UNIVERSITÀ DI PISA

Distributed Data Analysis and Mining

eCommerce behaviour data from multi category store - Project Report

Davide Chen - 544795
Hiba El Rhell - 659866
Xinyi Gu - 584731
Xhenis Jaku - 639180
Yian Li - 667279

Department of Informatics
Data Science and Business Informatics
Academic year 2023/2024

Contents

1 Introduction & Dataset Description 1
1.1 Predictive Analysis of eCommerce Shopping Session Outcomes . . . 1
1.2 Dataset Overview . . . 1

2 Data Preparation & Understanding 2
2.1 Data cleaning and Preprocessing . . . 2
2.2 Stratified Sampling . . . 2
2.3 Working with Sampled Data: Data Overview and Cleaning . . . 3
2.3.1 Splitting Sampled Data into Two Pools . . . 3
2.4 Data Understanding . . . 5
2.4.1 Time Distribution . . . 5
2.4.2 Category Distribution and Purchase Insights . . . 6
2.4.3 Analysis of Time Spent on One Purchase . . . 6

3 Classification 7
3.1 Feature Extraction . . . 7
3.1.1 Correlation . . . 8
3.2 Random Forest Classifier . . . 8
3.2.1 Classification for balanced data . . . 8
3.2.2 Hyperparameter tuning . . . 8
3.2.3 Model evaluation . . . 9
3.2.4 Model visualization . . . 9
3.3 Logistic Regression . . . 10
3.3.1 Model building . . . 10
3.3.2 Model visualization . . . 10
3.4 Conclusions . . . 11

1 Introduction & Dataset Description

1.1. Predictive Analysis of eCommerce Shopping Session Outcomes

During the last years consumer behavior has changed and so have businesses. There has been an increasing number of transactions occurring in online retail [1], therefore understanding user interactions and predicting their shopping session outcomes has become crucial for businesses.

This project focuses on the analysis of eCommerce behavior data sourced from a large multi-category online store [2]. The dataset contains data for 7 months (from October 2019 to April 2020). The analysis is made on one month, March 2020, because it is a month without festivities, thus, it is less subject to bias.

1.2. Dataset Overview

Each row in the dataset represents a distinct event, showing relationships between products and users. The events are categorized into four types: "view", "cart" and "purchase". For the project, the "purchase" event is the most important because it represents a successful transaction. A single session can involve multiple purchase events, showing the diversity of user interactions within a given timeframe.

Event Types:	**View**: A user viewed a product. **Cart**: A user added a product to the shopping cart. **Purchase**: A user successfully completed a purchase.
Product information:	**Product ID** (product_id): Unique identifier for each product. **Category ID** (category_id): Product's category classification. **Category Code** (category_code): Taxonomy code for meaningful product categories. **Brand**: Downcased string representing the brand name. **Price** (price): Float value indicating the price of a product.
User information:	**User ID** (user_id): Permanent identifier for users. **User Session** (user_session): Temporary session ID, changed upon the user's return from a prolonged pause.
Time information:	**event_time**: The temporal aspect is captured through the timestamp, denoting when each event occurred in Coordinated Universal Time (UTC).

The steps of the data set analysis involve applying data understanding and preparation techniques, classification, and model evaluation to predict the outcome of shopping sessions. By employing machine learning methodologies, we aim to find patterns within the dataset that can contribute to the accurate prediction of user behavior, specifically predicting whether a shopping session will end up in a successful purchase.

2
Data Preparation & Understanding

The data was analyzed using `PySpark`, a powerful framework for large-scale data processing. Firstly all the necessary libraries are imported for data analysis and visualization, as well as components from the PySpark module. After creating a Spark session, the data of March month is loaded into a PySpark DataFrame. The original dataset consists of 56,341,241 rows, representing various user interactions with products on the online store, with columns such as `event_time`, `event_type`, `product_id`, `category_id`, `category_code`, `brand`, `price`, `user_id`, and `user_session`.

2.1. Data cleaning and Preprocessing

Null Value Analysis

Null values are checked in the DataFrame by counting the nulls in each column. The results indicate that the columns `category_code` and `brand` have a considerable number of null values, specifically 5,938,692 null values for `category_code` and 8,116,543 for `brand`. The heatmap (Figure 2.1) visually represents the distribution of missing values, providing insights into patterns and areas with high null counts.

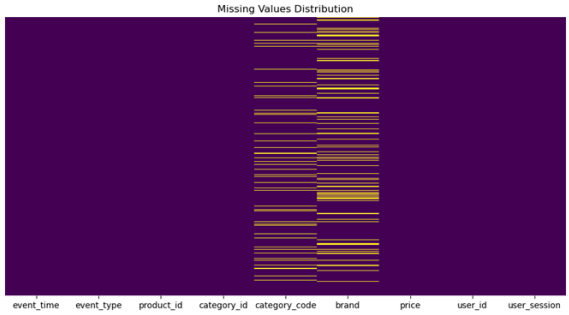

Figure 2.1: Heatmap visualizing missing values in original dataset

Handling Null Values

Handling missing brands: Inconsistent brands for each `product_id` are identified, then brands are filled based on the most common brand within each `product_id`.
Handling missing categories: missing values in the `category_code` are handled in a similar way to the brand column, ensuring consistency within each `category_id`.

Data Transformation: Splitting Category Codes

The `category_code` column is split into separate columns. This is done through a user-defined function, `split_cat`, applied to the DataFrame. Through this transformation the main category is extracted from the category_code column and then unnecessary columns (`category_id` and `brand`) are dropped.

Category Distribution Analysis

The distribution of products across different categories is done by counting the products in each category and calculating the respective percentages to have a category overview.

2.2. Stratified Sampling

A stratified sample is created in order to make an efficient analysis by selecting portions of the data set. The sampled data provides insight while maintaining a balanced representation across product categories.

Stratified Sampling (0.5%)

A stratified sampling is performed on the 'category_code' column, selecting a 0.5% fraction of the data for each category. User IDs with missing information are filtered out and the unique user IDs from the sampled data are then identified and saved to a 'unique_id' file for further analysis.
The unique user IDs are then joined with the processed dataset, keeping only the users with full information. The resulting data is saved to a CSV file named "sampled".

2.3. Working with Sampled Data: Data Overview and Cleaning

The sampled data is read, a schema is defined and a temporary view is created. The sampled data contains 6,112,428 rows with no null values and seven columns:

- `user_id`: int, 138,930 unique values.
- `event_time`: timestamp, 2,0936,48 unique values.
- `event_type`: str, 3 unique values.
- `product_id`: int, 128,312 unique values.
- `category_code`: str, 13 unique values.
- `price`: float, 46,066 unique values.
- `user_session`: str, 1,025,154 unique values.

Zero-price products

There are 4,630 entries with a price of 0.0. The categories with the highest occurrences of zero-priced products are 'electronics', 'appliances','apparel,' and 'construction'. Moreover, there are no entries with `event_type` 'purchase' and a price of 0.0.
Products with zero prices have varying average prices when priced. For example the product with `product_id` equal to 100145591 has some views with a price of 0.0 and others with a non-zero price (e.g., 1776.88).

Zero-priced products may require special consideration, as they seem to have variable prices and might be associated with specific categories. A theory could be that a zero-price product may be a new product for which price is not updated immediately and therefore cannot be purchased. This analysis helps identify whether there are valid products with occasional zero-price entries.

Event Type analysis

event_type	count	pct
purchase	941.869	1.869%
view	46.808.637	92.87%
cart	2.652.043	5.262%

Table 2.1: Table of the distribution of "event_type" in the original cleaned dataset (50.322.648 entries)

event_type	count	pct
purchase	138.883	2.272%
view	5.598.685	91.595%
cart	374.860	6.133%

Table 2.2: Table of the distribution of "event_type" in the original cleaned dataset (6.112.428 entries).

The event distribution in the sampled data is generally consistent with the original data as shown in Table 2.1 and Table 2.2. The proportion of purchase events is slightly higher in the sampled data.

2.3.1. Splitting Sampled Data into Two Pools

The sampled data is separated into two pools based on whether a user session successfully completes some purchases. The **Purchased pool** contains distinct user sessions with associated product IDs where the event type is a purchase. The **Non-Purchased pool** contains distinct user sessions with associated product IDs where the event type is not a purchase, which consists of the difference between the Total Unique Sessions and Sessions with Purchases. Both pools contains the columns `user_session, product_id, user_id, event_time, event_type, category_code, price` and both pools are stored in separate CSV files named `purchased` and `noPurchase`.
-**purchased**: it contains only the events related to the products been purchased that a user session has made. From the sampled data, unique combinations of user sessions and product IDs where purchases occurred are identified. Then, the sampled data is joined to get the trace of the complete events that each user session took to make one purchase.
-**noPurchase**: it contains the events that a user session has done for products that a user did or did not decide to buy. The dataset is simply the difference between `purchased` and `sampled`.
Through SQL queries the proportions of each event type (view, cart, purchase) are computed in the 'purchased' and 'noPurchase' datasets.

As shown in the Table 2.3 and Table 2.4 the proportion of event types varies significantly between the purchased and non-purchased datasets.

event_type	count	pct
purchase	138.883	20.437%
view	330.236	48.594%
cart	210.461	30.969%

Table 2.3: Proportion of each event type in purchased dataset, which has 679.580 entries

event_type	count	pct
view	5.268.449	96.974%
cart	164.399	3.026%

Table 2.4: Proportion of each event type in noPurchase dataset, which has 5.432.848 entries

2.4. Data Understanding

2.4.1. Time Distribution

The time distribution analysis provides insights into when users engage in shopping activities.
The hourly distribution of users activities on the site is analyzed by extracting hour from the "event_time" and counting events for each hour.
In the bar chart (2.2) it is visible that shopping is lowest in the early morning hours and gradually increases, peaking in the afternoon and evening.

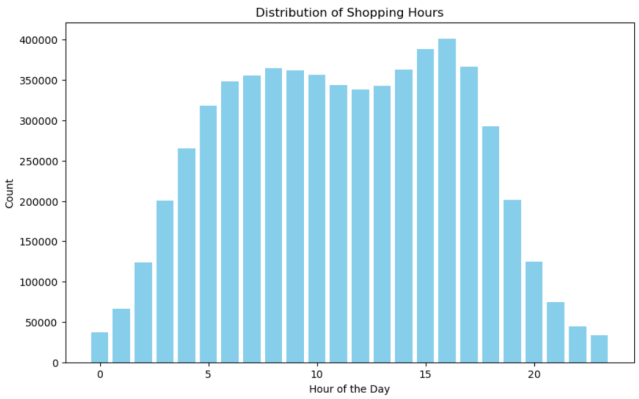

Figure 2.2: Bar chart represents the distribution of shopping activities based on hours of the day.

An analysis on which days of the week are most popular for user engagement has been done. The distribution of user activity based on the day of the week (combined across weeks) show that there are no major differences across the days, and that Monday is the day with less purchases (Figure 2.4).

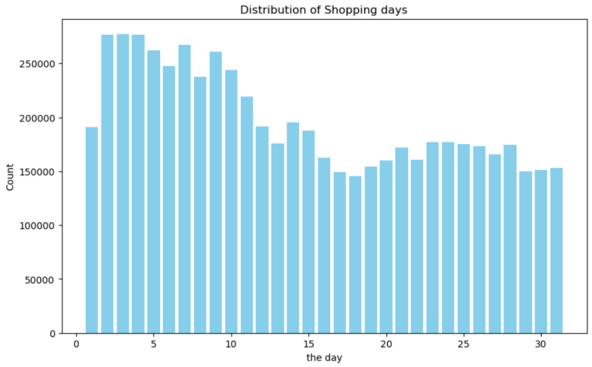

Figure 2.3: Distribution of shopping activities based on days of the month

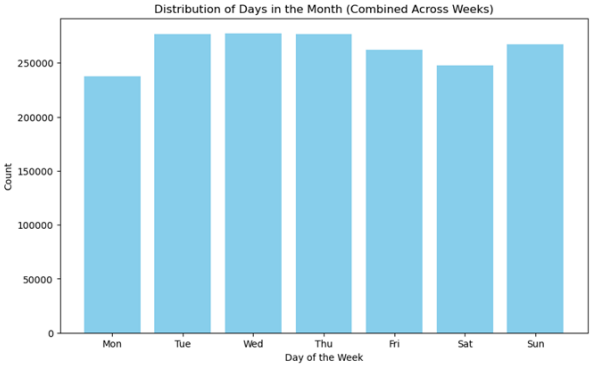

Figure 2.4: distribution of shopping activities based on weeks of the year

In Table 2.6 it is noticeable that there is higher engagement of the 10th week which corresponds to the dates from the 2nd of march until the 8th of march. The lowest engagement happens on the 9th week which is from the 24th of February until first march.

Week number	From Date	To Date
9	February 24, 2020	March, 1 , 2020
10	March, 2 , 2020	March, 8 , 2020
11	March, 9 , 2020	March, 15 , 2020
12	March, 16 , 2020	March, 22, 2020
13	March, 23 , 2020	March, 29 , 2020
14	March, 30 , 2020	April, 5 , 2020

Table 2.5: Days included in the analysis

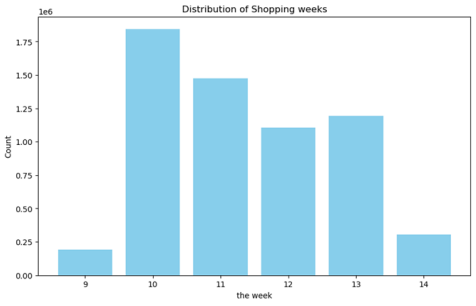

Table 2.6: distribution of shopping activities based on weeks of the year

2.4.2. Category Distribution and Purchase Insights

The following visualizations and analyses provide a comprehensive understanding of the category distribution in the sampled and purchased datasets, identification of repeated purchases, and insights into the time spent on individual purchases.
In the following pie charts it is shown that the "construction" category has the highest number of occurrences, followed by "electronics" and "appliances" in all category analyses.

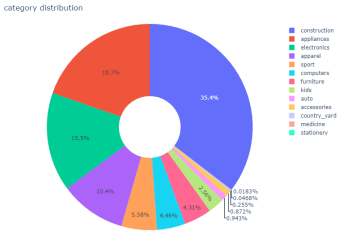

Figure 2.5: Overall Category Distribution on sampled dataset

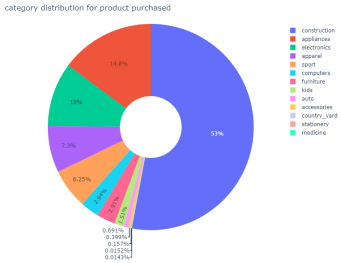

Figure 2.6: Category Distribution for Product Purchased

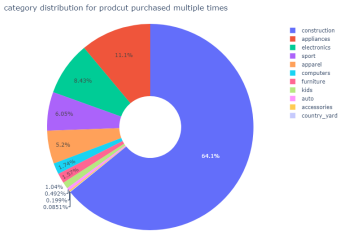

Figure 2.7: Category Distribution for Products Purchased Multiple Times

2.4.3. Analysis of Time Spent on One Purchase

The time spent on one purchase varies widely, ranging from 7 seconds to 684,915 seconds (approximately 190 hours). The mean time spent is around 315 seconds (approximately 5 minutes), but the standard deviation is high (3293.47 seconds), indicating significant variability in user behavior. The binned histogram (Figure 2.8) shows the distribution of time spent on one purchase, with a focus on values below 5000 seconds. It uses logarithmic bins to visualize the distribution across a broader range of values. The second histogram (Figure 2.9) focuses on values below 1000 seconds for a more detailed view.

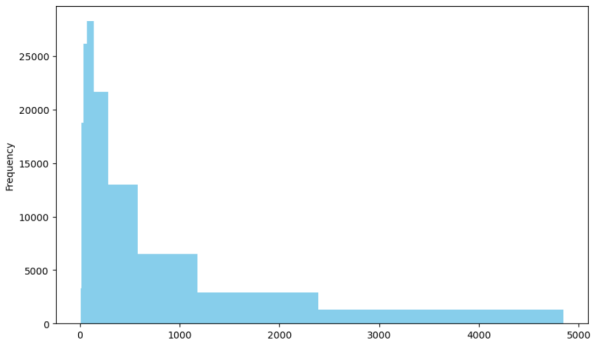

Figure 2.8: Histogram with logarithmic bins highlighting the distribution of time spent for purchases

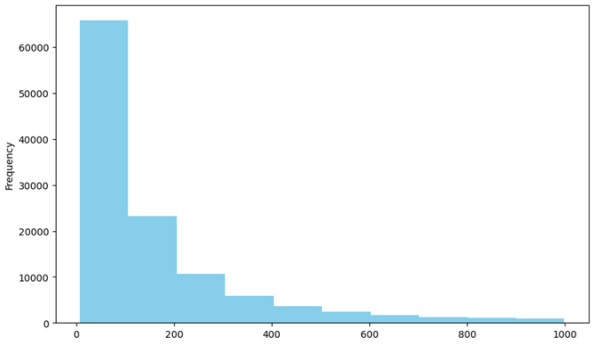

Figure 2.9: Histogram focused on shorter time intervals (<1000 seconds)

Bar plot provides information into the average time users spend on purchases within different categories and individual users.
The average time spent varies across categories is calculated (Figure 2.10), providing insights into user engagement with different product types.
From these analyzes, it has been observed that the wide range of time spent on one purchase suggests diverse user behaviors. The histograms indicate that a significant proportion of purchases involve relatively short durations. The average time spent varies across categories, providing insights into user engagement with different product types. Analyzing time spent by user_id helps identify individual preferences and behaviors.

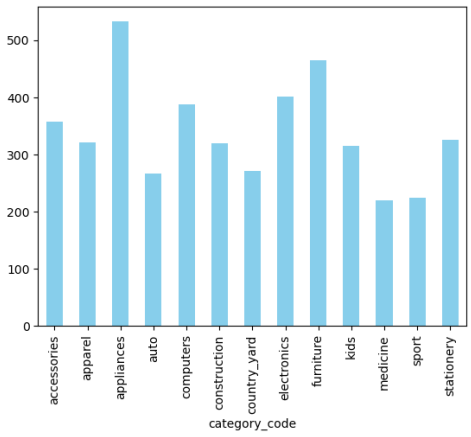

Figure 2.10: The bar chart illustrates the average time spent on one purchase for each category.

3
Classification

Setting up environment

In order to proceed with classification tasks, the two datasets **purchased** and **noPurchase** are loaded into Spark DataFrames. The schema is defined to ensure proper data types for each columns. The set up environment allows for exploration analysis and visualization using Spark and relevant libraries.

3.1. Feature Extraction

Valuable features are extracted from the both the data sets. These features are useful for understanding user behavior and predicting future purchases. Then we aggregated the data set, for purchased, each row is to record the events that a user session has performed to complete one purchase, while for "noPurchase", each row is to record all the events a user session has made for certain product.

Adding columns:

- **AddedToCart**: based on the presence of "cart" events
- **count_view**:
  - purchased: number of times of “view” before purchase
  - noPurchase: total number of times of view for one product during the session
- **timeSpent_sec**:
  - purchased: time spent for make one purchase
  - noPurchased: time spent for one product during the session
- **isPurchased**:
  - purchased: 1 assigned
  - noPurchase: 0 assigned
- **eventDay:** to store the day of the event

Then, sessions with only purchase (without view and/or cart) are removed as lack of user sessions' performance. And only the first purchase of a product is considered dropping the repeated purchase.

Union of the two datasets

The data sets are unbalanced, therefore to reach an optimal result during the classification tasks, it has been decided to balance them by randomly choosing rows from "noPurchase" dataset similar to rows in the "purchase" dataset and then combined them.

```
+-----------+-------+------------------+
|isPurchased|  count|               pct|
+-----------+-------+------------------+
|          1| 122537|0.0351360371979715|
|          0|3364965|0.9648639628020285|
+-----------+-------+------------------+
```

Figure 3.1: Distribution of classes in the unbalanced case

```
+-----------+------+------------------+
|isPurchased| count|               pct|
+-----------+------+------------------+
|          1|122537|0.4998613048763574|
|          0|122605|0.5001386951236426|
+-----------+------+------------------+
```

Figure 3.2: Distribution of classes in the balanced case

The datasets 'purchased' and 'noPurchase' are combined using `UnionALL` operation. and it is made sure that the columns of 'purchased' dataset (purchaseToCsv3) and the 'noPurchase' dataset (maxCountView) are the same. The final combined DataFrame "df_clf" is written to a CSV file. The aim of this is to create a balanced classification dataset for training machine learning models.

The dataset "df_clf" contains 245.142 records and 202.400 distinct user sessions. A correlation matrix is generated for the selected features: 'categoryIndex', 'price', 'AddedToCart', 'eventDay', 'count_view', 'timeSpent_sec', and 'isPurchased'.
The matrix is visualized using a heatmap (Figure 3.3). The correlation matrix provides insights into how each feature is correlated with others. Positive values indicate a positive correlation, while negative values indicate a negative correlation. The closer the value is to 1 or -1, the stronger the correlation. The heatmap shows a high correlation between the features "addedToCart" with "IsPurchased", while the correlation between "IsPurchased" and the rest of the features is close to 0. This could bring biased results during the classification tasks. Therefore, the "addedToCart" feature is dropped and additional features are added ("multiChoice", "PurBefore").

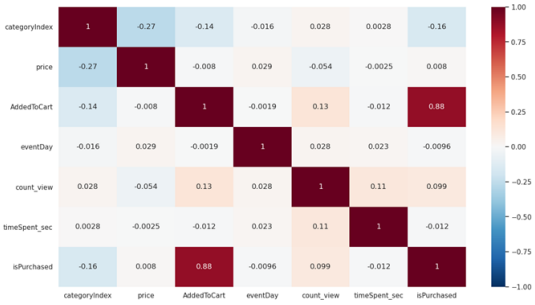

Figure 3.3: Heatmap of correlation matrix

3.1.1. Correlation

Additional Feature Extraction and Data Analysis

Two additional attributes "multiChoice" and "PurBefore" are added to the existing datasets ("df_clf", "pur_new", "noPur_new").

- "multiChoice" : indicates whether a user session has viewed multiple products in the same category during the session.
- "PurBefore" : represents whether a user has ever made a purchase on the site before.

A correlation analysis between features has been performed between "isPurchased" and "multiChoice" features in "df_clf".
Next, purchase history has been examined, and adjustments were made to the count of the number of purchases made by a user session up to the current event to ensure that the count starts from 0 for the first purchase.
The column 'event_time' is dropped and new datasets are written into csv files: df_clf_3, pur_new, noPur_new.
New correlation matrix analyses are done:

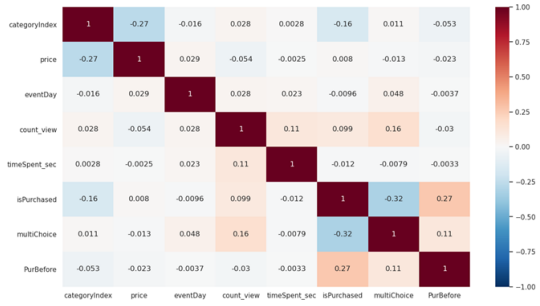

Figure 3.4: Heatmap of correlation matrix performed on balanced dataset "bal_df"

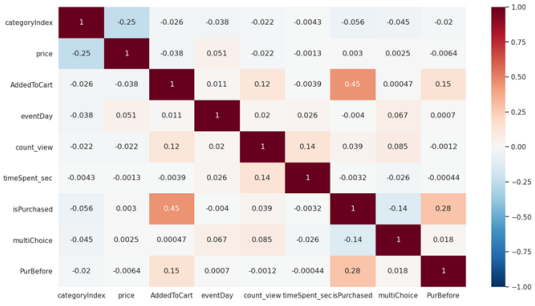

Figure 3.5: Heatmap of correlation matrix performed on unbalanced dataset "unbal_df"

3.2. Random Forest Classifier

3.2.1. Classification for balanced data

A Random Forest Classifier is chosen as predictive model. A first test with 5 trees and default maxDepth (also 5) is trained and evaluated. The area under the ROC curve for this model is 0.977, accuracy of 0.91 and precision of 0.86.

3.2.2. Hyperparameter tuning

In order to reach even better performance and find the optimal hyper-parameters, a gridSearch with cross-validation approach is used with different combinations of the trees' number and maximum depth. The

best model is selected based on the area under the ROC curve. The best hyper-parameters identified are 30 trees and a maximum depth of 15. Additional tests could have been done with higher values of trees and maxDepth, but the test resulted already in optimal values (as we will see in the following section), thus we preferred to keep the model less computationally expensive and not further complicate it.

3.2.3. Model evaluation

The performance of the best model was evaluated on the test set. The area under the ROC curve is 0.981, which indicates strong prediction.
The analysis shows that RandomForestClassifier is effective in predicting user purchases based on their online activities. In fact, his high ROC curve value indicates its ability to discriminate between positive and negative instances.

3.2.4. Model visualization

A confusion matrix is generating by comparing the predicted labels against the true labels on the test set, in order to assess the model's performance.

The confusion matrix provides a breakdown of predicted outcomes (rows) against actual outcomes (columns). The matrix has four components: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The model achieved an accuracy of approximately 92.04% and a precision of 89.31%. Accuracy represents the overall correctness of the model, while precision measures the accuracy of positive predictions. These metrics indicate the model's ability to make correct predictions.

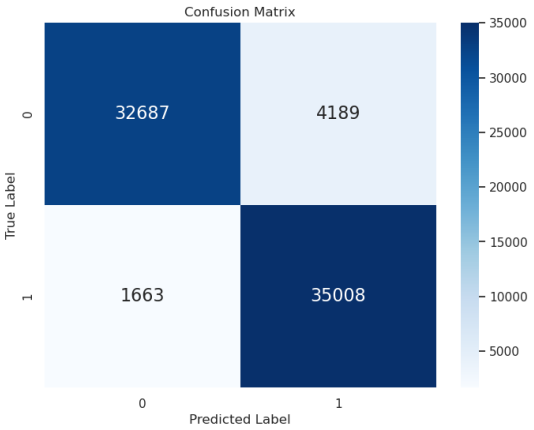

Figure 3.6: Confusion Matrix

Precision-Recall-F1 Score:

- **Class 0 (No Purchase)**: Precision of 0.97, Recall of 0.85 and F1 Score of 0.90.
- **Class 1 (Purchase)**:Precision of 0.86, Recall of 0.98 and F1 Score of 0.92.

A visualization of feature importance is created to understand the factors influencing the model's prediction. The top features contributing to the RFC model are identified: 'timeSpent_sec' and 'count_view', which initially were identified by the correlation matrix as not correlated features with the target class; "multiChoice" and "PurBefore", which had a moderate correlation, but they contributed less for the prediction model.

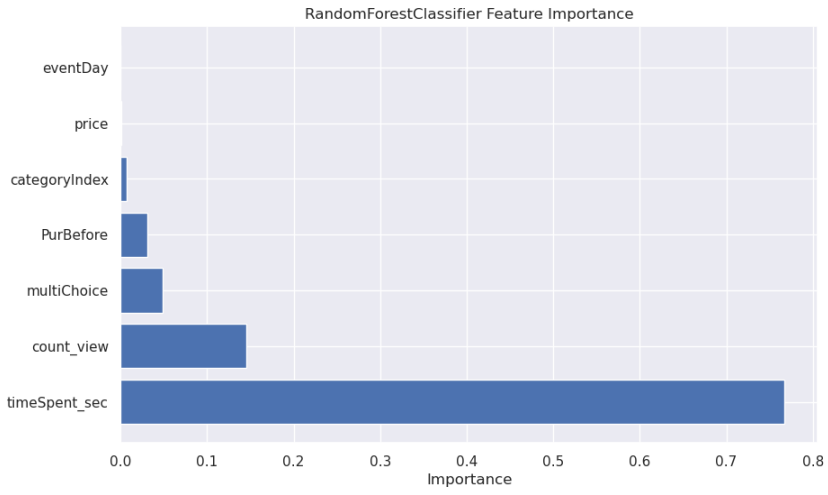

Figure 3.7: Feature Importance

3.3. Logistic Regression

The goal is to understand user behaviour specifically focusing on the time spent by users before making a purchase, using the extracted features.

Exploratory Data Analysis on shoppers who viewed/added items to cart but did not make a purchase
The most popular category for items in the cart but not purchased is "construction" followed by "appliances" and "electronics".
Summary statistics:

- The average price of items added to the cart but not purchased is $297.36.
- The average time spent by users who added to the cart but did not purchase is 522.93 seconds.
- The total number of views for users who did not make a purchase is 192.605
- There are 39.637 users who added to cart and then purchased.

Furthermore, potential outliers are identified in "price", "timeSpent_sec" columns using the Interquartile Range (IQR) anomaly detection method.

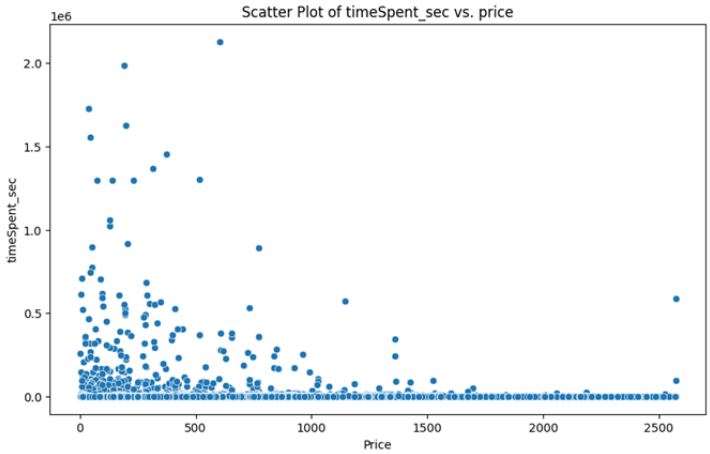

Figure 3.8: Scatter plot on "price" and "timeSpent_sec"

The analysis shows that logistic Regression Classifier although is best for binary classification has performed slight worse than the random forest for the task of predicting user purchases based on their on-line activities. This effect is based the removal of one feature, on a preliminary step noticed only very high correlation between “AddedToCart” and our target variable.

3.3.1. Model building

We defined a logistic regression with default parameters. Then created a pipeline that includes categorical features indexing, vector assembling, feature scaling (StandardScaler) and logistic regression stages. The dataset is split into 80-10-10 as train, valid and Test set respectively. The hyperparameter tuning is done using cross validation, which results in having: **RegParam** (regularization parameter) equal to 0.01 and **ElasticNetParam** (Elastic Net mixing parameter) equal to 0.5.

3.3.2. Model visualization

The ROC curve value of almost 80% indicates its ability to discriminate between positive and negative instances (Figure 3.9)
The model achieved an accuracy of approximately 72% and a precision of 75%. Overall, these metrics indicate the model’s ability to make correct predictions.

Precision-Recall-F1 Score:

- Class 0 (No Purchase): Precision of 0.70, Recall of 0.78 and F1 Score of 0.74.
- Class 1 (Purchase):Precision of 0.76, Recall of 0.66 and F1 Score of 0.71.

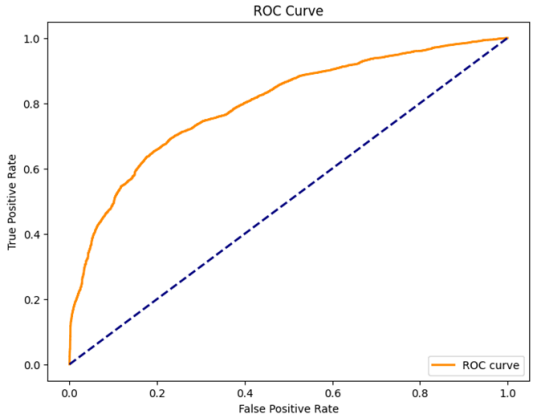

Figure 3.9: ROC curve

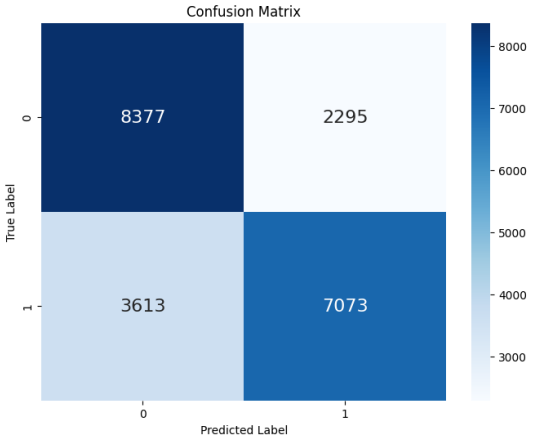

Figure 3.10: Confusion Matrix

Model evaluation

The model has been tuned and evaluated on both the validation and test set, the AUC values suggest good discriminatory ability. The chosen hyper parameters performed better on the validation set. The logistic regression model performs well with high AUC and accuracy.

3.4. Conclusions

The analysis conducted on the eCommerce dataset, with the aim to predict the shopping session outcomes resulted in two main classification tasks. One done with Random Forest Classifier and the other performed with Logistic Regression. Based on the analyzes done, Random Forest Classifier performed better. If we compare the results we get:

	Random Forest Classifier	**Logistic Regression Classifier**
AUC on the Test Set	0.981	0.7900
Best Hyperparameters	Max Depth: 15, Num Trees: 30	RegParam: 0.01, ElasticNetParam: 0.5
Accuracy	92.04%	72.3%
Precision	89.31%	75.5%

The AUC values for the Random Forest classifier are significantly higher than those for the Logistic Regression classifier on both the validation and test sets. A higher AUC indicates better discrimination between positive and negative classes.
The accuracy and precision of the Random Forest classifier are also higher, suggesting better overall classification performance and a better balance between true positive predictions and total positive predictions.
Additionally Random Forest provides more information about features' importance, which could be used for further analysis in the future.
It would have been useful to see the difference in performance compared between the balanced and unbalanced datasets. Unfortunately, given the high size of the dataset and the limited resources made available by "SoBigData" and "Colab", the task could not be completed properly.

Bibliography

[1] ESW. (2022, February 15). How Ecommerce has Changed Consumer Behaviour. ESW. https://esw.com/blog/how-ecommerce-has-changed-consumer-behaviour/

[2] eCommerce behavior data from multi category store. (n.d.). Retrieved December 4, 2023, from https://www.kaggle.com /datasets/mkechinov/ecommerce-behavior-data-from-multi-category-store