our category column by imputing them with brand values.

Extract Category & Product from Category Code:



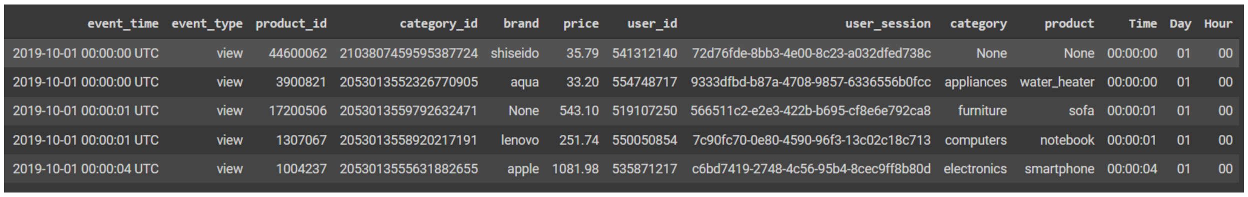
```python
@udf
def extract_category(category, brand):
    newlist = str(category).split('.')
    if newlist[0]=="empty":
      if brand == "empty":
        return "unknown"
      return brand
    return newlist[0]

@udf
def extract_product(category, brand):
    newlist = str(category).split('.')
    if newlist[-1] == "empty":
      if brand == "empty":
        return "unknown"
      return brand
    return newlist[-1]

df_category_product_extracted = preprocessed_df.select("*", extract_category("category_code", "brand"),
 extract_product("category_code", "brand"))
df_category_product_extracted = df_category_product_extracted.withColumnRenamed("extract_category(category_code,
brand)","category").withColumnRenamed("extract_product(category_code, brand)", "product")
df_category_product_extracted = df_category_product_extracted.drop("category_code")
df_category_product_extracted.limit(5).toPandas()
```

Extract Date & Hour from Timestamp:



```python
df_time = df_category_product_extracted.withColumn('Date', split(preprocessed_df['event_time'], '
').getItem(0)).withColumn('Time', split(preprocessed_df['event_time'], ' ').getItem(1))
df_time = df_time.withColumn('Day', split(df_time['Date'], '-').getItem(2)).withColumn('Hour',
split(df_time['Time'], ':').getItem(0))
df_time = df_time.drop("Date")
df_time.limit(5).toPandas()
```

Processed Dataset



event_time	event_type	product_id	category_id	brand	price	user_id	user_session	category	product	Time	Day	Hour
2019-10-01 00:00:00 UTC	view	44600062	2103807459595387724	shiseido	35.79	541312140	72d76fde-8bb3-4e00-8c23-a032dfed738c	None	None	00:00:00	01	00
2019-10-01 00:00:00 UTC	view	3900821	2053013552326770905	aqua	33.20	554748717	9333dfbd-b87a-4708-9857-6336556b0fcc	appliances	water_heater	00:00:00	01	00
2019-10-01 00:00:01 UTC	view	17200506	2053013559792632471	None	543.10	519107250	566511c2-e2e3-422b-b695-cf8e6e792ca8	furniture	sofa	00:00:01	01	00
2019-10-01 00:00:01 UTC	view	1307067	2053013558920217191	lenovo	251.74	550050854	7c90fc70-0e80-4590-96f3-13c02c18c713	computers	notebook	00:00:01	01	00
2019-10-01 00:00:04 UTC	view	1004237	2053013555631882655	apple	1081.98	535871217	c6bd7419-2748-4c56-95b4-8cec9ff8b80d	electronics	smartphone	00:00:04	01	00

OBJECTIVES

The goal of the project is to use this processed clickstream data and big data techniques:

- To Analyze key performance indicators and find insights to improve revenue
- Provide insights for personalized digital marketing
- Measure marketing efforts of retailers, and optimize the overall user experience

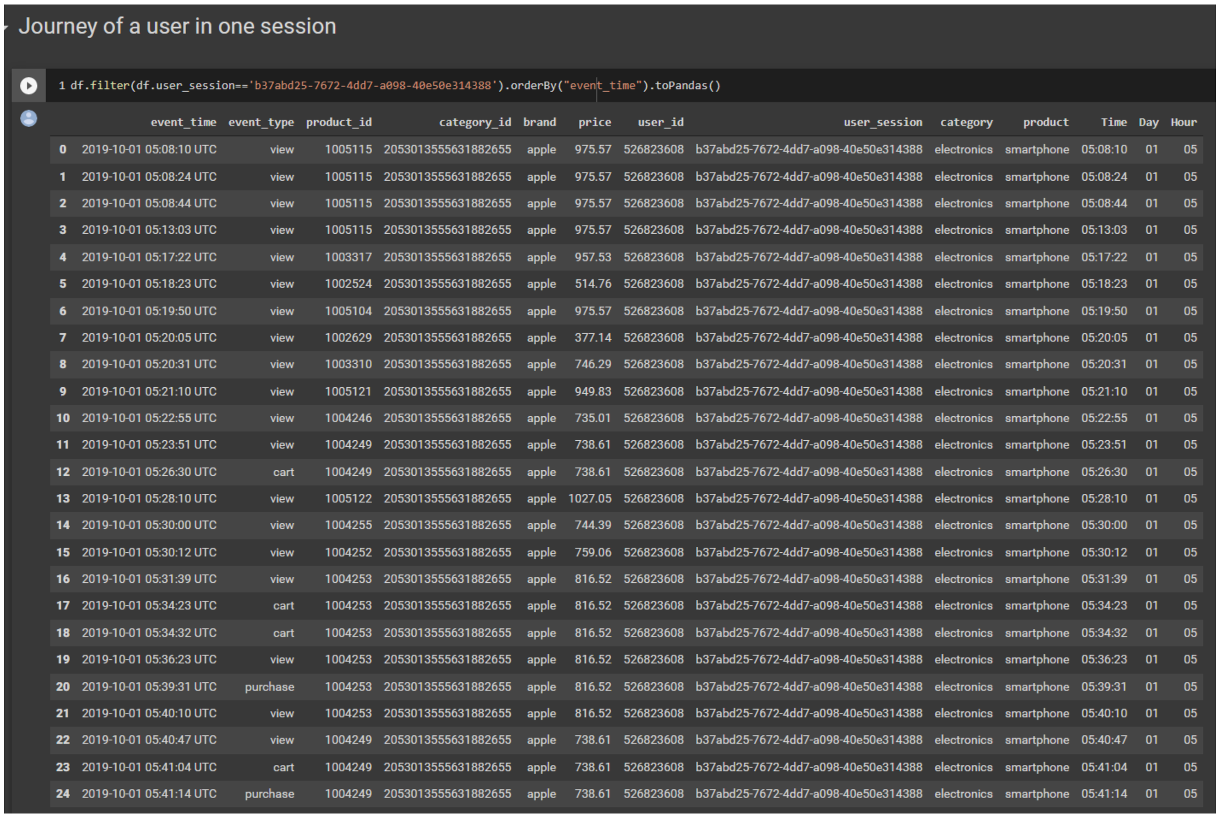
Such analysis will help us gain insights and make critical business decisions. By using historical clickstream data, retailers can get an understanding of user behavior, what products the user is likely to purchase, etc. Hence, the objectives we've explored in this project are:

- Category Analysis
    - Determine best performing categories on the e-commerce site based on purchases
    - Find Brands that generate the highest traction in these best performing categories
- Effect of Adding to Cart
    - Correlation between impact Adding to Cart → Purchase: Cart Conversion Ratio
    - Evaluate Cart Abandonment Rate across categories and brands
- Effect of day-time on purchase trends
    - Analyze the Purchase trends across the month
    - Determine E-Commerce Prime Time
- Build a Real-Time classification model that predicts a purchase using clickstream features

Hence, in this project, we aim to use Big Data techniques on handcrafted features to find insights, correlations and key actions. We also work towards building a predictive modeling Machine Learning pipeline to boost purchases of items in real-time.

TRACKING A USER’S JOURNEY

Querying for a specific user session, we can track a user’s behavior through our website. According to the Figure below, we can infer that the user is interested in buying Apple smartphones. The user browses through various products with different prices, some viewing multiple times and eventually adding to cart and purchasing them.



Journey of a user in one session

```
df.filter(df.user_session=='b37abd25-7672-4dd7-a098-40e50e314388').orderBy("event_time").toPandas()
```

	event_time	event_type	product_id	category_id	brand	price	user_id	user_session	category	product	Time	Day	Hour
0	2019-10-01 05:08:10 UTC	view	1005115	2053013555631882655	apple	975.57	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:08:10	01	05
1	2019-10-01 05:08:24 UTC	view	1005115	2053013555631882655	apple	975.57	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:08:24	01	05
2	2019-10-01 05:08:44 UTC	view	1005115	2053013555631882655	apple	975.57	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:08:44	01	05
3	2019-10-01 05:13:03 UTC	view	1005115	2053013555631882655	apple	975.57	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:13:03	01	05
4	2019-10-01 05:17:22 UTC	view	1003317	2053013555631882655	apple	957.53	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:17:22	01	05
5	2019-10-01 05:18:23 UTC	view	1002524	2053013555631882655	apple	514.76	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:18:23	01	05
6	2019-10-01 05:19:50 UTC	view	1005104	2053013555631882655	apple	975.57	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:19:50	01	05
7	2019-10-01 05:20:05 UTC	view	1002629	2053013555631882655	apple	377.14	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:20:05	01	05
8	2019-10-01 05:20:31 UTC	view	1003310	2053013555631882655	apple	746.29	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:20:31	01	05
9	2019-10-01 05:21:10 UTC	view	1005121	2053013555631882655	apple	949.83	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:21:10	01	05
10	2019-10-01 05:22:55 UTC	view	1004246	2053013555631882655	apple	735.01	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:22:55	01	05
11	2019-10-01 05:23:51 UTC	view	1004249	2053013555631882655	apple	738.61	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:23:51	01	05
12	2019-10-01 05:26:30 UTC	cart	1004249	2053013555631882655	apple	738.61	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:26:30	01	05
13	2019-10-01 05:28:10 UTC	view	1005122	2053013555631882655	apple	1027.05	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:28:10	01	05
14	2019-10-01 05:30:00 UTC	view	1004255	2053013555631882655	apple	744.39	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:30:00	01	05
15	2019-10-01 05:30:12 UTC	view	1004252	2053013555631882655	apple	759.06	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:30:12	01	05
16	2019-10-01 05:31:39 UTC	view	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:31:39	01	05
17	2019-10-01 05:34:23 UTC	cart	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:34:23	01	05
18	2019-10-01 05:34:32 UTC	cart	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:34:32	01	05
19	2019-10-01 05:36:23 UTC	view	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:36:23	01	05
20	2019-10-01 05:39:31 UTC	purchase	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:39:31	01	05
21	2019-10-01 05:40:10 UTC	view	1004253	2053013555631882655	apple	816.52	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:40:10	01	05
22	2019-10-01 05:40:47 UTC	view	1004249	2053013555631882655	apple	738.61	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:40:47	01	05
23	2019-10-01 05:41:04 UTC	cart	1004249	2053013555631882655	apple	738.61	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:41:04	01	05
24	2019-10-01 05:41:14 UTC	purchase	1004249	2053013555631882655	apple	738.61	526823608	b37abd25-7672-4dd7-a098-40e50e314388	electronics	smartphone	05:41:14	01	05

Given below is the eCommerce Conversion Funnel that represents the user journey across 3 different event_types.

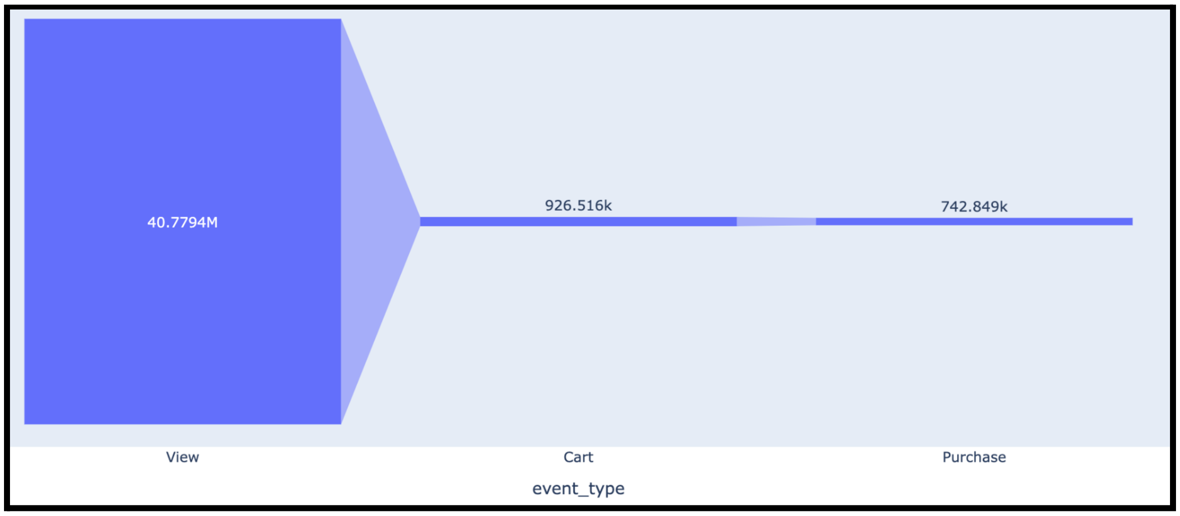


Figure 3: eCommerce Conversion Funnel

OBJECTIVE 1 - CATEGORY ANALYSIS

We grouped data by category to analyze the performance of products and different brands. This can be the first step in attempting to gauge the user's interest and understand what actionable steps can the organization take to increase its sales.

OBJECTIVE 1.a Determining the best performing categories based on purchases.

Techniques: GroupBy, Filters, UDF, Count, Bar Plots

Insights: It is observed that Electronics, Appliances, and Computers are the most Browsed (both Viewed and Carted) and Purchased Categories. Moreover, Electronics is the highest revenue-generating category. We observed an interesting trend in Furniture, which is although highly browsed but has a low purchase rate. This reflects a general notion that people prefer in-person viewing of furniture-based items for comfort and first-hand experience.

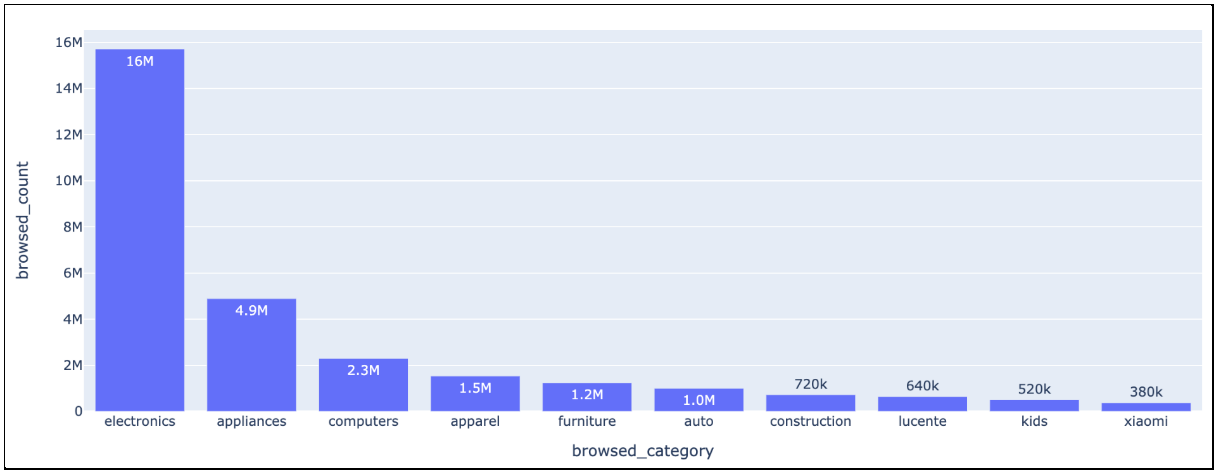


Figure 4: Top Brands in Top Performing Categories

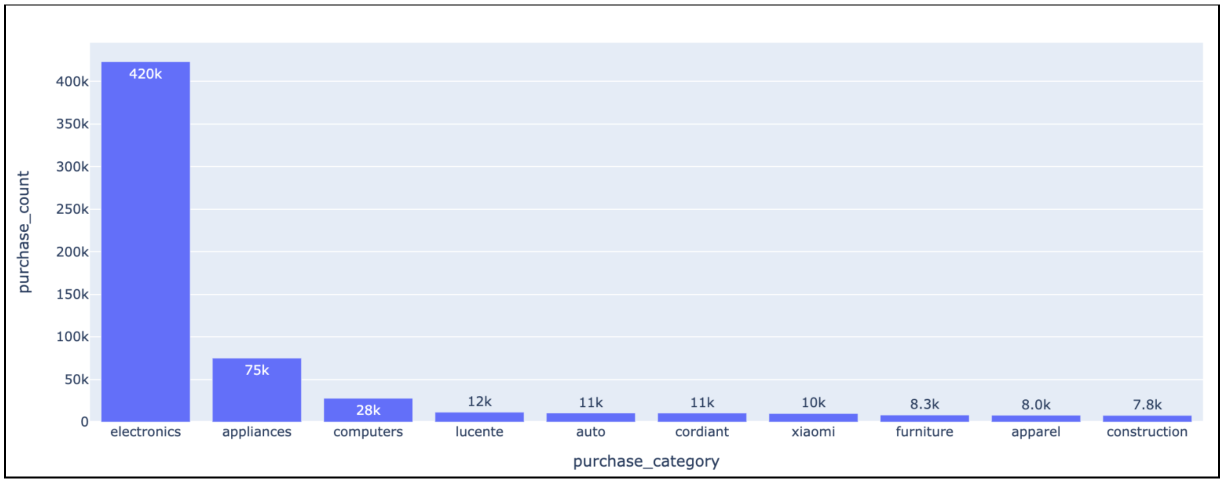
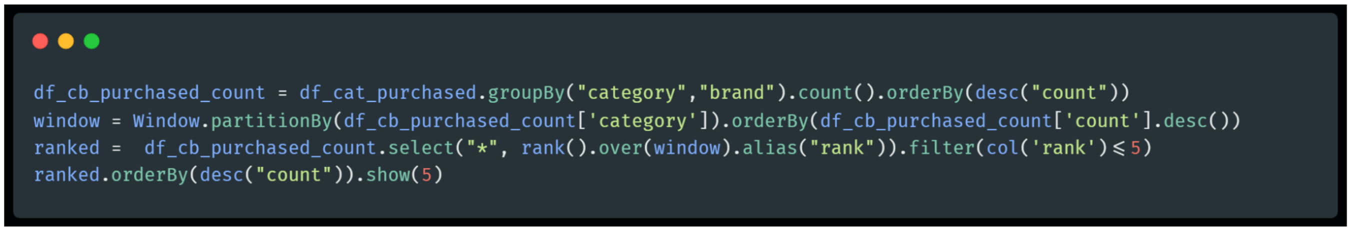


Figure 5: Top Brands in Top Performing Categories

Actions: Since electronics is the highest browsed as well as highest purchased category, allocation of required resources, and data should be done to ensure its performance. Onboarding of new vendors can be done to keep the revenue incoming. On the other hand, we need to analyze the root cause of conversion sales in apparel/furniture. Enhancing customer experience by introducing new 3D viewing techniques can be adopted.

OBJECTIVE 1.b: Determining the best performing product in each of the best performing categories:



```
df_cb_purchased_count = df_cat_purchased.groupBy("category","brand").count().orderBy(desc("count"))
window = Window.partitionBy(df_cb_purchased_count['category']).orderBy(df_cb_purchased_count['count'].desc())
ranked =  df_cb_purchased_count.select("*", rank().over(window).alias("rank")).filter(col('rank')<=5)
ranked.orderBy(desc("count")).show(5)
```

Insights: We went one step further in top-performing categories to analyze top-performing brands as shown in Fig-5. We observed that Samsung has a very strong brand presence across Electronics and Appliances with 159k purchases amounting to nearly 38% of all electronic purchases.

Actions: From these insights, we can take some concrete steps like onboarding more products/vendors under these categories and brands. Identifying the Brand's top user group for personalized targeted marketing can incentivize regular customers to increase their purchases. Tracking brand reviews, and performing sentiment analysis for brands that aren't getting good purchase conversion could potentially be one of the few measures to be taken.

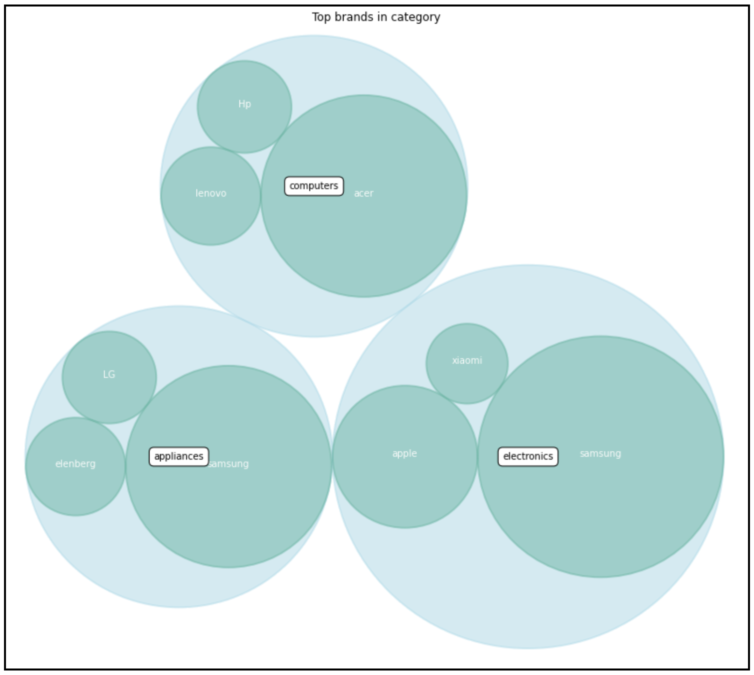


Figure 6: Top Brands in Top Performing Categories

OBJECTIVE 2 - EFFECT OF ADDING TO CART

Here we analyzed the correlation between the impact of adding to cart → purchase: cart conversion ratio and evaluated cart abandonment rate across categories and brands.

Objective 2.a: Correlation between impact Adding to Cart → Purchase: Cart Conversion Ratio

Insights: It's observed that there is an 80% Cart Conversion Ratio (Cart: 926k, Purchase: 742k). We also observe from Fig-7.1 & 7.2 that smartphones are the most added to the cart as well as the highest purchased and have a 63% conversion rate. On the contrary, clocks have a 90% conversion ratio and we believe it is because there are fewer options to choose from, hence the specs are pre-determined leading to a purchase.

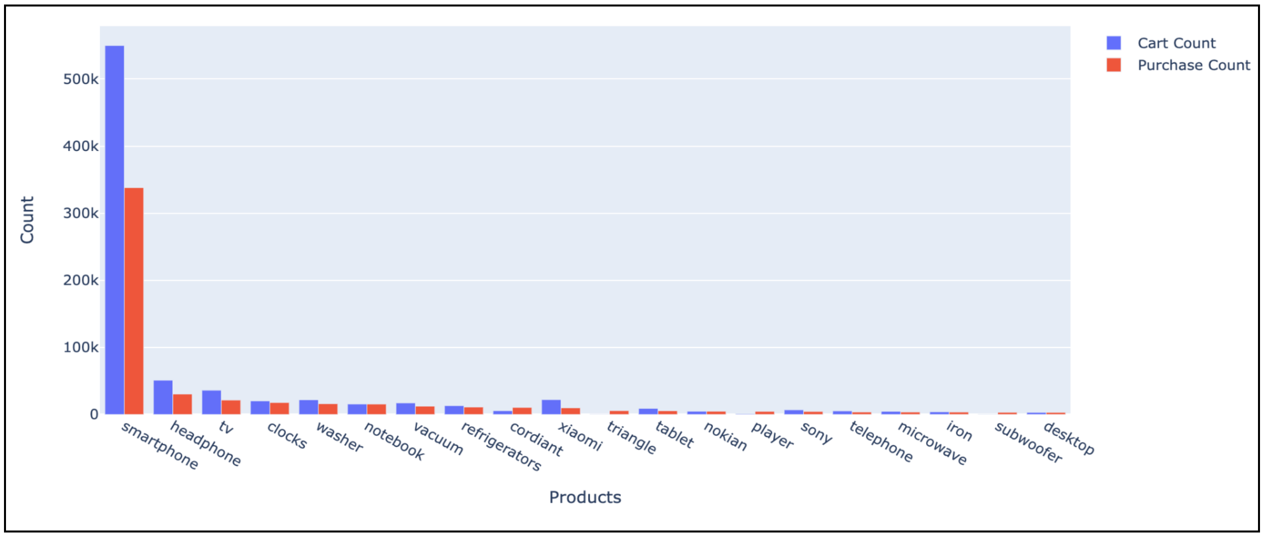


Figure 7.1: Cart vs Purchase count

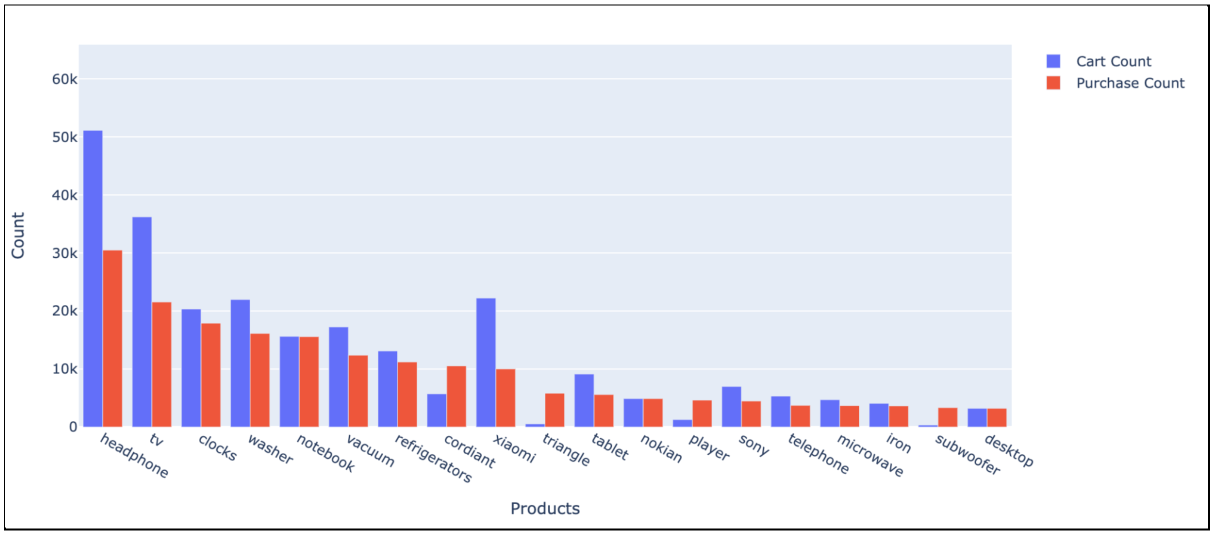
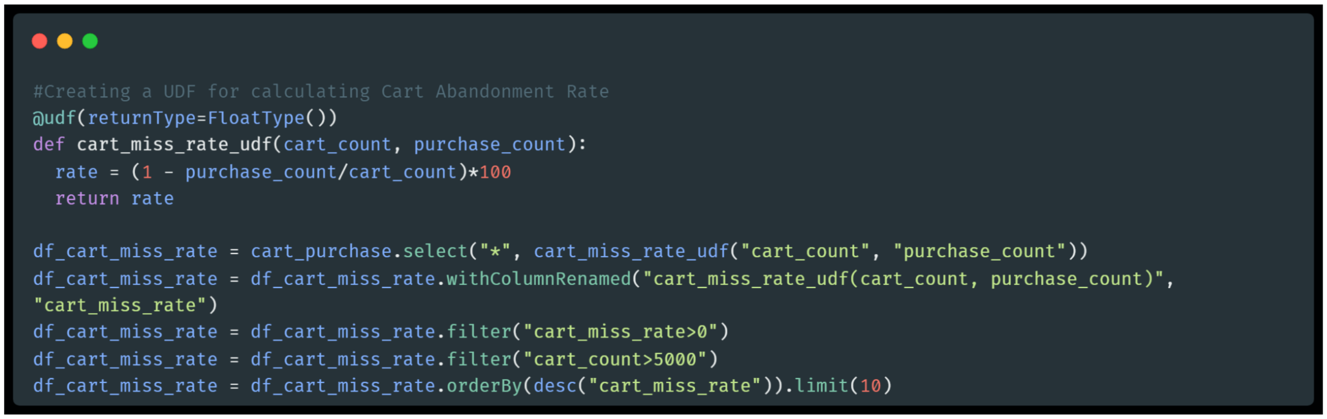


Figure 7.2: Cart vs Purchase count(zoomed in)

Actions: Based on these insights, we can identify key reasons why products already in the cart don't get purchased. The reasons might be due to better deals, return policy, etc from other sites. It's useful to perform A/B testing when introducing a new product or feature in the products.

Objective 2.b: Evaluate Cart Abandonment Rate across categories and brands

Insights: Further drilling down, we calculated cart abandonment rate for categories as well as for brands and observed that Construction and appliances have the highest cart abandonment rate. And although electronics was highly purchased and browsed, it also has a high Cart Abandonment Ratio of 37%. A similar analysis is made on brands and we see that Oppo, Huawei, Xiaomi, Samsung, and Apple have 46%, 44%, 43.5%, 43.2%, and 31% CAR respectively, clearly showing that the Apple brand is more trustworthy with a significantly lower CAR.



```
#Creating a UDF for calculating Cart Abandonment Rate
@udf(returnType=FloatType())
def cart_miss_rate_udf(cart_count, purchase_count):
  rate = (1 - purchase_count/cart_count)*100
  return rate

df_cart_miss_rate = cart_purchase.select("*", cart_miss_rate_udf("cart_count", "purchase_count"))
df_cart_miss_rate = df_cart_miss_rate.withColumnRenamed("cart_miss_rate_udf(cart_count, purchase_count)",
"cart_miss_rate")
df_cart_miss_rate = df_cart_miss_rate.filter("cart_miss_rate>0")
df_cart_miss_rate = df_cart_miss_rate.filter("cart_count>5000")
df_cart_miss_rate = df_cart_miss_rate.orderBy(desc("cart_miss_rate")).limit(10)
```

Actions: To curb this high cart abandonment, we need to thoroughly analyze the return options, shipping costs, and payment methods to understand why and what exactly is causing such items to go unpurchased. Some viable solutions can be partnering with vendors/brands to offer deals and discounts on products that are not converted. Finding specific user groups that purchase these products and provide personalized marketing to customers from these user groups who abandon the same

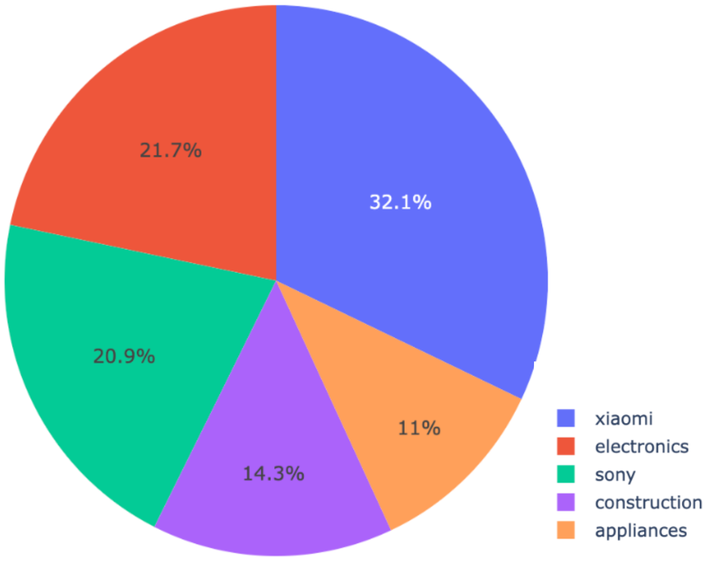


Fig 8.1 - Cart Abandonment Rate by Category

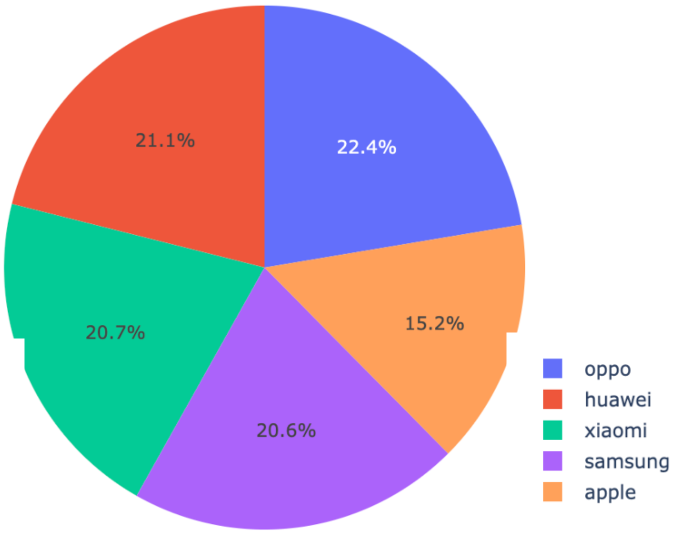
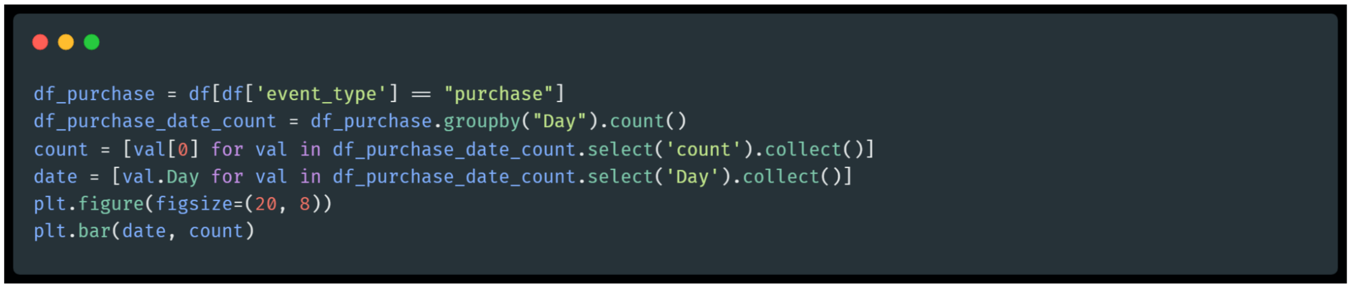


Fig-8.2 - Cart Abandonment Rate by Brands

OBJECTIVE 3 - EFFECT OF DAY-TIME WITH PURCHASE

Objective 3.a: Finding e-commerce Prime Time



```
df_purchase = df[df['event_type'] == "purchase"]
df_purchase_date_count = df_purchase.groupby("Day").count()
count = [val[0] for val in df_purchase_date_count.select('count').collect()]
date = [val.Day for val in df_purchase_date_count.select('Day').collect()]
plt.figure(figsize=(20, 8))
plt.bar(date, count)
```

Insights: We continue our analysis of the purchasing pattern of users in October. The bar plot in Fig-9, depicts that the sale of items is highest around the middle of the month which can be attributed to a couple of reasons like generally salaries get credited at the beginning of the month and people are figuring out expenses like rent, loans, etc.

Actions: Offering mid-month sale/discount from day 11 until 17 would act as a catalyst to increase sales. Further, this analysis can be scaled across all the months to identify peak traffic thereby giving users lucrative offers. It would be tremendously helpful to do such an analysis throughout the year to gauge a similar pattern in a monthly fashion.

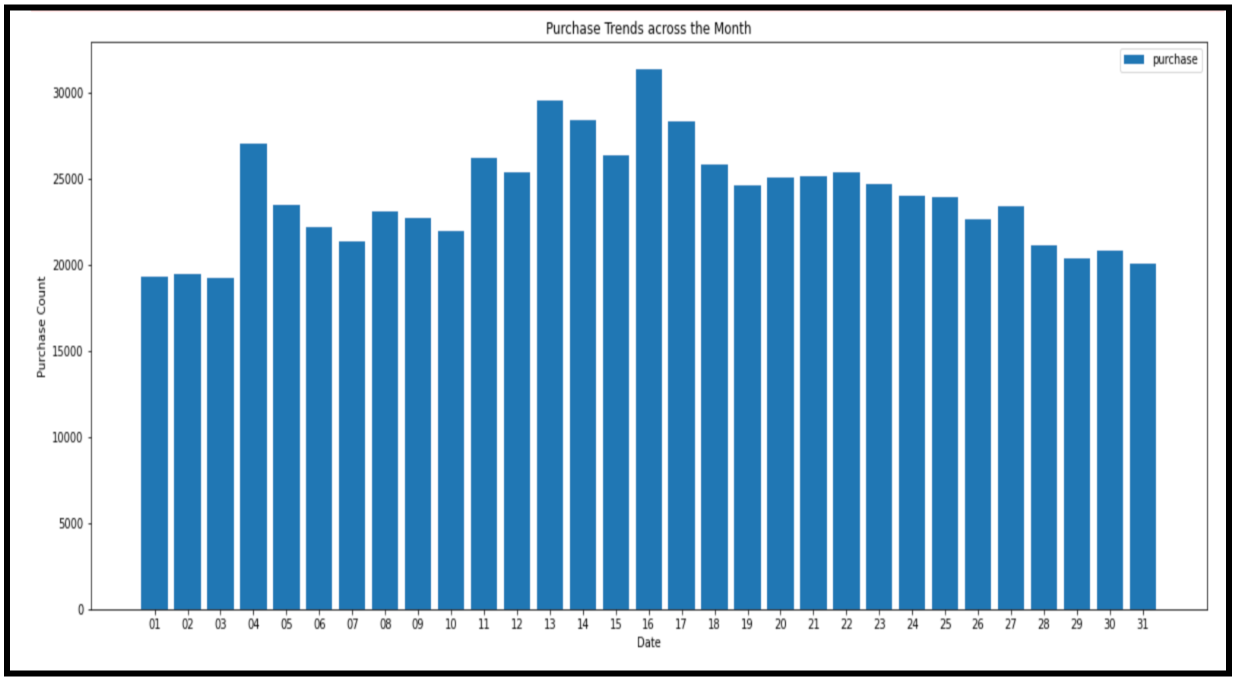


Fig: 9 - Purchase Trends in October.

<u>Objective 3.b: Finding e-commerce Prime Time</u>

<u>Insights:</u> We also focussed on analyzing shopping patterns per day to understand peak shopping time as shown in Fig 10. Almost 2M users have already accessed the e-commerce site by 11:00. We observed that it's generally around evening that users are actively browsing. There is also a high purchase in the morning denoted by the green bar, and it's significantly low through the tail end.

<u>Actions:</u> Time-bound flash sale/discount would be an actionable item to coerce users to complete their purchase journey.

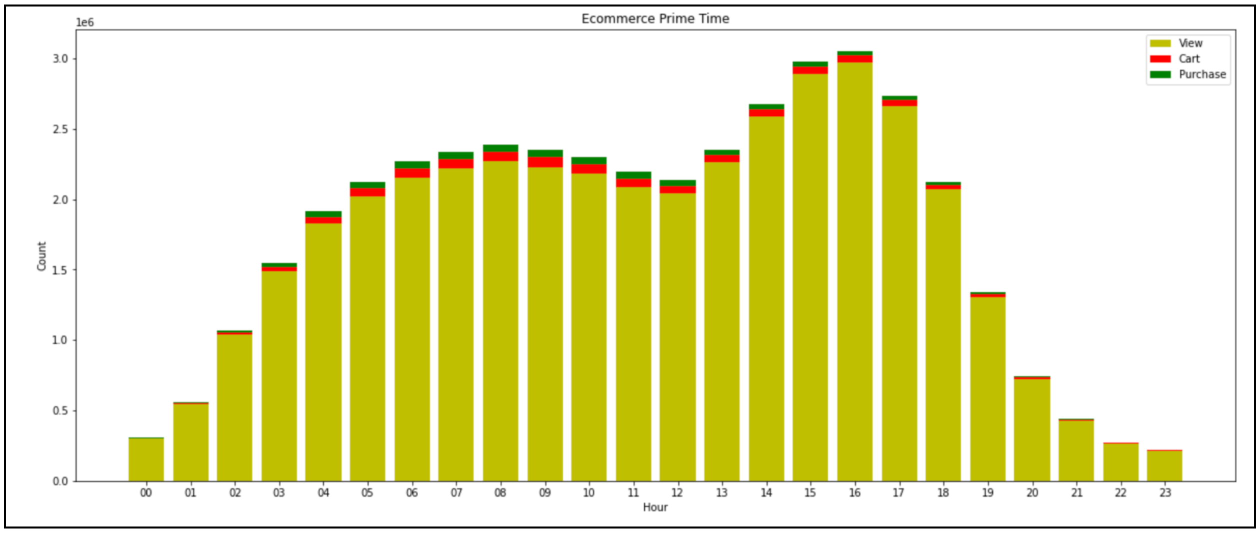


Fig-10 -Peak Traffic during different hours of the day.

OBJECTIVE 4: Build A Real-Time Machine Learning Classification Model To Predict Purchase

For Objective 4, we aimed to build a binary classification model that can predict whether there would be a purchase in a given user session. Through our analysis, we see that there is a high conversion ratio once the user adds a product to the cart. Features like brand, price, and how many times carted, have a direct impact on the purchasing trend of a customer. Through our predictive analysis pipeline, based on ordinal features like brand, category, product_type, and numerical features like price, week of the month, and activity count, we get the prediction result.

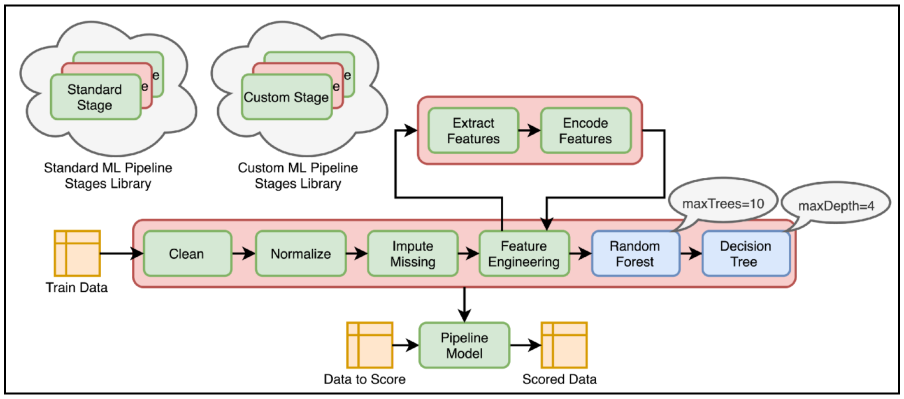
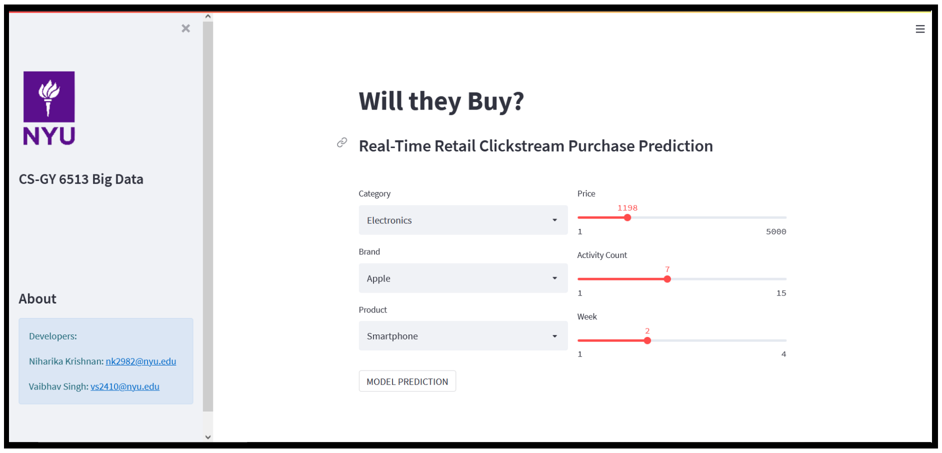


Fig-11 -ML pipeline for predictive modeling

PySpark ML was used for building the pipeline. We empirically tested a few classification models available in Pyspark like Logistic Regression, and XGBoost, and selected the random forest model which gave the highest accuracy of 78.42%. To simulate the inference pipeline deployment in the real world, we built an interactive web app using streamlit for demonstration.

DEMO USING STREAMLIT

Streamlit is a free, open-source, all-python framework that enables data scientists to quickly build interactive dashboards and machine learning web apps. For our use case, we've used Streamlit to demo a mockup of a real-time inference engine that can be plugged into an existing retail analytics solution that captures clickstream data.



NYU

CS-GY 6513 Big Data

About

Developers:

Niharika Krishnan: nk2982@nyu.edu

Vaibhav Singh: vs2410@nyu.edu

Will they Buy?

Real-Time Retail Clickstream Purchase Prediction

Category: Electronics

Brand: Apple

Product: Smartphone

Price: 1198 (1 – 5000)

Activity Count: 7 (1 – 15)

Week: 2 (1 – 4)

MODEL PREDICTION

CONCLUSION

Clickstream analysis is used by leading Retailers to make important business decisions using Big Data techniques. It also requires specific skills and resources necessary to capture, collect and analyze this information. As we saw through our project there are many benefits of such a detailed analysis such as optimizing user routes customers take to reach a page or to make a purchase. Deeper insights into consumer behavior like how visitors get to the website; what they do once there; how long they stay on a page; the number of page visits visitors make; and the number of unique and repeat visitors can help in the development of quantifiable metrics to boost revenue generation. Such analysis can also help in running narrowly-targeted marketing campaigns. In short, we observed that

"Tracking customer behavior in an online store is instrumental to offering a personalized customer experience and selling the right products in the right way"

FUTURE SCOPE

When solving any problems for Big Data, the first and most critical step is to actually understand what is in the data, which is difficult because of the size of the data. When we first started looking at the data, it was difficult to get a full picture quickly. Even a simple count or filter on such huge data using pandas failed. This helped in appreciating big data technologies in a practical scenario. There was also a paradigm shift in our understanding and implementation from centralized and functional programming to distributed and parallel computing. This project not only provided us with hands-on experience on a practical problem but also made us technically skilled to tackle such problems in real-world scenarios. This work can be extended to incorporate more components like the integration of databases and message queues, to ingest dynamic data in real-time. We also limited ourselves to pyspark, but modern state-of-the-art solutions like Dask, Fueger, and Ray could also be explored to enhance performance.

Github Link - https://github.com/niharikakrishnan/Retail-Clickstream-Analysis-and-Prediction