SD210-Machine Learning

Data Challenge

ZHANG Bolong
WANG Yuqing
YAN Ming
ZHU Fangda

Data Challenge

SD201-Machine Learning

by

ZHANG Bolong
WANG Yuqing
YAN Ming
ZHU Fangda

Report for the Comparative Law

Contents

1 Introduction **1**

2 Analysis of features **3**

2.1 Feature analysis for discrete data . . . 3

2.1.1 Gender . . . 3

2.1.2 Sytle of shoes . . . 4

2.1.3 Upper material of shoes . . . 4

2.1.4 ISO code . . . 5

2.1.5 Product Type . . . 6

2.2 Feature analysis for continuous data . . . 7

2.2.1 Number of items in the shopping cart . . . 7

2.2.2 The price of the shoes . . . 7

2.2.3 Time gap between two purchases . . . 8

2.2.4 Customer's age . . . 9

2.3 The relation between two features . . . 9

3 Computation **13**

4 Conclusion **15**

1 Introduction

In electronic commerce, a large number of product return is untolerable and it will bring negative impacts on margin. An E-commerce company that sells shoes has a high return rate of more than 20%. Our work is to help this company to well understand this phenomenon, by analysing its database. Provided by the company, we have access to its database of orders placed between October 2011 and October 2015, its product feedback data, and its customer and product databases.

The main goals of our work include:

1. Identify conditions that favor product return
2. Build a return forecast template for each product from a shopping cart.

To achieve goal 2, the process of feature analysis(namely goal 1) and the reconstruction for training set are indispensable. In order to reconstruct a fine training set, we need to extract the features that has a large impact on return rates, and also need to cast away the features that are indifferent to return rates. We will illustrate the process in detail later. At last, after countless trying and searching, our predict accuracy reaches 0.7120, which we think it is a successful and joyful result.

2
Analysis of features

2.1. Feature analysis for discrete data

In this section, What we want to do is, according to the return rate of each feature that we have chosen, finding those options that influence most in a feature.

For each feature we have chosen, the structure is as follows:

First, give the total number of sample in the feature and also give the number of each label in this feature;

Second, give the bar diagram of return rates;

Third, analyse the result and give a conclution.

Our return rate is calculated by the numbers of return product with a specific label in a feature divided by the total number sold with this label in this feature.For example, in the feature "Gender", the total number is 790413, which includes "Femme", "Homme", "NC". While in this scope, the number of samples attached with label "Femme" is 646050, which includes the product returned and not returned. While in this 646050 samples, we calculate the number of product returned. Then this number is divided by 646050. In this way, we get the return rate.

What's more, what we need to pay attention to is those returned products attached to a specific label with a large base of samples. If the number with a specific label is less than 1% of the total number(ignore NAN when counting), we can ignore this part because this part of sample is too small. We only focus on those data with a large base of samples.

In the experiment, we list more than 10 features to evaluate its importance, limited by the time and text, we decide to only present those features which give us an interest ing result or which tell obvious differences on return rates.

2.1.1. Gender

```
X_train_final['Gender'].count()
```

Result:790413

```
X_train_final['Gender'].value_counts()
```

Items	Femme	Homme	NC
Values	646050	144361	2

According to these data shown in figure 2.1, we found a interesting phenomenon: women shop online more frequently than man(the number of woman client is 646050, while the number of man client is 144361). While for gender unknown, there are only 2 samples, which we will obviously ignore. Clearly presented in the bar diagram, the return rate by women clients reaches 22%, which is 6% higher than that by men clients. Although the difference 6% might seem not convincing enough, it can reflect something. We would like to say that women tends to return more products they buy online than men.

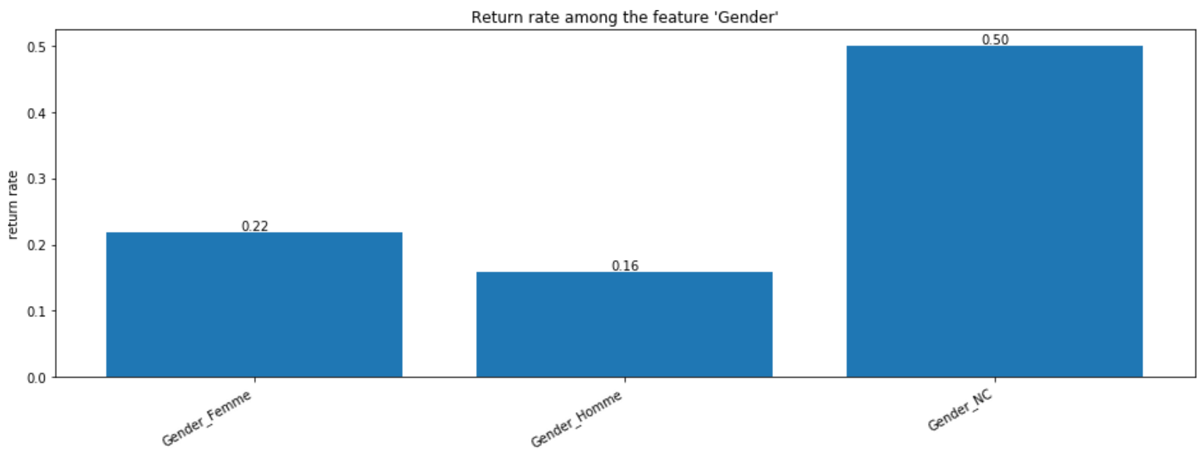

Figure 2.1: Feature 'Gender'

2.1.2. Sytle of shoes

```
X_train_final['UniverseLabel'].count()
```

Result:689670

```
X_train_final['Gender'].value_counts()
```

Result:

Items	Détente	Ville	ND	Sport	Green	Luxe	Confort	Soirée
Values	350100	209264	92508	13000	11210	7738	5167	683

UniverseLabel here stands for the style of shoes. Its return rate is shown in figure **??**.

First we need to ignore those samples whose number is less than 1% of the total number. Thus style confort and style soirée discarded. We only concern about the styles: "détente","ville", "ND", "sport", "green" and "luxe". Their return rates are shown in the diagram above. We can see for shoe styles "détent", "ND" "green", their return rates are around 20%. There is no big difference among them. their return rate is reasonable since the whole return rate is around 20%. We set 20% as the normal level of return rate.

We find something interesting for shoe styles "luxe", "ville" and "sport". The return rate of shoe style "luxe" and "ville", which both reach 27%, are a bit higher than normal level. While the style "sport" has the lowest return rate of 16%, which is lower than normal return rate. The highest return rate is 11% larger than the lowest. So we can say that those shoes with style "luxe" or style "ville" are more likely to be returned than the shoes with style "sport".

2.1.3. Upper material of shoes

```
X_train_final['UpperMaterialLabel'].count()
```

Result: 82114

```
X_train_final['UpperMaterialLabel'].describe()
```

Result:

Item	Value	Item	Value	Item	Value
Cuir	29774	Nubuck	21968	Textile	8542
Cuir/Textile	6338	Synthétique	5748	Cuir verni	5237
Cuir/Synthétique	4110	Caoutchouc	386	Fourrure véritable	11

First of all, we need to ignore the label "Caoutchouc" and "Fourrure véritable", which are less than 1% of the total number 82114. The return rates corresponding to these materials are shown in figure 2.3.

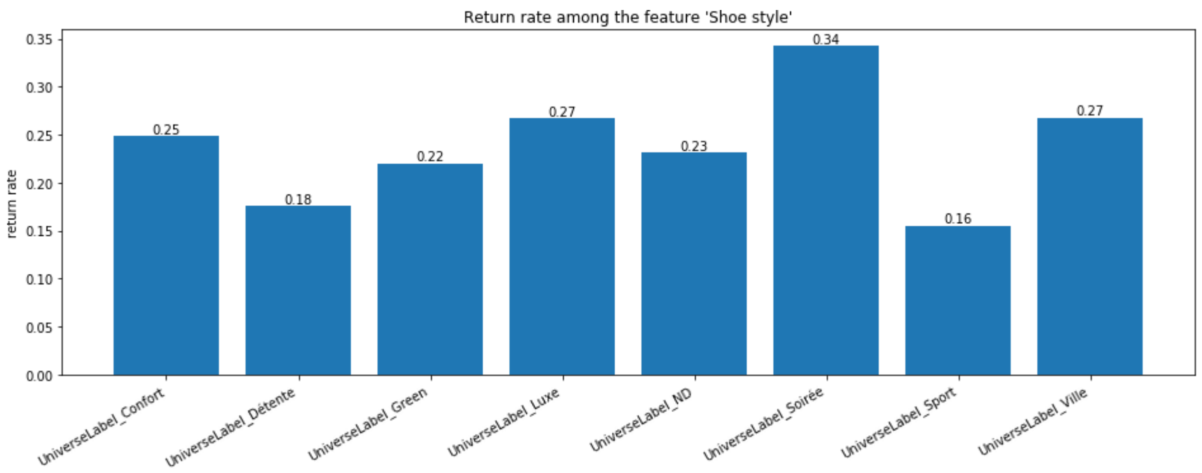

Figure 2.2: Feature ’UniverseLabel’

Among all the labels in upper material, we find that the material "cuir verni" has the highest return rate of 31%, while the lowest(apart from "Fourrure") return rate 16% lies on the material "Cuir/Synthétique". There exists a huge gap of 15% between the highest and the lowest. We can conclude that product of cuir verni has a high possibility (31%) to be returned. Which is explainable. For the material "cuir verni", it is easy to get dirty and maybe it will loss color while delivery. It has a high chance to be returned in the situation mentioned above. While for the material cuir/Synthétique, it does not like "cuir verni" so glossy. It is more durable and it stains difficultly.So it has a low return rate. Our suggestion for the producer: If the product is in the material of "cuir verni", be careful in the delivery, because this type of material is easy to get dirty or loss color, which will lead to product-returns.

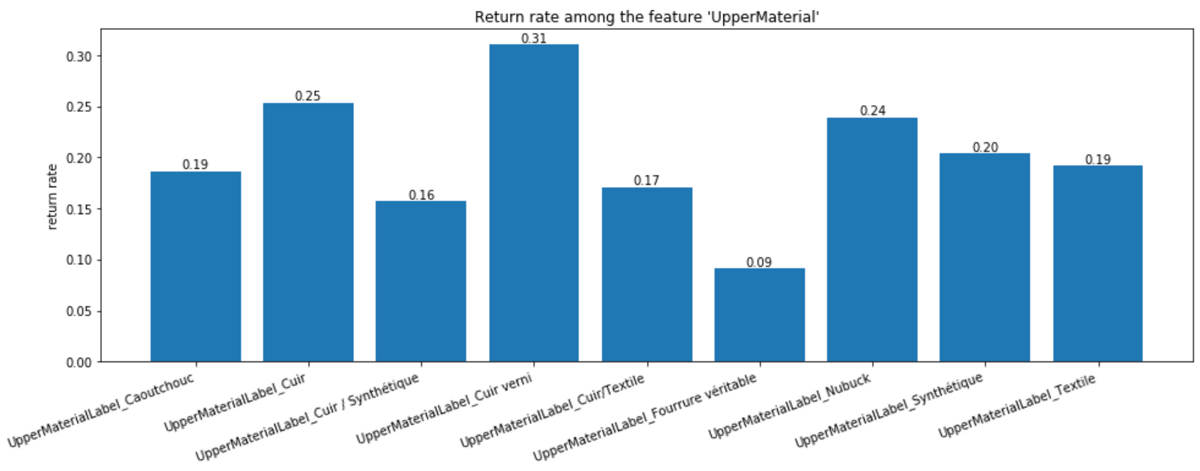

Figure 2.3: Feature ’UpperMaterialLabel’

2.1.4. ISO code

```
X_train_final['IsoCode'].count()
```

Result: 1067290

```
X_train_final['IsoCode'].value_counts()
```

Result:

After ignoring those small samples (number less than 10673), there rest Iso code "FR","DE", "BE", "NL", "GB", "ES", "IT". Their correspond return rates are shown in figure 2.4

Item	Value	Item	Value	Item	Value	Item	Value	Item	Value
FR	882181	DE	37600	BE	28849	NL	28730	GB	26125
ES	22582	IT	17714	PL	9163	DK	5957	SE	4733
LU	1376	AT	548	FI	456	GR	294	IE	237
PT	208	RO	135	CZ	127	SK	53	SI	47
HU	37	LT	34	EE	30	BG	29	LV	22

Since all the data come from a French shoes selling website, most customer are French as shown above in the table (number of 882181 out of 1067290). Let's deem the return rate by French customers as the normal return rate(20%). While, We get some interesting results according to the bar graph. It seems that German(Iso code DE) clients have less confidence in French products. The return rate by German clients reaches an amazing summit(44%), which is more than twice of that by French clients. While, the return rates by Belgium clients(Iso code BE), British clients(Iso code GB), Spanish clients(Iso code ES) are respectively 22%, 25%, 17%, which are similar to the return rate by French clients. The return rates by Dutch clients(Iso code NL) is 29%, which is a half higher than the normal return rate 20%. This point need also to be concerned. Furthermore, Italien clients have the lowest return rate 12%, which is almost half of the normal return rate 20%. To conclude, the feature Iso code influence a lot on return rates. German clients and Dutch clients return more often products. While the italien tend to return less products.

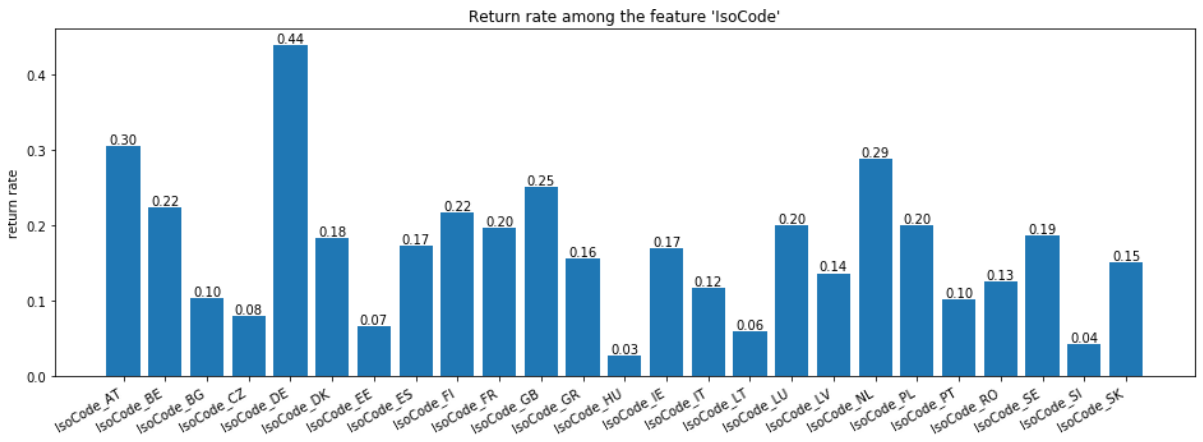

Figure 2.4: Feature 'IsoCode'

2.1.5. Product Type

```
X_train_final['ProductType'].count()
```

Result: 687194

```
X_train_final['ProductType'].value_counts()
```

Result:

Item	Value	Item	Value	Item	Value
Baskets	181070	Sandales et nu-pieds	102918	Bottines et boots	93170
Ballerines	53163	Escarpines	48495	Chaussures à lacets	40245
Chaussures de sport	39761	Bottes	31906	Tongs	28287
Mocassins	17830	Sacs à main	9390	Mules et sabots	9330
Chaussons	7635	Chaussures à scratch	5752	Espadrilles	5553
Produits d'entretien	4020	Petite Maroquinerie	1978	Semelles	1317
Sacs pochettes	1036	Sacs à dos	891	Sacs homme	761
Sacs de sport	601	Lacets	529	Bagages	482

Apart from those small samples, ProductType “Basket”, “Sandales et nu-pieds”, “Bottines et boots”, “Ballerine”, “Escarpins”, “Chaussure à lacets”, “Chaussure de sport”, “Bottes”, “Tongs”, “Mocassins”, “Sac à main”, “Mules et Sabots”, “Chaussons” leave. Their corresponding return rates are shown in figure 2.5

The return rate for different product types varies from 11%("Chaussons") to 35%("Escarpins"). we see that there exists a huge rate difference between these two types. While, to classify, "Chaussons"(11%) and "Tongs"(13%) have a less return rate; "Escarpins"(35%) and "Bottes"(30%) have a relatively high return rate; the other types have a medium return rate which is around 20%. The peaks and bottoms of return rates are comprehensible. For peaks("Escarpins" and "Bottes"), because "Escarpins" and "Bottes" are the type of shoes which size is not easy to choose. If the size of shoes is not perfectly suitable for customer’s feet, It will be uncomfortable to wear them. This phenonenon is quite often, especially for the type of shoes "Escarpins" and "Bottes", one of which has high heel and the other has long shoe tubes. On the contrary, the type "Tong" and "Chaussons" have a low return rate, which is also explainable. Because people always wear "Tong" and "Chaussons" at home. These type of shoes are not the shoes for going out. So people care more about comfort rather than good-looking. Thus, people have less expectation on "Tong" and "Chaussons".

To sum up, the shoe type "Escarpins" and "Bottes", which need careful size choosing, are more likely to be returned; while the shoe type home-wearing like "Tong" and "Chaussons" are less likely to be returned.

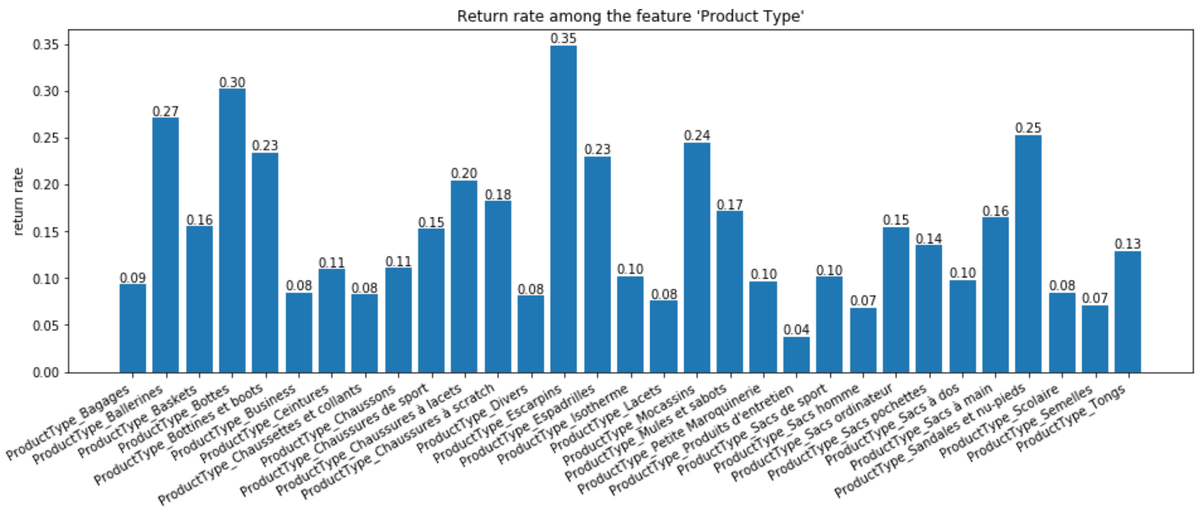

Figure 2.5: Feature ’ProductType’

2.2. Feature analysis for continuous data

2.2.1. Number of items in the shopping cart

```
X_train_final['TotalLineItems'].value_counts()[:10]
```

Items	1	2	3	4	5	6	7	8	9	10
Number	593805	256860	106597	52164	24724	12909	6968	4475	2582	1516

The feature ’TotalLineItems’ means the number of the item in the shopping cart when the customer conform his purchase. We can divide the ’TotalLineItems’ into 7 parts as shown in figure 2.6.

We can see from the figure 2.6 that when a customer confirm his purchase, the more items in his shopping cart, the higher possibility that he will return this product. In our lives, there some customer buy several shoes at one time. After receiving those shoes, they would try them all, choose the most satisfying one and return the others. Normally, customers with just 1 item in their shopping cart when they pay for it are more likely to be determined to buy this product or they just don’t care so much of the shoes. So this kind of person usually have a lower return rate.

2.2.2. The price of the shoes

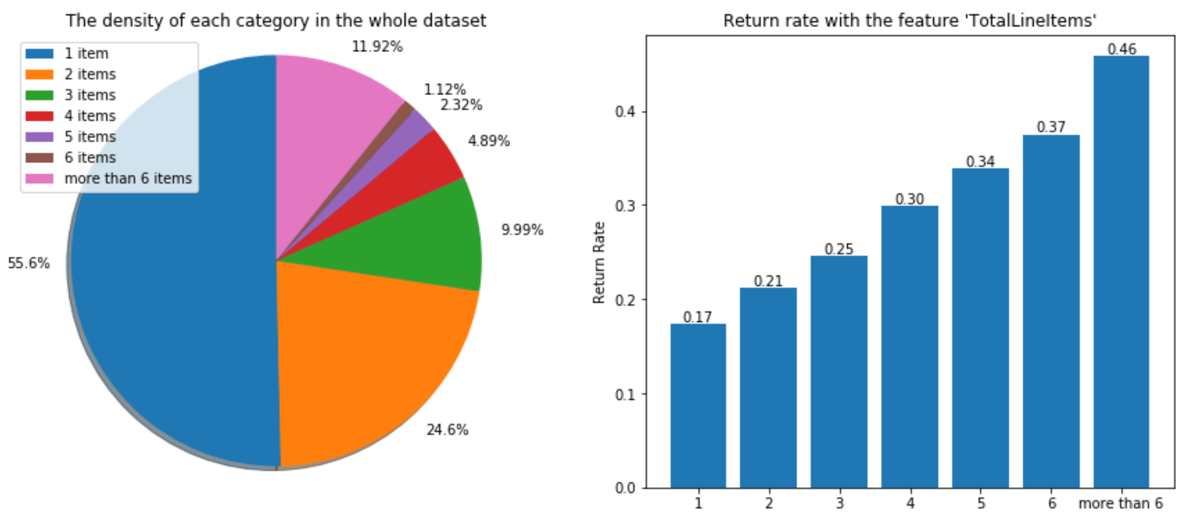

Figure 2.6: Feature ’TotalLineItems’

```
X_train_final['UnitPMPEUR'].describe()
```

Items	count	mean	std	min	25%	50%	75%	max
Value	1.07e+06	29.8	20.39	0	16.8	26	37.9	1100

The feature ‘UnitPMPEUR’ means the price of the shoes. Since the price is continuous, we regrouped the numbers, we cut the whole values into 11 sections as shown in figure 2.7.

We take the means of each section as the xlabel and draw a line chart to present the relation between the return rate and the prince. We can see that the expensive shoes are more likely to be returned. Maybe customers buy expensive usually by implusions. After they calm down, they may regret the purchase, so they finally returned the products. As for the cheap shoes, customers may care not too much about the refund and don’t take trouble returning it.

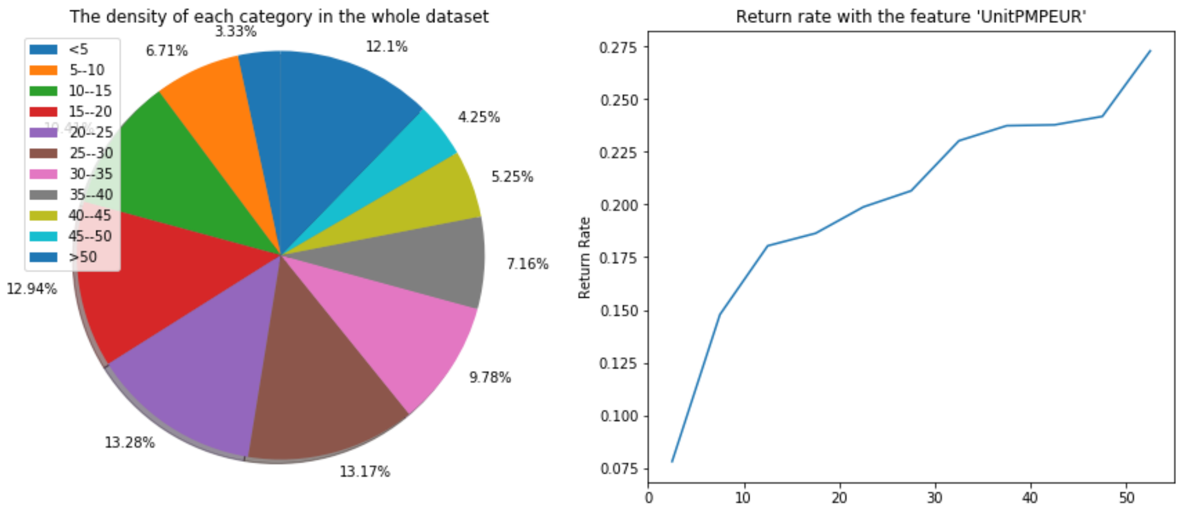

Figure 2.7: Feature ‘UnitPMPEUR’

2.2.3. Time gap between two purchases

```
    X_train_final['OrderInterval'].describe()
```

Items	count	mean	std	min	25%	50%	75%	max
Value	7.90e+05	375	503.30	0	0	118	639	2849

The feature ‘OrderInterval’ means the number of the days between the day of this purchase and the day when the customer buy a shoes of this company for the first time. We regrouped the ‘OrderInterval’ and chose 14 sections as shown in diagram 2.8.

‘OrderInterval=0’ could be the first purchase of a customer and it could also mean that a customer makes his second or third purchase in the same day. And this kind of customer has a obvious lower return rate than others. In fact, the return rate grows along with the time gap between two purchase of one customer.

But in total, it is higher than the global return rate.

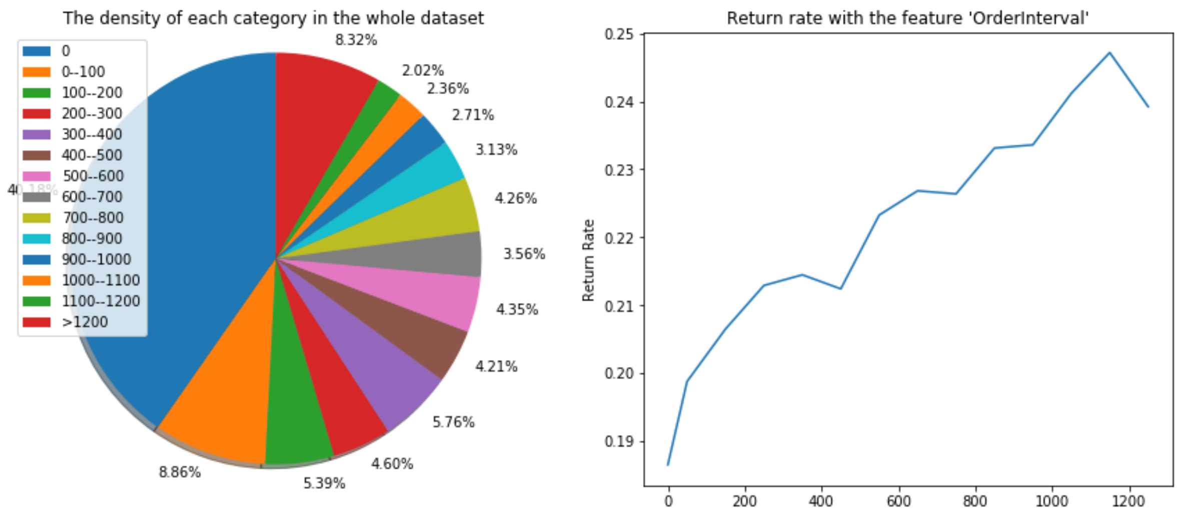

Figure 2.8: Feature ’OrderInterval’

2.2.4. Customer’s age

```
X_train_final['OrderCreationAge'].describe()
```

Items	count	mean	std	min	25%	50%	75%	max
Value	7.90e+05	37.34	11.60	-4.00	30	37	44	245

The feature ‘OrderCreationAge’ is got by the birthday in customer data, they are given by the customers by their willings, and some customers may give the wrong information. That’s why there are some negative numbers. Still, we regrouped these data into 8 sections as shown in figure 2.9.

We take the means of each sections as xlabel, and draw the line chart above. We can see that the return rate is extremely high in two sides, especially the left part (customers less than 5 years). Usually, the age less 10 years old or more than 70 years old are given by customers who pay more attention on their privacy. However, it seems that they are more likely to return a product.

And we can see from chart that there is a obvious minimum value in the section which represents the teenagers between 15 and 25 years old.

2.3. The relation between two features

We have also considered the situation of two different features combining together and then cause a larger influence to the return rate. We came up with a idea of finding two features that appears together frequently in the returned items. It’s like the classical "beer and diaper" case. In our case, we just take every feature as items in the basket. For example, we supposed that if a man buys a woman shoes, than he is more likely to return. So the ’Gender_homme’ and ’ProductGender_femme’ would often appear together in the returned items. What we are going to do is to find out if there exists a pair of discrete feature works like that.

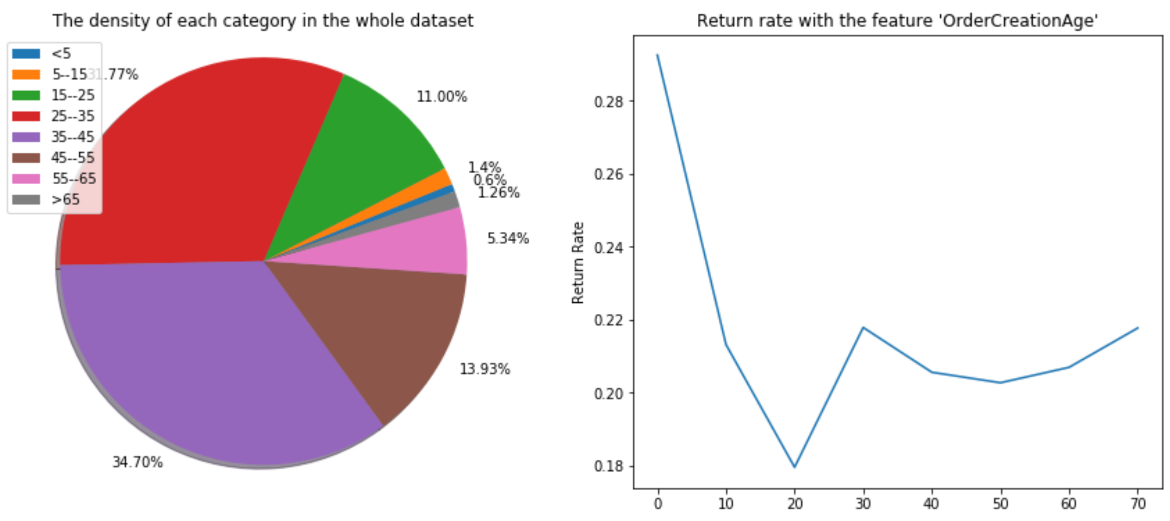

Figure 2.9: Feature 'OrderCreationAge'

First, we separate the whole dataframe into two parts, we chose all the returned items and formed a new dataframe to deal with. After using the "funk_mask" function, we ignored all the continuous data, and we got a dataframe in which there are only 0-1 data. The next step is to collect all the index of the data "1" for each item. Then we can use the "fpgrowth" algorithm to count the frequency of occurrence.

Items	Values	Items	Values	Items	Values
3	216482	(3,6)	117835	(3,9)	163488
(3, 9, 17)	102749	(3, 9, 28)	141010	(3, 9, 28, 46)	117229
(3, 9, 46)	135221	(3, 9, 53)	106740	(3, 9, 71)	102045
(3, 9, 71, 89)	100245	(3, 9, 89)	105619	(3, 17)	131478
(3, 17, 28)	114722	(3, 17, 46)	106170	(3, 28)	168929
(3, 28, 46)	140139	(3, 28, 53)	112355	(3, 28, 71)	105201
(3, 28, 71, 89)	103337	(3, 28, 89)	108762	(3, 46)	177184
(3,46,53)	110654	(3, 46, 71)	108450	(3, 16, 71, 89)	106376
(3, 46, 89)	112185	(3,53)	137439	(3,71)	136249

This is the result, the numbers in the bracket pairs are the index of features in the dataframe, for example, (3):216482 means among all of the return items, there are 216482 items carry the fourth label('OrderTypelabel_DIRECT'). (3, 6):117835 means that there are 117835 items carry both the fourth label ('OrderTypelabel_DIRECT') and the seventh label ('SeasonLabel_Printemps/Eté').

However, we found out that this result is much influenced by the quantity, so we couldn't find the potential relations between two features and we have to do some preparations before applying this algorithm.

We calculated the individual return rate for each label. Then we select 20 highest labels (individual return rate>30%) in the figure 2.10.

```
index = []
for i in np.arrange(0, rate.size):
  if rate[i] > 0.3:
    index.append(i)
item_return.columns[index]
```

Result:

```
Index([u'PayementModeLabel_BankTransfer_DE', u'IsoCode_AT', u'IsoCode_DE',
u'Gender_NC', u'MarketTargetLabel_contemporain',
u'UniverseLabel_Soir??e', u'ProductType_Bottes',
u'ProductType_Escarpins', u'SubtypeLabel_Babies',
u'SubtypeLabel_Bout ouvert', u'SubtypeLabel_Bout pointu',
u'SubtypeLabel_Cavalieres', u'SubtypeLabel_Low boots',
u'SubtypeLabel_Salomes', u'SubtypeLabel_Santiags',
```

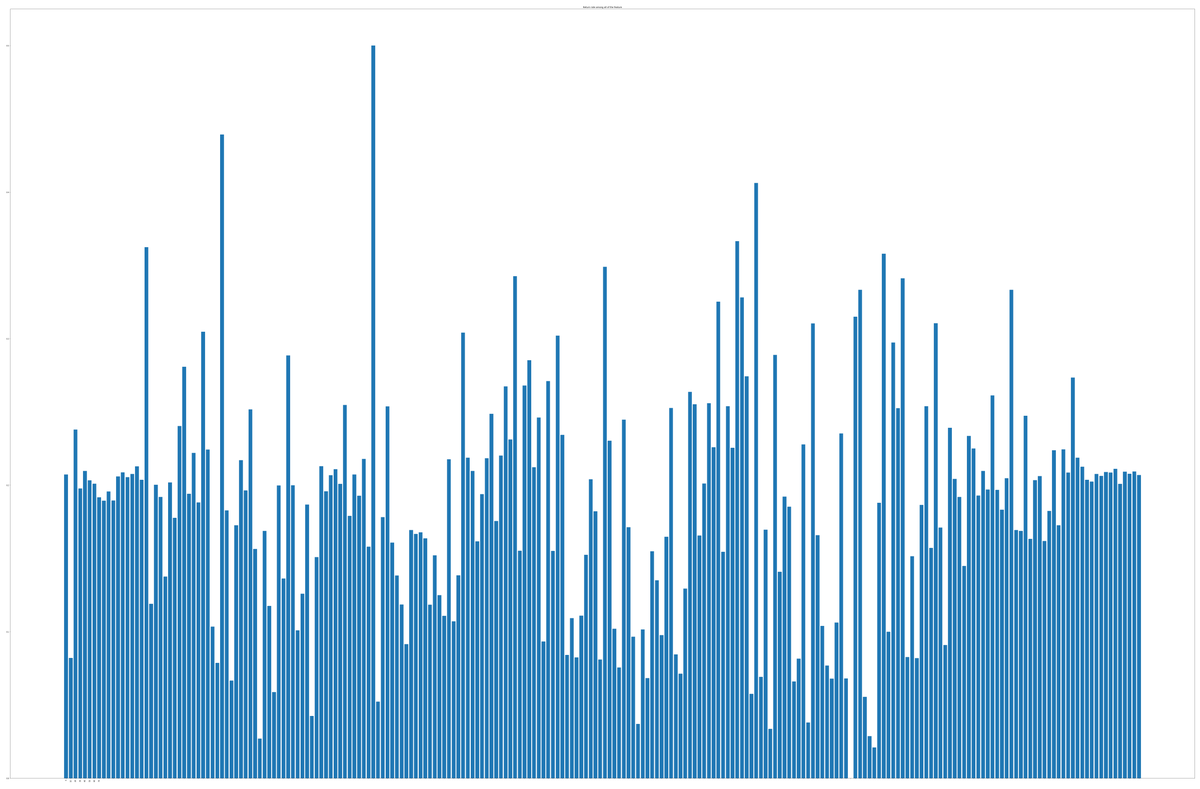

Figure 2.10: All feature

```
u'SubtypeLabel_Stretch', u'SubtypeLabel_Talon fin et aiguille',
u'UpperMaterialLabel_Cuir verni', u'OutSoleMaterialLabel_Bois'],
dtype='object')
```

```
np.array(num).T
```

Result:

For those labels whose quantity is less than 1000, we just ignored them. In order to reduce the influence of the quantity, we separated these labels into to two groups according to the quantity.

The first group is for the labels which appears more than 1000 times but less than 5000 times.

```
[0, 9, 17]
```

There are only 3 labels in this group: "PayementModeLabel_BankTransfer_DE" (index 0) which appears 2551 times, "SubtypeLabel_Bout ouvert"(index 9) which appears 3227 times, "UpperMaterialLabel_Cuir verni"(index 17) which appears 1626 times. Then we applied "fp-growth" algorithm to this group with a threshold of 200 (the pair of labels will only be printed if they appears more than 200 times). Finally we got this result:

```
patterns = pyfpgrowth.find\_frequent\_patterns(X\_return, 200)
patterns
```

Result

```
{(0,):2551, (1,): 3227, (2,): 1626}
```

It means that there are relation between two labels in this group. So we deal with the second group, in this group, all label appears more than 5000 times, but less than 20000 times.

```
  [2, 6, 7, 10]
```

There are 4 labels in this group: "IsoCode_DE" (index 2) which appears 16520 times, "ProductType_Bottes" (index 6) which appears 9636 times, "ProductType_Escarpins" (index 7) which appears 16925 times, "SubtypeLabel_Bout pointu" (index 10) which appears 5301 times. Then we applied "fp-growth" algorithm to this group with a threshold of 2000. Finally we got this result:

```
patterns = pyfpgrowth.find\_frequent\_patterns(X\_return, 2000)
patterns
```

Result:

```
  {(0,): 16520, (0,2): 2098, (1,): 9636, (2,): 16925, (2, 3): 3327, (3,): 5301}
```

We were surprised to find that "ProductType_Escarpins" and "SubtypeLabel_Bout pointu" appear together 3327 times, that is to say, for a returned item, if it carries the label “SubtypeLabel_Bout pointu”, than it is 62.76% the possibility that it carries also the label “ProductType_Escarpins”. So maybe there are some potential relations between these two labels. In order to affirm this conclusion, we checked these two labels in the dataframe of no-returned items. We applied the “fp-growth” algorithm again, and got this result:

```
df_test = pd.DataFrame(columns=['ProductType_Escarpins'], data = items_not_return['
                                                ProductType_Escarpins'])
df_test['SubtypeLabel_Bout pointu'] = items_not_return['SubtypeLabel_Bout pointu']
X_return_test = transform(df_test)
patterns_test = pyfpgrowth.find_frequent_patterns(X_return_test, 2000)
patterns_test
```

Result:

```
{(0,): 31570, (0, 1): 6120, (1,): 10854}
```

We can see that the percentage of those two labels appear together is 6120/10854, which is not far from the 62.76%. So the combination of these two labels don’t influence much on the return rate.

So, our idea of finding the combination of two features which would bring a bigger influence on the return rate didn’t work out.

3 Computation

After loading the data, the first thing that needs to be done is data processing, namely the feature engineering. We observe each feature. In order to create new features that are easy to classify. On the one hand, we merge some features that are too sparse. On the other hand,we do processing on those data which might have relations among them. As for how to decide 2 linked features, it is based on life experience.

In this section, we will create as many new features as possible. As for which features will eventually be adopted, it is due to the performance of these features in the classifier.

With regard to the features of the individual return rate, the individuals include customers, products, brands, and so on. Take the customer as an example, we extract a part of the customer's historical return records in X_train and y_train. Then we generate new features such as CReturnTimes, COrderTimes and CReturnRate, and add them to the database of customers. CReturnTimes represents the number of items returned by the customer in this portion of X_train; COrderTimes represents the number of items purchased by the customer in this portion of X_train. Then we have: CReturnRate = CReturnTimes/CorderTimes. Just like a series of features C(CReturnTimes, COrderTimes, CReturnRate) which is added for CustomerId, we also added V for VariantId in products (VReturnTimes, VOrderTimes, VReturnRate, followed by the same naming rule), added P for ProductId, added PC for ProductColorId PC, added B for BrandId.

It should be emphasized that the above-mentioned a portion of the X_train and y_train refers to the first 600,000 lines of data, which will not be reused in subsequent supervised learning process. It is precisely because we think that the individual's historical record is very important, 600,000 lines of data are used from the training set to get individual features (historical records). As for whether the classifier trained by our reduced training set is effective and how the figure of 600,000 itself is decided, it is the conclusion of our experiments. we have tried and compared several ways. That is, "not using individual features and using all training set for training", or "extracting 400,000 lines of data for individual features and the rest for training", or "extracting 800,000 lines of data for individual features and the rest for training." Our existing practice performs better among all our attempts.

One customer's birthday data in the original customers table is incorrect, because the number is too large and it causes an error because the date cannot be calculated. So we changed the customer's birth year from the original 4958 to 1958. This is not the only data with unreasonable birth date, but it is the only one that affects the code running. Regarding to the Gender feature, we find that except for two NCs, the rest are either male or female. So we can create new features "IsMale" in which 1 for True, 0 for False. Thus after applying musk, the options in the feature gender are reduced from the original 3 to 1. FirstOrderDate seems to have very useful information, but the date itself is difficult to put into the classifier as a feature. So in order to use it, we convert the time point to a time period, building a new feature "FirstOrderTime". we find that the earliest FirstOrderDate in the training set is 2005. So we set a time point "2005-09-01 00:00:00" as the starting point. FirstOrderTime is the difference compared to the starting point counted by month.

SeasonLabel in product database and SeasonLabel in X_train have the same name, so we have renamed it to ProductSeasonLabel in product database.

Similar to the aforementioned "IsMale", we have created the new features "IsProductSeasonAH" and "IsRemovableSole" for ProductSeasonLabel and RemovableSole. GenderLabel includes Femme, Homme, Enfant, Sacs, and Accessoires. We find that Sacs and Accessoires are relatively few. We have merged them into Not shoes, and we have added it to the new feature ProductGender along with Femme, Homme, and Enfant. In the same way, we process features UniverseLabel, TypeBrand, ProductType, SupplierColor, SubtypeLabel, and SizeAdviceDescription to generate the corresponding new features: BrandUniverse, BrandType, TypeProduct, IsBlack, Subtype, Description.

Observing MaxSize, we find a lot of "unreasonable" shoe sizes, most of which are not the shoe size, but the size of Sacs or Accessoires. So we only consider shoes by ignoring the items with label “Not shoes” in the feature ProductGender. Besides, the shoe size must be less than 60. We save the filtered data into ShoesSizeMax. Similar operation is done on MinSize and a new ShoesSizeMin is created.

Then, just like the way we process IsMale mentioned above, we deal with the features of only two options. Create “IsOrderDirect”, “IsNouveau”, “IsSeasonAH” based on OrderTypelabel, CustomerTypeLabel, and SeasonLabel.

Like the aforementioned ProductGender, we deal with features that have more options and we merge those options that have small amount into one option. Create New “PayementMode”, “Country”, “IsDeviceND” based on PayementMode, IsoCode, DeviceTypeLabel.

Then we process the missing value. We observe that there are quite a lot missing values because the customerId and VariantId are not found in the table customers and products. The situation of missing data is the same, so we create New “IsCustomerNull” and “IsProductNull” to mark this situation. In addition, there are some features with additional data missing. To solve this problem, we create new IsCalfTurnNull, IsUpperHeightNull, IsHeelHeightNull, IsSubtypeLabelNull, IsRemovableSoleNull, IsDescriptionNull, and IsMaterialNull. It is worth mentioning that since the amount of data for UpperMaterialLabel, LiningMaterialLabel and OutSoleMaterialLabel is too small, we will merge these three items. If these three items are all NaN, then record IsMaterialNull as 1.

In addition, we have also built several new features.

PriceHTRate represents the ratio of PurchasePriceHT to UnitPMPEUR. It should be noted that 0.01 is added both on numerator and denominator, in order to avoid error in several cases below: 1.in the case where the price is zero(denominator is zero); 2.in the case that PricePriceThrough and UnitPMPEUR both equal to 0.

IsRateHigh indicates whether PriceHTRate is greater than 1.

OrderInterval represents the number of days passed from the FirstOrderDate to the OrderCreationDate.

OrderCreationAge represents the user’s age at the OrderCreationDate. As mentioned earlier, some of these ages are unreasonable, such as being too young or too old. We still remain this data because the accuracy we get here is the user’s self-proclaimed age, and the false age itself is a type of feature that can be used for judgment.

Next, we want to extract the information in OrderCreationDate as much as possible. The year should be useless. Our data is arranged in the order of time. The time of the test set is after the training set, and decision tree model is hard to make effective predictions for unknown data. So we only extract the month (OrderMonth) and date (OrderDay), and further get the week (OrderWeekday). These are continuous values of the int type. We will then discretize it to get OrderSeason (Spring, Summer, Autumn, Winter), OrderPeriod (in the first-middle-last ten days of a month), OrderDayofweek (Monday to Sunday) and IsOrderWeekend.

In addition, although OrderShipDate seems to be a very helpful feature, we can’t actually use this feature, due to the requirement that we need to predict the return probability at the moment of purchase.

	CustomerId	CountryISOCode	BirthDate	Gender	FirstOrderDate	CReturnTimes
578916	12568532.0	FR	4958-02-12 00:00:00	Femme	2011-12-09	15:19:37

4
Conclusion