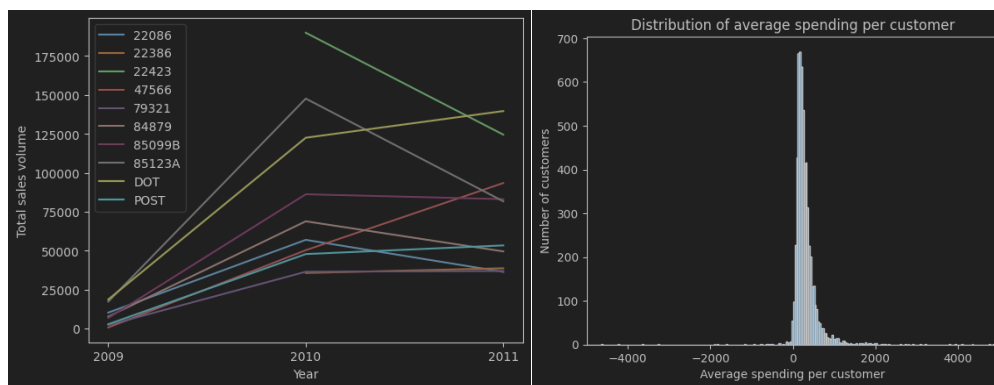


## Q4. Recommendation and Business Analysis

### Visualization and business insights

#### 1. Total sales volume per product per year

This is the annual sales trend of the top ten products in terms of sales, and it can be seen that most of the best-selling products have experienced a decline after an upward period. It is worth noting that product 47566 has maintained an upward trend.

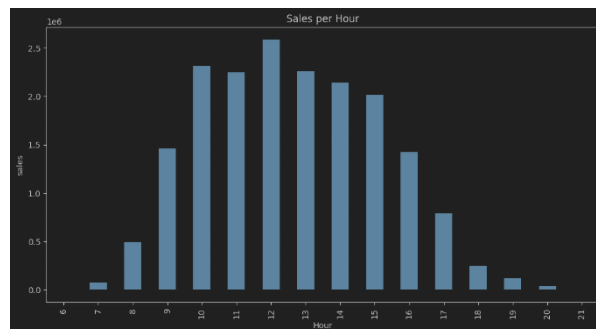


#### 2. The average spending per customer and the overall distribution

This chart shows the average consumption of each customer each time they shop. It can be seen that most people spend less than 1000 yuan per purchase, and only a very small number of people spend more than 2000 yuan per purchase.

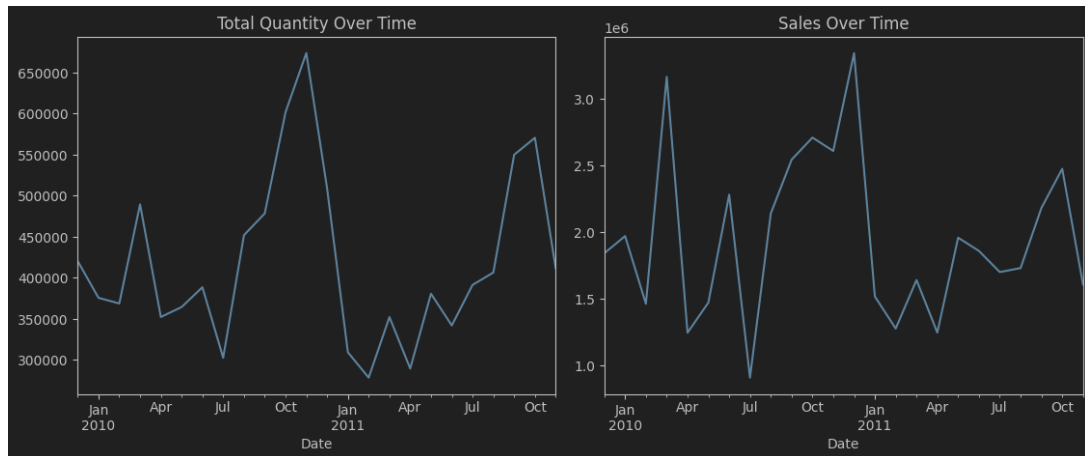
#### 3. Sales pre hour

Through this chart, we can intuitively understand the high and low turnover of each time period every day. We can see that the turnover is very low in the morning and evening, and the best time is around noon. This may suggest that the operating hours of the store can be adjusted according to the turnover of each time period.



#### 4. Time series chart of transaction quantity

It can be seen that sales vary greatly over time, with the peak season usually around April and October. By understanding the demand for goods in each period, some scheduling problems can be solved

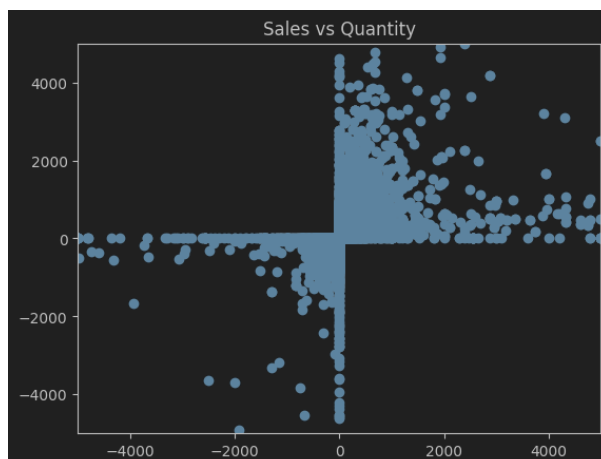


#### 5. Time series chart of sales revenue

Similarly, we can also know when we will have sufficient funds in our hands, and at certain times our weekly transfer of funds slows down, which helps us better allocate funds to achieve maximum benefits

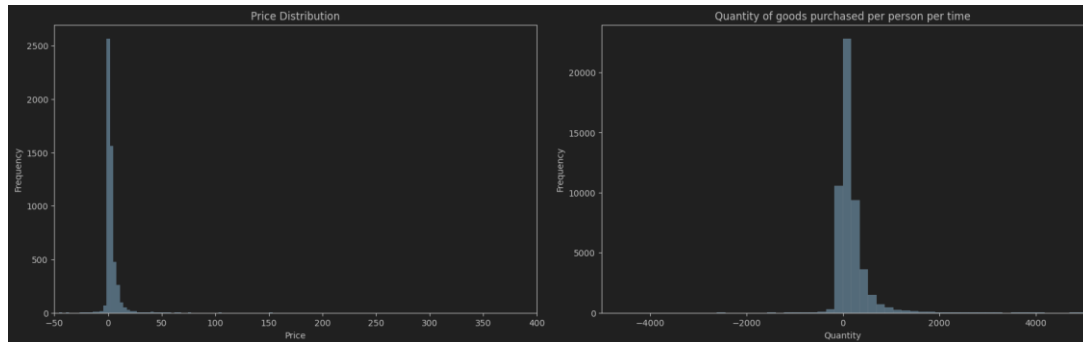
#### 6. Scatter plot of sales revenue and transaction quantity

We can find that most products are either high sales with low sales volume, or low sales with high sales volume. Which products with larger areas can be found in the picture, which are profitable products



## 7. Distribution of unit price of goods

The unit price of a product often does not exceed 50, and the vast majority of products have very low unit prices.

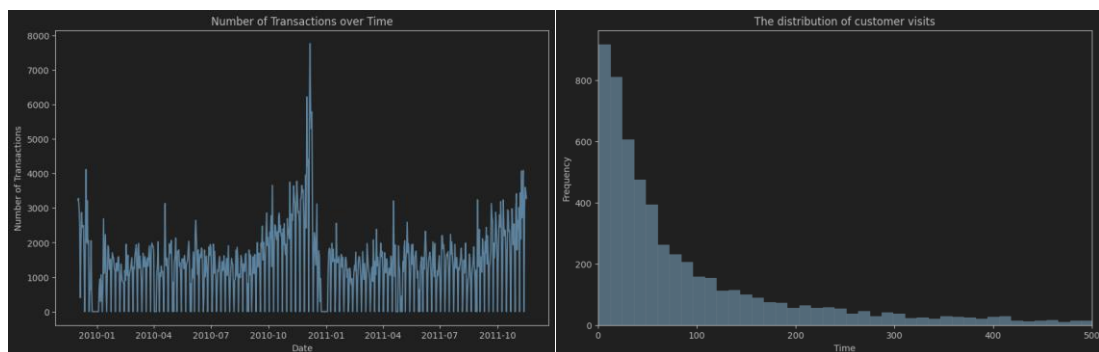


## 8. Quantity of goods purchased per person per time

This picture shows the quantity of goods purchased by each person each time. Most people will purchase less than 100 items, and the quantity here is the product of each category by their quantity.

## 9. Changes in daily transactions flow

It can be seen that the daily order quantity fluctuates greatly, but there seems to be a cyclical trend every year. It can be seen that the order quantity from October to January is larger than other months.



## 10. The distribution of customer visits

According to this chart, it can be seen that the more frequent the visits, the fewer customers there are. However, the decline after 200 visits becomes slow, which confirms that the store has many

loyal customers as well as many repeat customers.

## Association rule analysis

### 1. Apriori algorithm

The Apriori algorithm is one of the most influential algorithms for mining frequent itemsets of Boolean association rules.

Apriori uses an iterative method called layer by layer search, where  $k$ -itemsets are used to explore  $(k+1)$ -itemsets. Firstly, identify the set of frequent 1-itemsets. It is called set  $L_1$ .  $L_1$  is used to find the set  $L_2$  of frequent 2-itemsets, while  $L_2$  is used to find  $L_3$ , and so on until frequent  $k$ -itemsets cannot be found.

In order to improve the efficiency of generating frequent itemsets layer by layer, an important property called Apriori property is used to compress the search space

Apriori property: If itemset  $X$  is a frequent itemset, then all its non empty subsets are frequent itemsets

The process of finding  $L_k$  using  $L_{(k-1)}$  consists of connections and pruning:

(1) Connection step: To find  $L_k$ , connect  $L_{(k-1)}$  with oneself to generate a set of candidate  $k$ -itemsets. The set of candidate itemsets is denoted as  $C_k$ . Let  $l_1$  and  $l_2$  be the itemsets in  $L_{(k-1)}$ . The symbol  $l_i[j]$  represents the  $j$ -th item of  $l_i$ . For convenience, it is assumed that the items in a transaction or itemset are sorted in dictionary order. Execute connection  $L_{(k-1)} \bowtie L_{(k-1)}$ , where the elements of  $L_{(k-1)}$  are concatenable. The result of connecting  $L_1$  itemsets and  $L_2$  itemsets is itemset  $l_1[1] l_2[2] \dots l_1[k-1] l_2[k-1]$

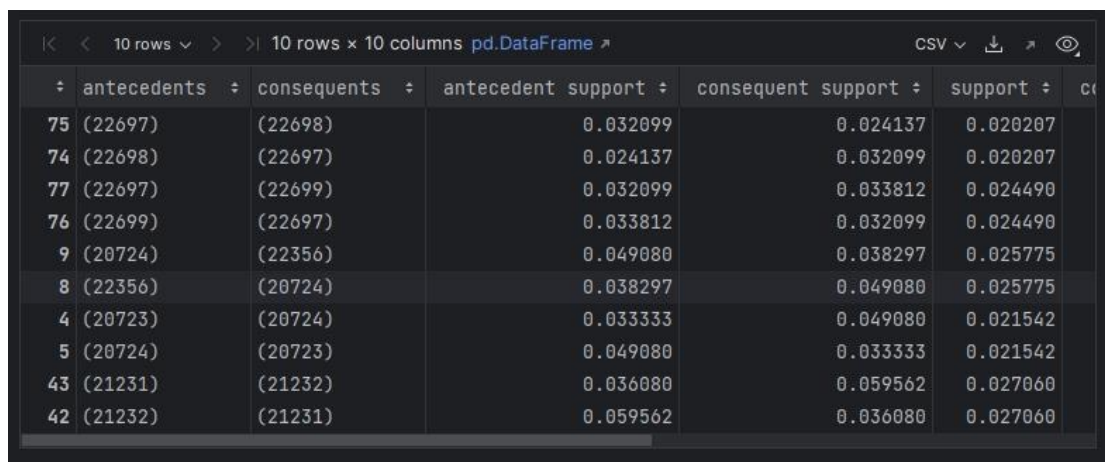
(2) Pruning step:  $C_k$  is a superset of  $L_k$ ; That is, its members can be or not frequency sets, but all frequency sets  $k$ -itemsets are included in  $C_k$ . Scan the database to determine the count of each candidate in  $C_k$ , thereby determining  $L_k$ . However,  $C_k$  may be very large, which would require a significant amount of computation. To compress  $C_k$ , the Apriori property can be used in the following way: no  $(k-1)$ -itemset of any non frequency set can be a subset of frequent  $k$ -itemsets. Therefore, if the  $(k-1)$ -subset of a candidate  $k$ -itemset is not in  $L_{k-1}$ , then the candidate cannot be frequent and can be deleted from  $C_k$ .

## 2. Implementation

In this part, we implement Apriori algorithm with mlxtend. Converting data types to TransactionEncoder is similar to single hot encoding, where each value is converted to a unique bool value. Then calculate frequent itemsets and generate association rules. We select rules based on lift:

$$\text{Lift} = \frac{\text{Support}(X \cap Y)}{\text{Support}(X) * \text{Support}(Y)}$$

Finally make the recommendation:



	antecedents	consequents	antecedent support	consequent support	support	confidence
75	(22697)	(22698)	0.032099	0.024137	0.020207	0.628571
74	(22698)	(22697)	0.024137	0.032099	0.020207	0.628571
77	(22697)	(22699)	0.032099	0.033812	0.024490	0.763438
76	(22699)	(22697)	0.033812	0.032099	0.024490	0.763438
9	(20724)	(22356)	0.049080	0.038297	0.025775	0.523148
8	(22356)	(20724)	0.038297	0.049080	0.025775	0.523148
4	(20723)	(20724)	0.033333	0.049080	0.021542	0.645833
5	(20724)	(20723)	0.049080	0.033333	0.021542	0.645833
43	(21231)	(21232)	0.036080	0.059562	0.027060	0.750000
42	(21232)	(21231)	0.059562	0.036080	0.027060	0.750000