

## 1. 概述

这个数据是关于黑色星期五一家零售店的交易。黑色星期五是美国感恩节后的星期五的一个非正式名称，在11月的第四个星期四庆祝。许多商店在黑色星期五提供高度促销的销售，这是自2005年以来美国一年中最繁忙的购物日。

它是Kaggle上一个非常流行的数据集，许多其他数据科学家已经研究了 this 数据集。他们的大多数分析只是EDA对原始数据，而其中一些也做预测。有一份报告，其中使用了一些先进的方法，似乎有类似的主题与外部项目。以下是该报告的链接：

<https://www.kaggle.com/dabate/black-friday-examined-eda-apriori> 直到我们完成我们的项目，我们才看过任何关于这些数据的研究报告。我们可以保证我们所有的工作都是100%原创的。

我们的主要目的是增加零售店的未来销售和利润。为了实现这样的目标，一方面，我们将客户聚类成几个组，零售店可以针对不同类型的客户提出不同的营销策略。另一方面，我们发现哪些产品可能是一起销售的，所以零售店可以决定是否捆绑销售这些产品以增加销售。我们的工作分为五个步骤。首先，在第一步设计思维中，我们思考数据和利益相关者，数据的场景使用和限制，然后决定我们的目标是什么，以及应该做些什么来实现这个目标。之后，在步骤2中获取我需要的数据，我们想知道我们所拥有的数据是否足够，同时也考虑到数据隐私问题..然后在步骤3中是适合使用的数据，我们对数据集进行了一般的查看，在其上实现了EDA，并将数据转换成适当的形式供以后使用。当涉及到第四步，使数据坦白时，我们实施了一些数学方法来处理数据，以解决我们在设计思维中提出的问题。最后，我们将在最后一部分故事讲述中展示我们的工作和结论。

## 2. 设计思维

“黑色星期五”，正如我们所知，今天是一个奢侈的销售，促销和长线外的商店。像Target、百思买、亚马逊和其他许多零售商都期待着这一天的到来，他们希望消费者能利用上门交易。“黑色星期五”一词也催生了其他零售假日，如“网络星期一”、“小企业星期六”和“给予星期二”。由于每年的销售活动，精明的购物者可以在圣诞节前节省大量的钱-甚至有助于消除最后一分钟购物的压力。

尽管黑色星期五和网络星期一的销售起源于美国，但在澳大利亚，无论是零售商还是购物者，都变得非常受欢迎，永远改变了当地的购物日历。在节礼日的销售之外，11月的最后几个星期是全国各地的商店在大量的产品（从服装到电子产品到家具）中提供一些最大的折扣来吸引购物者。此外，在中国，单身日就像黑色星期五一样，在这段时间里，这是一个折扣季节。

根据澳大利亚统计局(ABS)的数据，零售商肯定正在收获采用黑色星期五销售期的好处，2018年11月记录了迄今为止最高的在线零售营业额。澳大利亚统计局报告说，2018年黑色星期五期间，澳大利亚整体零售营业额比前一年增长了3.6%。在阅读了这份报告后，考虑到黑色星期五的销售似乎越来越激烈，下面，我们可以期待什么

一年的交易，我们能为明年的交易做些什么？

我们发现它有趣的是检查一个围绕一个假设的商店和它的购物者的数据的数据集，然后我们从Kaggle找到了一个合适的数据集。正如作者所描述的，“数据集由550,000个关于零售商店的黑色星期五购物者的观察组成，它包含不同类型的变量，无论是数值变量还是分类变量。它包含缺失值。”

数据集是关于零售商店中的交易。老实说，店主想更好地了解客户对不同产品的购买行为，并找到一些潜在的客户，除了VIP客户作为他们的新目标，以推广他们的项目，以便让商店在下个黑色星期五获得更多的利润。具体来说，这里的问题是分类问题，它可以在这个数据集中解决，因为几个变量是分类的，其他一些方法可能是“预测消费者的类型”，甚至“预测购买的商品类别”

经过具体的分析，这个数据集特别方便聚类。在典型的使用中，获取更多利润的最有效和最直接的方法是在其中找到不同的消费者集群，并预测在所购买的商品类别之间捆绑销售的可能性。更重要的是，店主为不同的消费者群体提供独特的服务，增加销售是非常重要的。在几乎典型的使用中，虽然该EDA按Tableau和通过R使数据坦白只使用所提供的数据集，但类似的技术可以应用于任何类似的数据集或业务问题。在非典型用途中，该数据集也可用于供应商解决配送中心和劳动力池的问题，以便更好地准备在黑色星期五。




正如我们所知，旨在通过提供更多的黑色星期五优惠来吸引更多客户的竞争每年都会变得更加激烈。同样，对于消费者来说，寻找最佳黑色星期五交易的竞争也在逐年加剧。如果这个数据集有关于产品库存和货物价格的数据，客户可以在他们自己的愿望清单中准确地获得关于本地库存和特定折扣的产品细节。对于所有的利益相关者，包括店主，客户，供应商，他们可以从我们基于这个数据集的分析中得到很多好处，并让客户更多地利用在黑色星期五的上门交易。






### 3. 得到我需要的数据

根据该数据集，可以观察到以下数据类型，包括结构化和空间。数据集可以通过外部来源(如Kaggle)获得，并且可以自由使用。值得注意的是，这个数据集已经为完美的隐私做了很多匿名化，但是，我们也很难展示更详细的消费者和产品信息，我们将尽力为我们的分析和结论提供更好的解释。

## 4. 数据是否适合使用

### 4.1. 数据集介绍

User_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status
User	Boolean	Age customer	Id Occupation of each customer			
	M 75% F 25%	26-35 40% 36-45 20% Other (5) 40%		B 42% C 31% Other (1) 27%	1 35% 2 19% Other (3) 46%	
1000001	F	0-17	10	A	2	0
1000001	F	0-17	10	A	2	0
1000001	F	0-17	10	A	2	0
1000001	F	0-17	10	A	2	0
1000002	M	55+	16	C	4+	0

User_ID	Product_ID	Product_Category_1	Product_Category_2	Product_Category_3	Purchase
User	Id Product				Purchase amount in dollars
	3623 unique values				
1000001	P00069042	3			8370
1000001	P00248942	1	6	14	15200
1000001	P00087842	12			1422
1000001	P00085442	12	14		1057
1000002	P00285442	8			7969
1000003	P00193542	1	2		15227

至于原始数据集，它是一个CSV文件，列如用户ID、性别、年龄、职业、城市、生活年限、婚姻状况和购买产品的信息，包括产品的ID、类别和购买金额。数据集中的每一行记录一个用户购买的产品之一。数据集中总共有537,577行，包括5,891个用户和5,623个不同的产品。用户的性别用“M”和“F”表示。他们的年龄分为7个部分（0-17、18-25、26-35、36-45、46-50、51-55和55岁以上）。在用户的职业方面，有0到20的数字用作职业ID。此外，这个数据集涵盖了城市A、B和C的人，并根据他们在他们的城市生活了多少年将他们分成5个部分（0、1、2、3和4+）。此外，用户的婚姻状况用“0”或“1”表示。（“0”表示单身，“1”表示未婚。当转到产品信息时，有18种类型的产品。并且类别ID占用并行类别的3列，这意味着一个产品至少在1个类别下，最多3个类别下。

## 4.2. 基本EDA

在我们开始做一些基本分析之前，将3个城市分开，考虑VIP用户和正常用户是更合理的，因为一些客户购买的商品比其他客户购买的商品多得多，这可能会对分析结果产生巨大影响。因此，我们首先使用库‘readr’将数据输入到R中，计算每个用户的购买量，然后生成一个数据集，其行表示所有不同的用户。

```
library(readr)
all_data<-read_csv(file=~"/Downloads/BlackFriday.csv")

# Get unique data of users
all_user<-unique(all_data[c('User_ID','Gender','Age','City_Category','Stay_In_Current_City_Years','Marital_Status','Occupation')])
# Make it in the right order so it can bind with purchase correctly
all_user<-all_user[order(all_user$User_ID),]
all_user_with_purchase<-cbind(all_user,aggregate(Purchase~User_ID,all_data,sum)$Purchase)
names(all_user_with_purchase)[8]<- 'Purchase'
```

接下来，我们在三个城市实现boxplot，并将这些不寻常的观察定义为VIP。

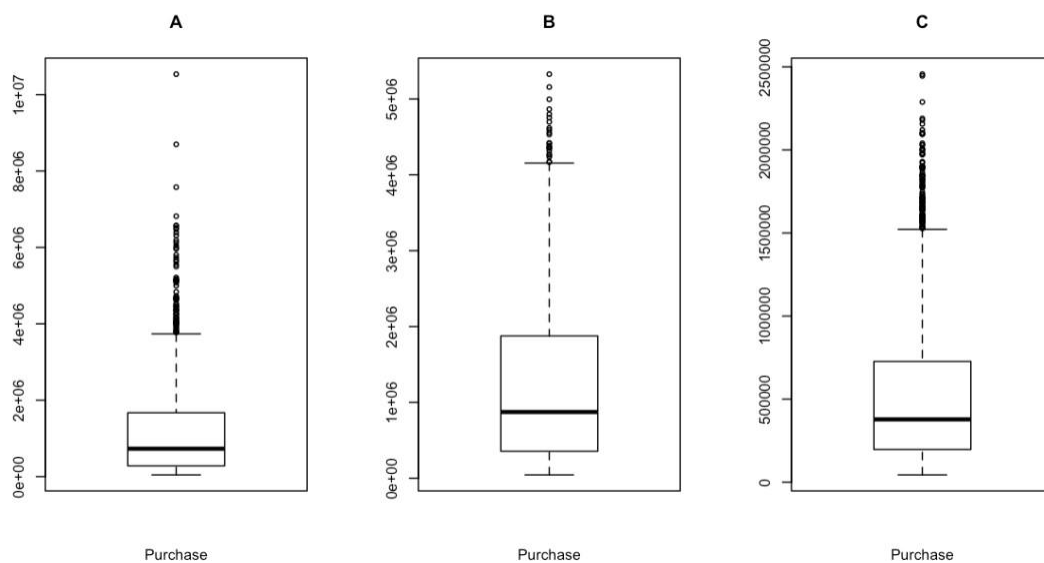
```
# Find vip users in different city (with Purchase)
all_user_A<-all_user_with_purchase[all_user_with_purchase$City_Category=='A',]
all_user_B<-all_user_with_purchase[all_user_with_purchase$City_Category=='B',]
all_user_C<-all_user_with_purchase[all_user_with_purchase$City_Category=='C',]

# Use outliers to distinguish vip and normal users
box_A<-boxplot(all_user_A$Purchase)
box_A$stats # min,1/4,median,3/4,max
box_A$out # outlier value(purchase)
nrow(all_user_A[all_user_A$Purchase>=box_A$stats[5,1],]) # The number of vip users

box_B<-boxplot(all_user_B$Purchase)
box_B$stats
box_B$out
nrow(all_user_B[all_user_B$Purchase>=box_B$stats[5,1],])

box_C<-boxplot(all_user_C$Purchase)
box_C$stats
box_C$out
nrow(all_user_C[all_user_C$Purchase>=box_C$stats[5,1],])

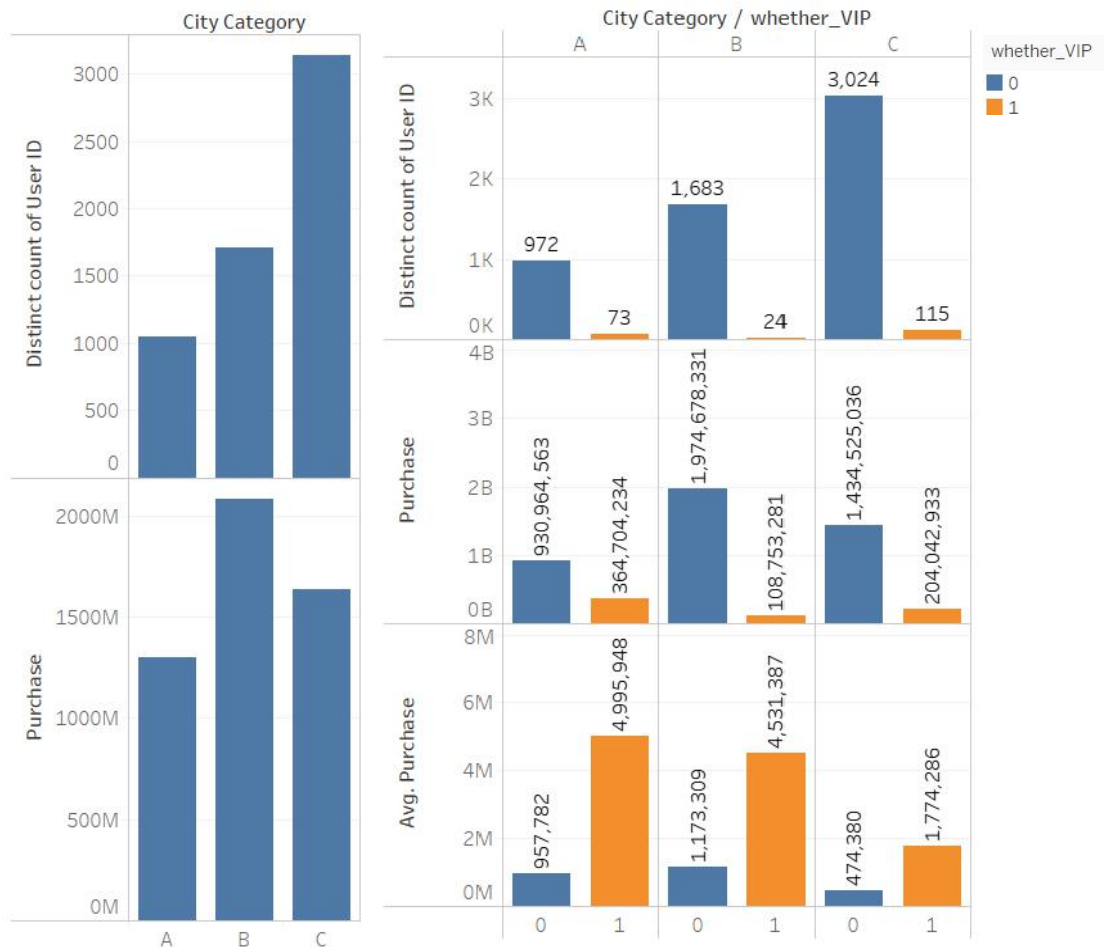
par(mfrow=c(1,3))
boxplot(all_user_A$Purchase,xlab='Purchase',main="A")
boxplot(all_user_B$Purchase,xlab='Purchase',main="B")
boxplot(all_user_C$Purchase,xlab='Purchase',main="C")
```



为了分析哪些因素会影响用户的购买行为，有一些EDA用于数据可视化。

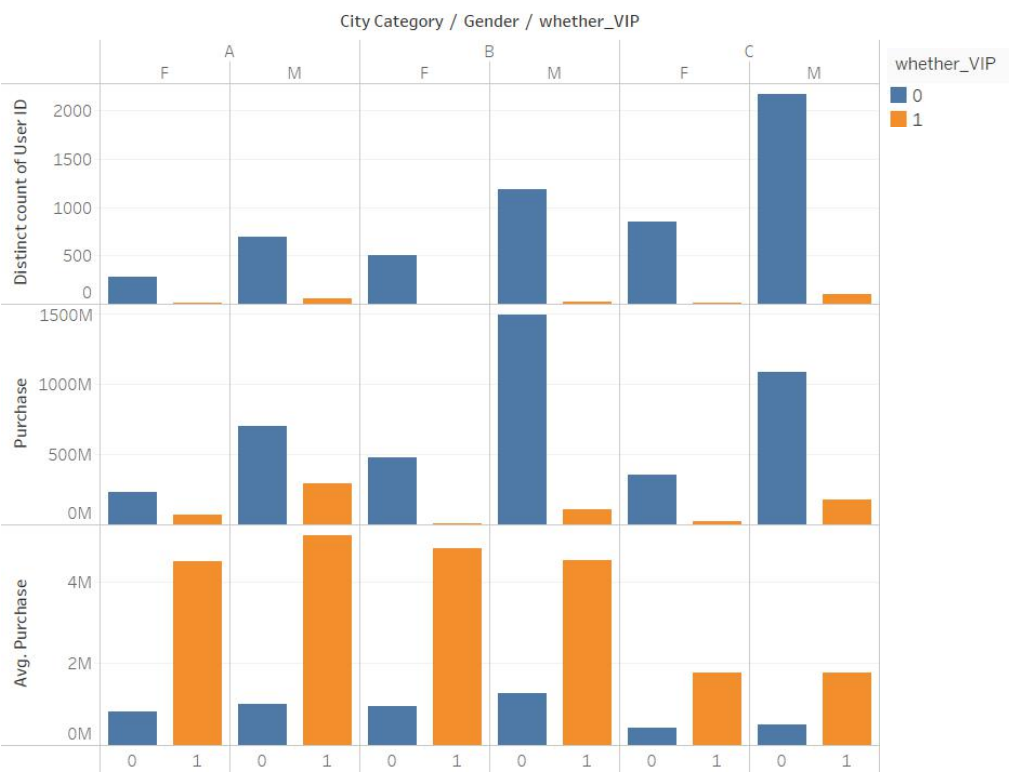
首先，从城市因素来看，C市人口最多(约为B市的两倍)，但其总购买量甚至低于B市，这是一个有趣的现象。同时，在城市A和B，购买与人口比例。因此，在进一步研究人们的购买行为时，应该考虑城市因素。

在另一种情况下，表明C市的vip用户最多，但根据他们的平均购买量（甚至不到任何其他城市的一半），他们的购买力远远弱于其他城市的vip用户。而A市的vip用户在黑色星期五购物上花费最多。至于正常用户，B市的人购买的总数和平均数最多。



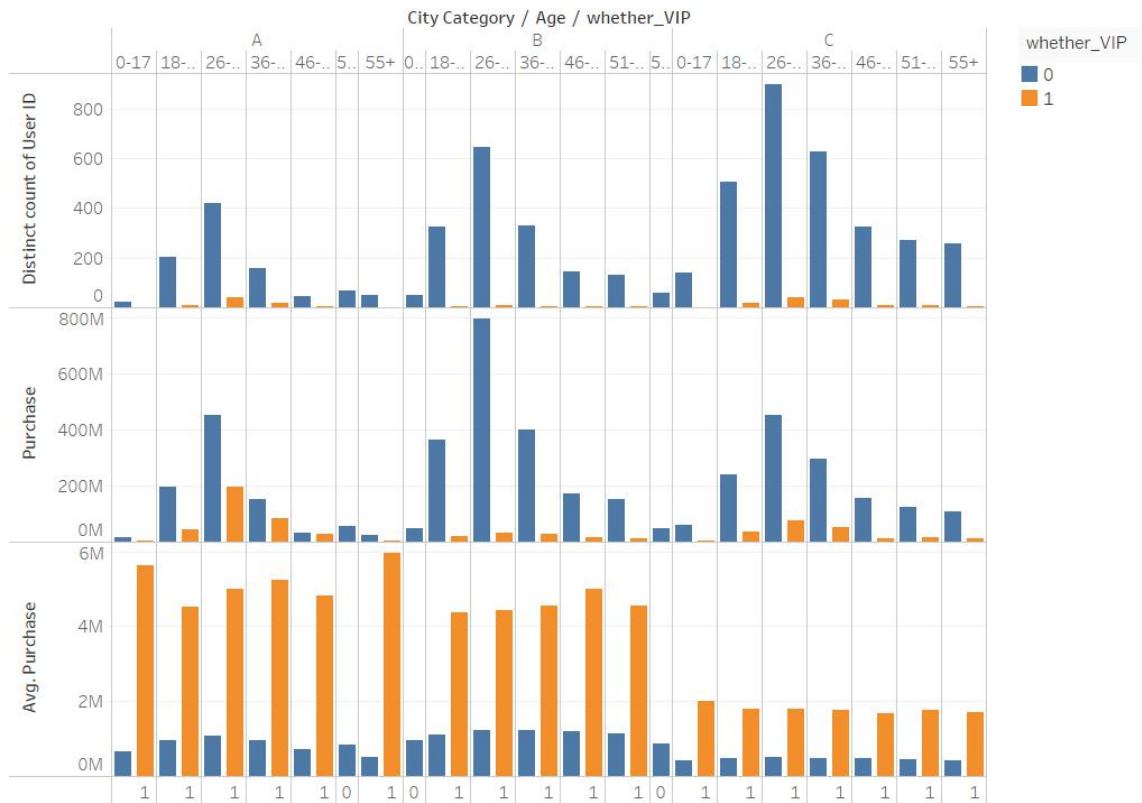
考虑的第二个因素是性别。结果表明，男性用户的数量是女性用户的两倍多，而男性用户的购买量是女性用户的近4倍。将用户分为“正常”和“VIP”后，发现VIP用户的平均购买量在性别之间接近，而正常男性用户的平均购买量略高于女性。

gender



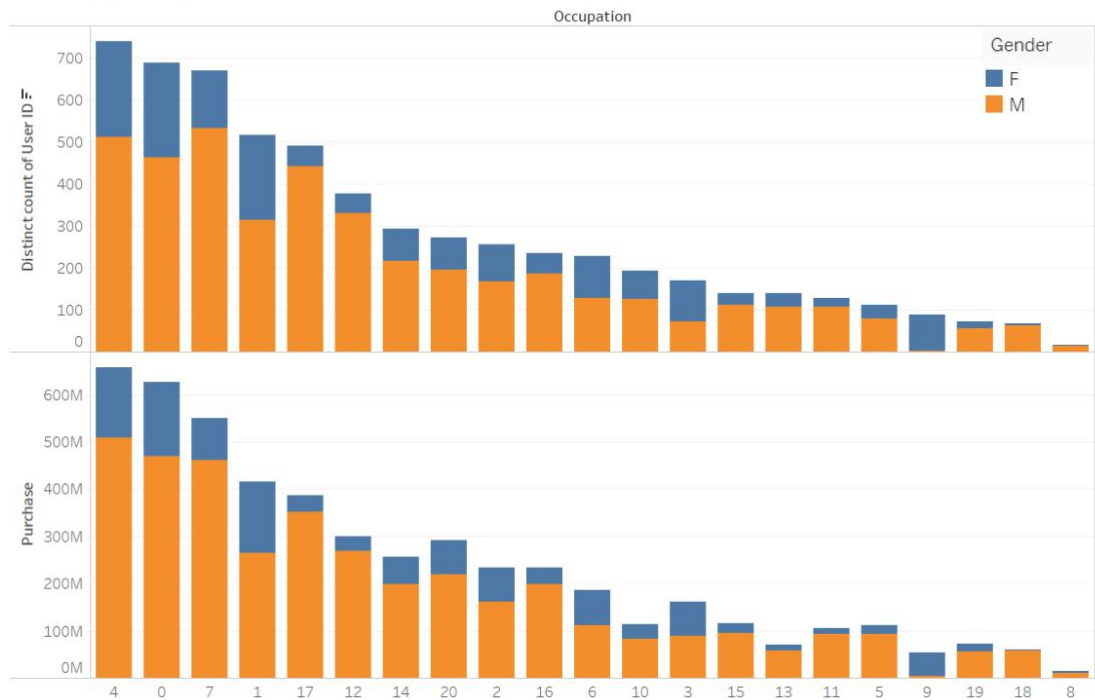
以下是年龄，3个城市的用户数量和他们根据年龄购买的趋势是相似的。在26-35岁中，用户最多，总购买量约为18-25岁和36-45岁的两倍。而其他年龄段的用户数量更少。此外，新用户最少，购买最少。根据平均购买情况，正常用户和vip用户在年龄之间有相似的购买能力，未成年人除外，他们不太可能是vip用户。

age



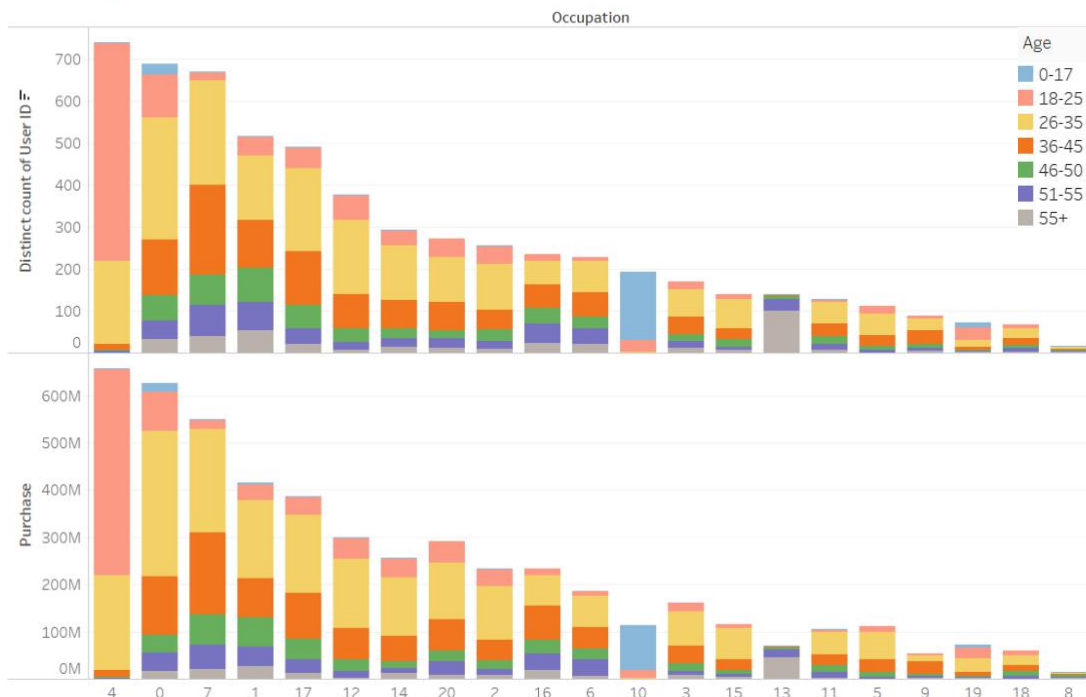
然后，在职业方面，不同的职业因其人口而在总人数和年龄分布上表现不同，但在性别分布上表现相似。除此之外，职业4、0和7的人口最多，购买最多。

occupation VS gender



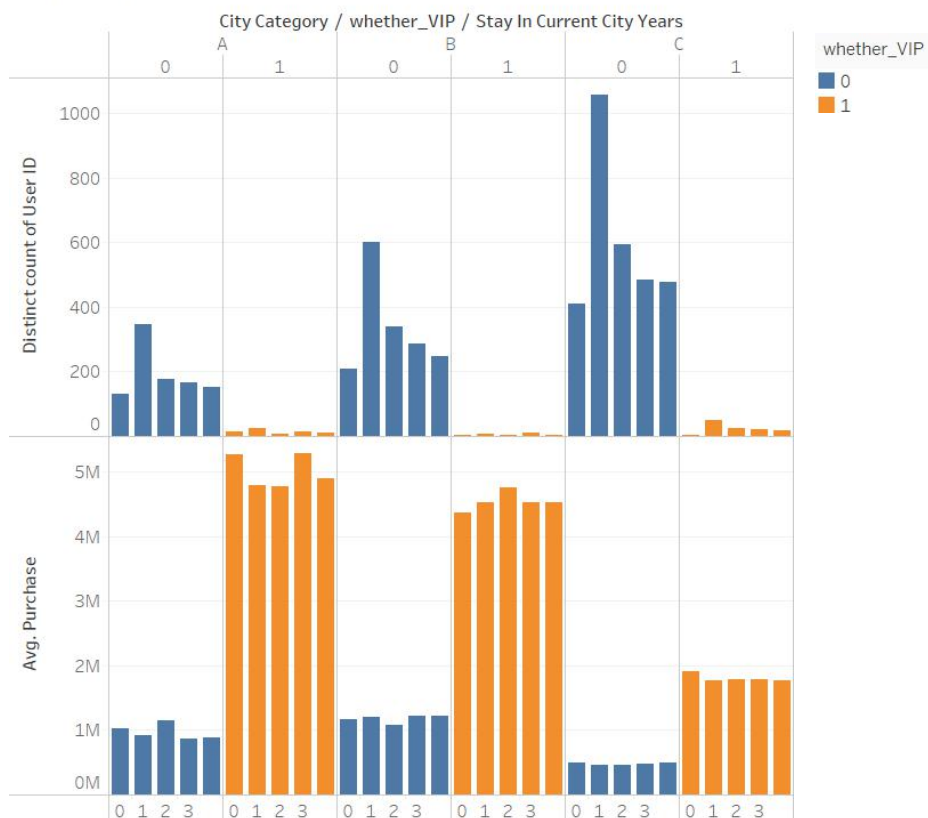


## occupation VS age



其次，居住年限和婚姻状况都显示了人口与购买的比例关系。而人口主要集中在仅停留一年的部分。至于消费能力，无论是VIP用户还是普通用户，不同停留年限之间几乎没有明显的现实关系。

## staying years





此外，3个城市在婚姻状况上的趋势几乎与未婚用户略高于已婚用户相同，消费能力的差异并不明显，无论是哪种类型的用户和婚姻状况。



#### 4.3. 数据转换

为了使数据适合我们在下一部分中的使用，我们首先必须对数据进行转换。

(1) 用于聚类

对于客户的集群部分，我们想知道他们购买了多少个类别的产品，然后根据它做集群。因此，我们将重点关注他们购买的产品属于什么类别和数量。我们创建一个1x18向量来存储用户购买的产品信息。其初始值均为0. 我们想出了两种方法来转换数据。第一种方法是，一旦出现类别，相应的位置就会增加1。例如，转换前后的数据如下所示。

User_ID	Product_ID	Product_Category_1	Product_Category_2	Product_Category_3
1000043	P00255842	16	NA	NA
1000043	P00217742	5	14	NA
1000043	P00250842	8	NA	NA
1000043	P00289942	3	4	5
1000043	P00255942	1	16	NA
1000043	P00034742	5	14	17
1000043	P00178242	8	NA	NA
1000043	P00278642	5	NA	NA
1000043	P00110842	1	2	5
1000043	P00307342	8	NA	NA
1000043	P00275142	5	8	NA
1000043	P00220242	3	12	NA
1000043	P00265242	5	8	NA

User_ID	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18
1000043	2	1	2	1	7	0	0	5	0	0	0	1	0	2	0	2	1	0

算法是，第一个产品属于第16类，所以我们在X16列中添加了1。第二个项目既属于第5类，也属于第14类，因此我们在X5和X14列中都添加了1...然后我们将为每个用户生成一个这样的记录。我们用下面的代码计算这个。

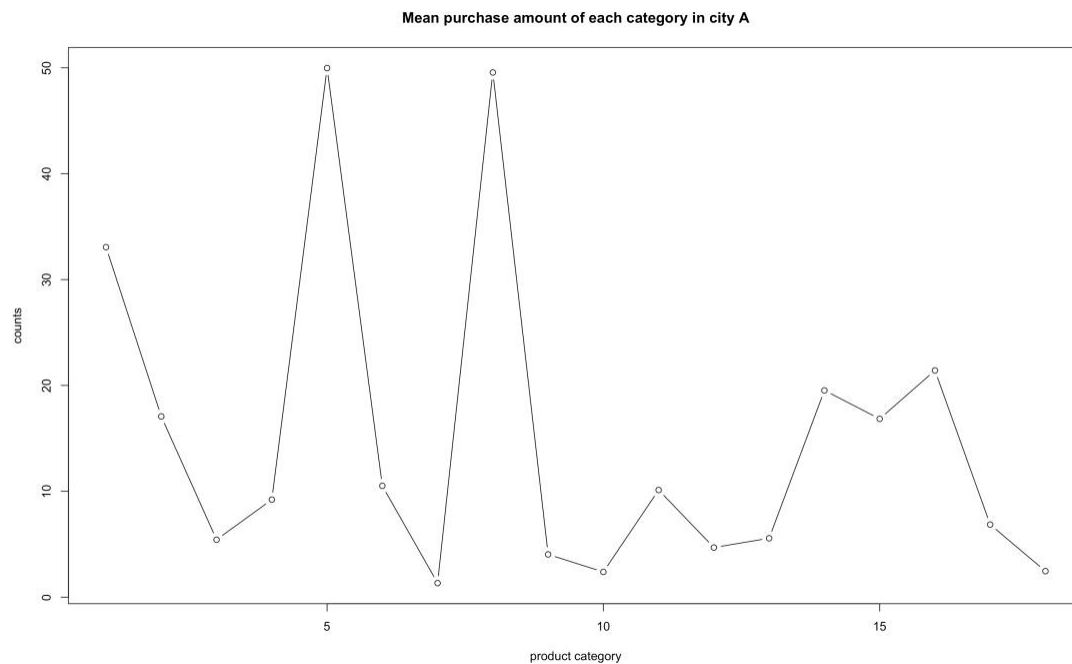
```
all_user_A_transformed<-data.frame(all_user_A,matrix(0,1,18))
all_data_A<-all_data[all_data$City_Category == 'A',]

start_tansform1_A<-Sys.time()
for (i in c(1:(nrow(all_data_A)))) {
  k=all_data_A[i,]
  all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_1]]=
  all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_1]]+1
  if (!is.na(k$Product_Category_2)) {
    all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_2]]=
    all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_2]]+1
  }
  if (!is.na(k$Product_Category_3)) {
    all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_3]]=
    all_user_A_transformed[all_user_A_transformed$User_ID==k$User_ID,][[8+k$Product_Category_3]]+1
  }
}
end_tansform1_A<-Sys.time()
```

另一种方法与第一种方法相似。而不是记录每个类别的数量，我们只关注这个客户是否购买了属于这些类别的产品。因此，转换后的数据只包含0和1.但我们认为忽视他们购买了多少产品是没有意义的。所以我们决定实现第一种方法..

然而，这还不够。后来我们意识到，有一些类别可能是日常用品或必需品，比其他类别更贵。

```
plot(colSums(all_user_A_transformed[9:26])/nrow(all_user_A_transformed),ylab = "counts",xlab =
"product category",main = "Mean purchase amount of each category in city A",type = 'b')
```



显然，我们需要在聚类之前进行规范化。我们建立一个归一化函数。  
然后在我们的数据上实现它。

```
# Bulid normalize function
normalize<-function(x){
  return ((x-min(x))/(max(x)-min(x)))
}

# Normalize data
all_user_A_transformed_normalize<-all_user_A_transformed
for (i in 9:26){
  all_user_A_transformed_normalize[i]<-normalize(all_user_A_transformed_normalize[i])
}
```

现在数据适合使用。 这里是转换数据的一部分。 我们对城市B和C也做了同样的改造。

User_ID	X1	X2	X3	X4	X5
1000001	0.019138756	0.026086957	0.22448980	0.118181818	0.014705882
1000003	0.071770335	0.130434783	0.02040816	0.009090909	0.044117647
1000005	0.086124402	0.043478261	0.04081633	0.036363636	0.076470588
1000006	0.038277512	0.043478261	0.12244898	0.109090909	0.061764706
1000015	0.148325359	0.156521739	0.02040816	0.027272727	0.105882353
1000019	0.263157895	0.234782609	0.38775510	0.154545455	0.123529412
1000020	0.047846890	0.017391304	0.00000000	0.000000000	0.005882353
1000022	0.267942584	0.252173913	0.10204082	0.090909091	0.223529412

## (2) 为了捆绑销售

对于捆绑销售部分，我们希望集中在每个用户购买的产品上，这样我们就可以分析哪些产品可能是一起购买的，然后我们可以在此基础上进行捆绑销售。因此，我们必须将数据转换为另一种格式。 我们创建一个包含超过3000列的向量，指示每个用户的所有产品。如果他/她购买了该产品，则值为1，否则为0。由于数据有这么多列，这里只是其中的一部分。

User_ID	P00069042	P00248942	P00087842	P00085442	P00193542	P00274942	P00251242
1000001	1	1	1	1	0	0	0
1000003	0	0	0	0	1	0	0
1000005	0	0	0	0	0	1	1
1000006	0	0	0	0	0	0	0

我们通过下面的代码来转换这个，这需要我们110分钟才能运行。我们对城市B和C也做了同样的改造，这分别需要200和680分钟才能运行，因为这两个城市的客户数量都比较大。最后，对数据进行了进一步的分析。

```
### Find products are probably bought together (base on city)

# Product data in A
all_product_A<-unique(all_data_A$Product_ID)
product_tuple_A<-data.frame(matrix(0,1,length(all_product_A)))
for (i in 1:length(all_product_A)) {
  names(product_tuple_A)[i]<-all_product_A[i]
}

# Transform data to see what product each user bought in A
product_A_transformed<-data.frame(all_user_A,product_tuple_A)

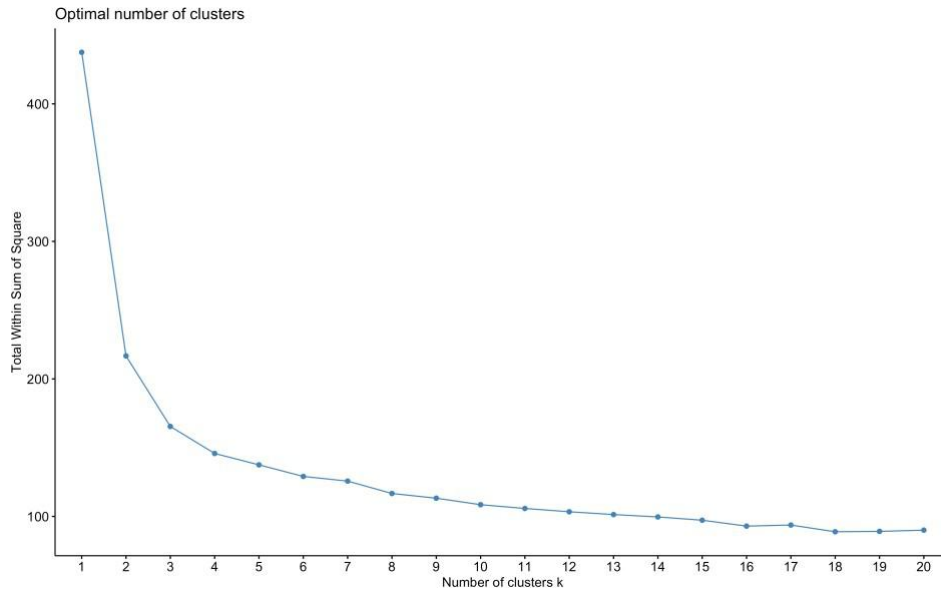
start_transform2_A<-Sys.time()
for (i in product_A_transformed$User_ID){
  k<-all_data_A[all_data_A$User_ID==i,]
  for (j in k$Product_ID){
    product_A_transformed[product_A_transformed$User_ID==i,][[j]]<-1
  }
}
end_transform2_A<-Sys.time()
```

## 5. 使数据坦白

### (1) 客户集群

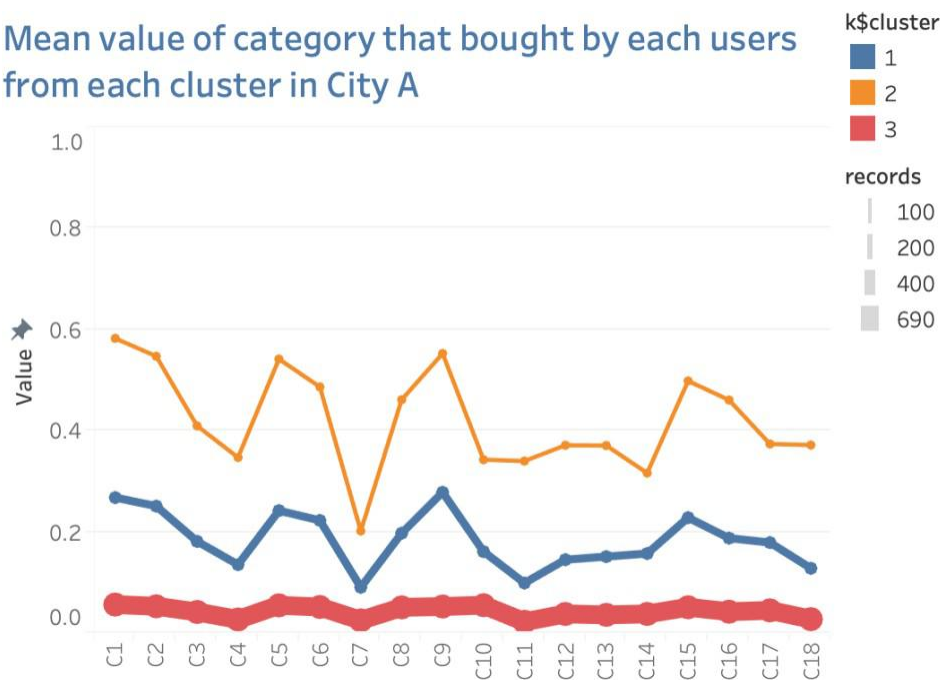
归一化后，数据适合集群。我们决定使用k均值方法基于归一化数据的欧氏距离对客户进行聚类。令我们困惑的是，我们应该做多少集群。因此，我们计算平方和内的总数，得到最合适的簇数。并绘制了一个关于总WSS和簇数的函数图像。为此，我们需要使用包ggplot2、factoextra和NbClust中的函数。首先，我们关注A市。

```
# Decide how many clusters to use
library(ggplot2)
library(factoextra)
library(NbClust)
dev.off()
fviz_nbclust(all_user_A_transformed_normalize[9:26][1:nrow(all_user_A_transformed_normalize)],kmeans, method = "wss",k.max = 20)
```



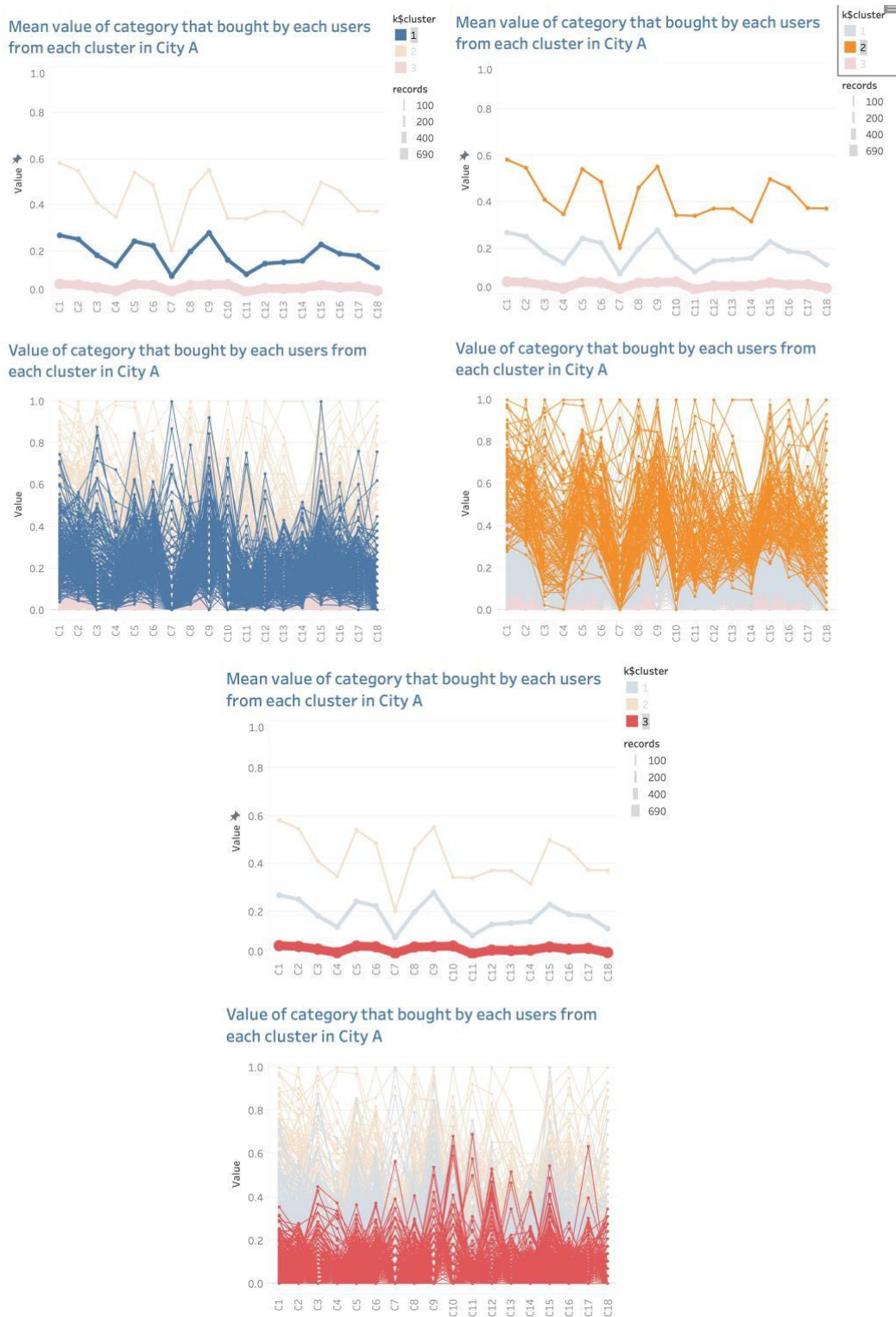
根据Elbow方法，当簇数 $k$ 为3时，似乎是合适的。然而，如果我们更进一步，看看每个集群和内部的用户，我们就会意识到3不是一个很好的 $k$ 值。

### Mean value of category that bought by each users from each cluster in City A



这是每个用户从三个集群购买的类别的平均值。由于归一化，y轴从0到1，x轴是类别的id。线的宽度意味着集群中的用户数量。我们可以看到，这3个集群在他们购买的项目类别中似乎有相同的模式，它们之间的区别只是购买金额。显然这是不合理的，没有办法所有用户都有相同的购买模式。如果我们看看每一个集群，我们就知道原因了。

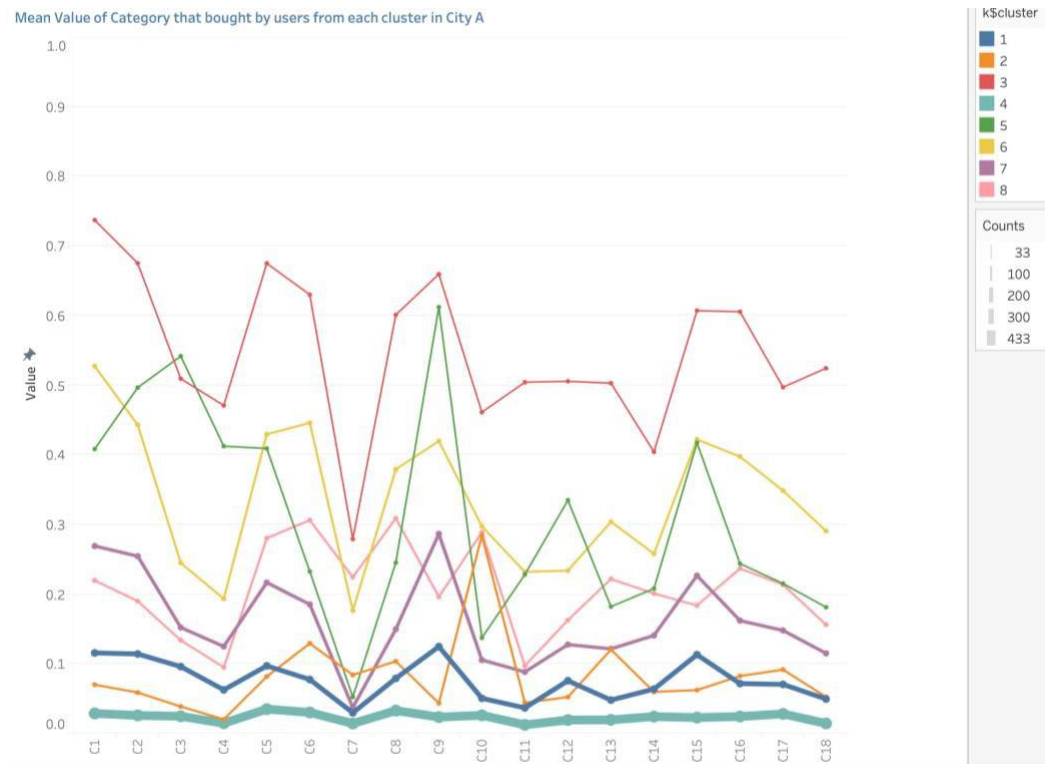




这些图片显示了三组中每个用户的具体购买信息。很明显，这些线是亮的，特别是在簇1和2 它们在每个类别的y轴上都有相当宽的范围，这意味着这些集群可能是不合理的。所以3肯定不是合适的簇数.. 我们希望集群数量足够大，以显示购买模式的差异，而它不应该太大，不应该是有用的。

那么我们应该选择什么簇号呢？？ 我们回顾WSS和集群的情节。在局部，当簇数k从7 转到8时，WSS总值似乎下降得更多。

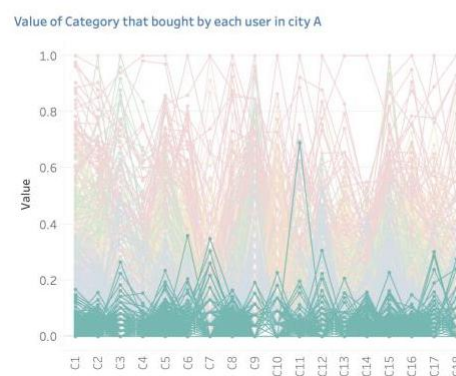
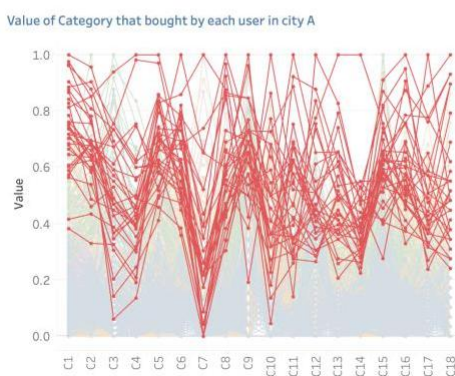
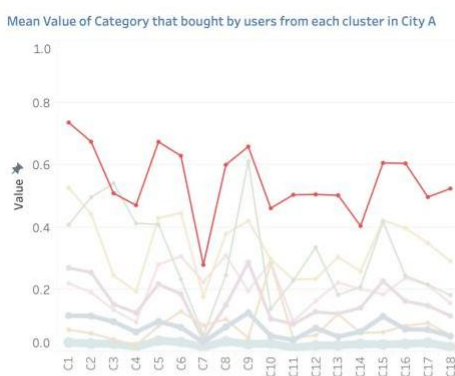
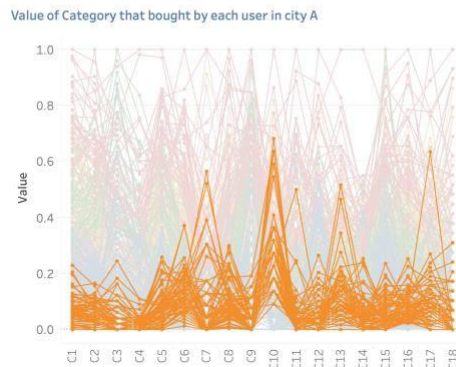
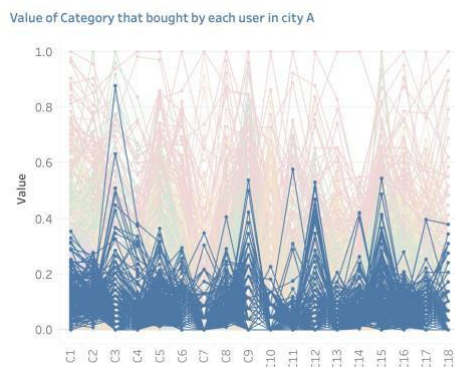
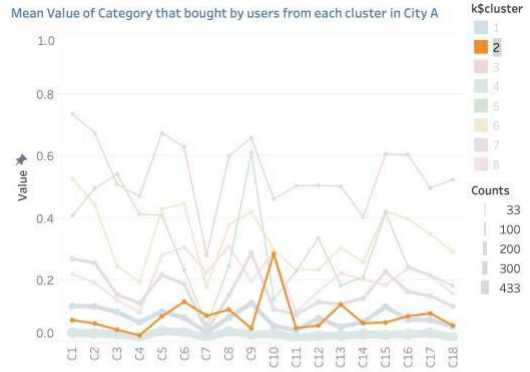
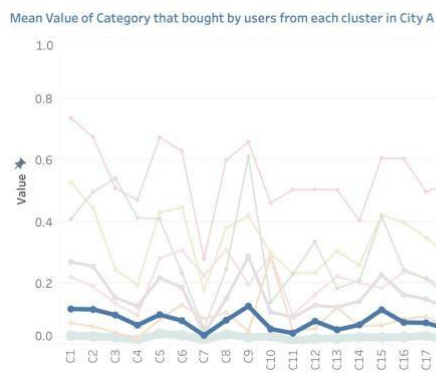
因此，我们决定在A市分组8组。

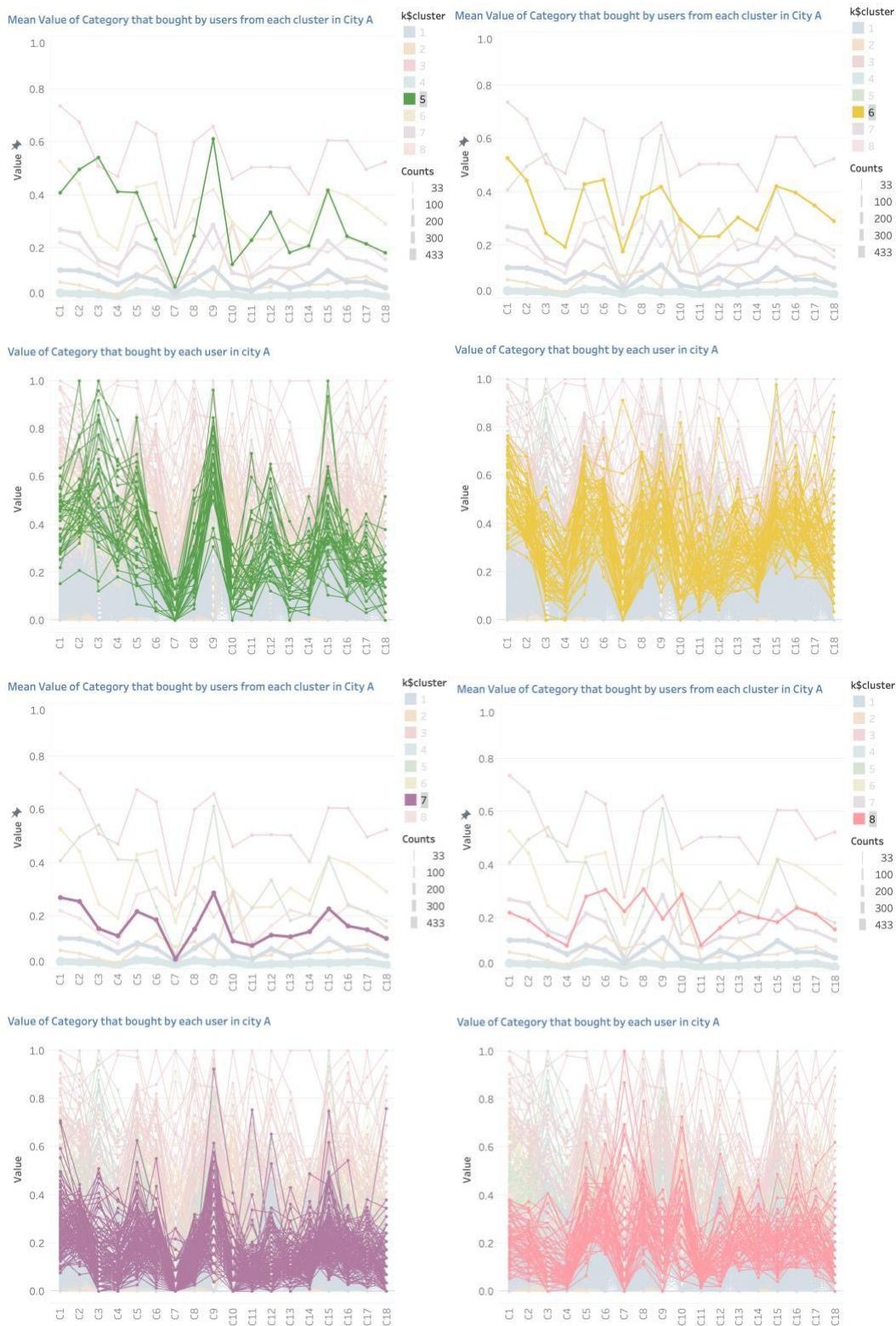


在这个图像中，我们注意到在A市，大多数人很少购买东西（集群4）。并且似乎有一些集群具有类似的模式，这可能是这个项目中的一个常见问题。一个解决方案是比较这些集群，并将其中一些集群视为相同的客户类型。在A市，我们可以将集群3和6视为客户喜欢第1、2、5、6、8、9、15和16类产品的一种类型。而集群1、5、7是客户最喜欢的另一种类型

在第1、2、5、9、15类中，第2组最喜欢第10类。而集群4很少买东西，而集群8是最后的集群类型。现在让我们往下钻，看看用户在每个集群中的购买模式。



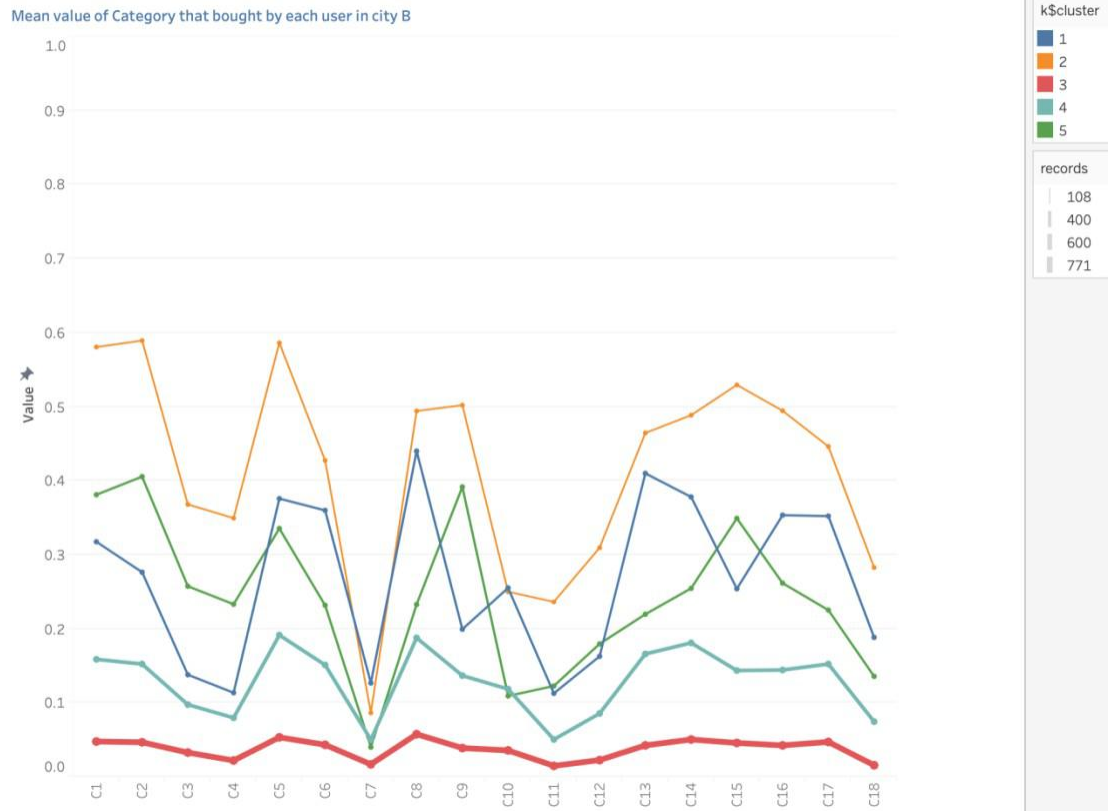
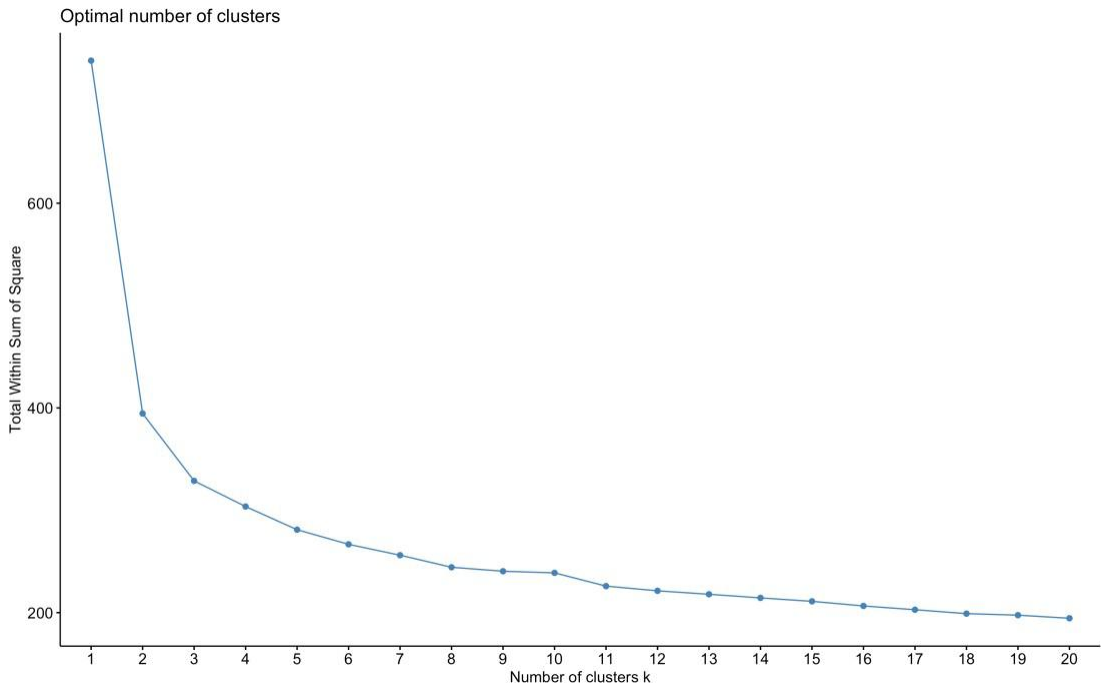




正如我们所看到的，聚类8组的性能优于3组。虽然在一些组中仍然有一点混乱，比如集群3和6。但总的来说，结果是令人满意的。

现在让我们看看B市 同样，我们还绘制了一个WSS和集群图像，以决定我们要集群多少组。从情节来看，很难说出它推荐的集群号。事实上，当数字10转到11时，有一个明显的下降，但11组似乎太多了。

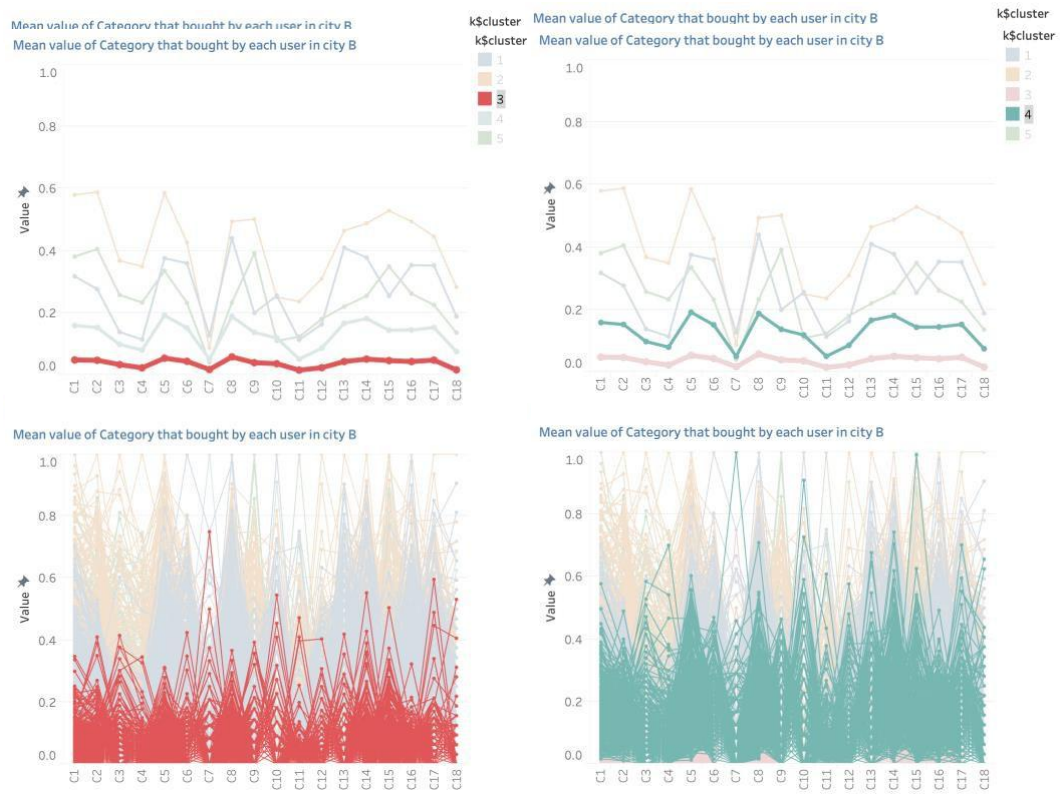
我们观察每两个点之间的线的斜率，并考虑在5个簇处有一个轻微的转弯。我们决定把它们分成5组。

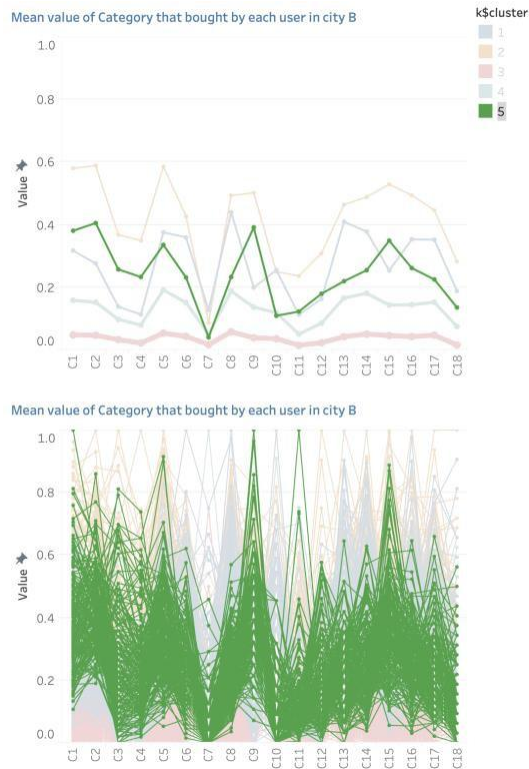


很明显，在B市，集群3的人数是最大的，他们中的大多数人在黑色星期五不太购买。集群4也有很多人，他们更喜欢1、2、5、8、13和14类的产品。第一类的人喜欢购买第一类、第五类、第六类、第八类、第十三类、第十四类的产品  
16和17 此外，集群5的人更喜欢1、2、5、9和15类产品。为了人民  
集群2的所有产品都有很强的购买力，特别是在第1、2、5、8、9、13、14类产品中，



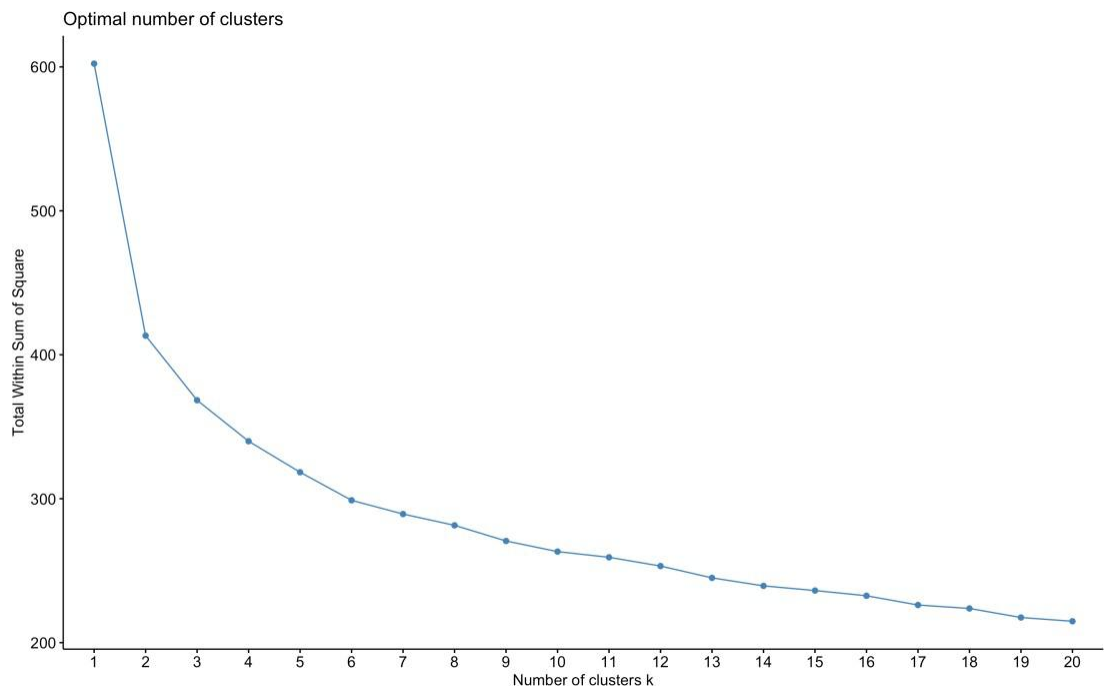
15、16和17。一件事应该注意的是，有一些类别，几乎所有的客户群体都喜欢。也许这些是必需品，也许不是。只有零售店才会知道。我们钻入每个集群来估计性能。

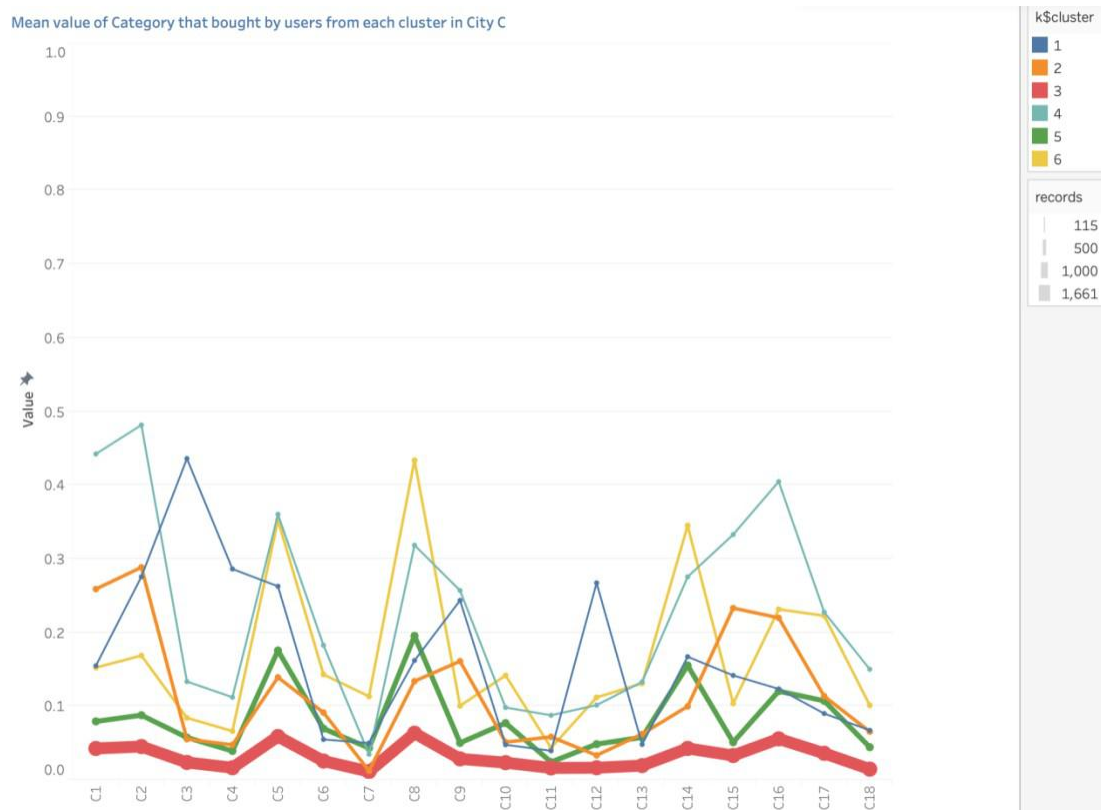




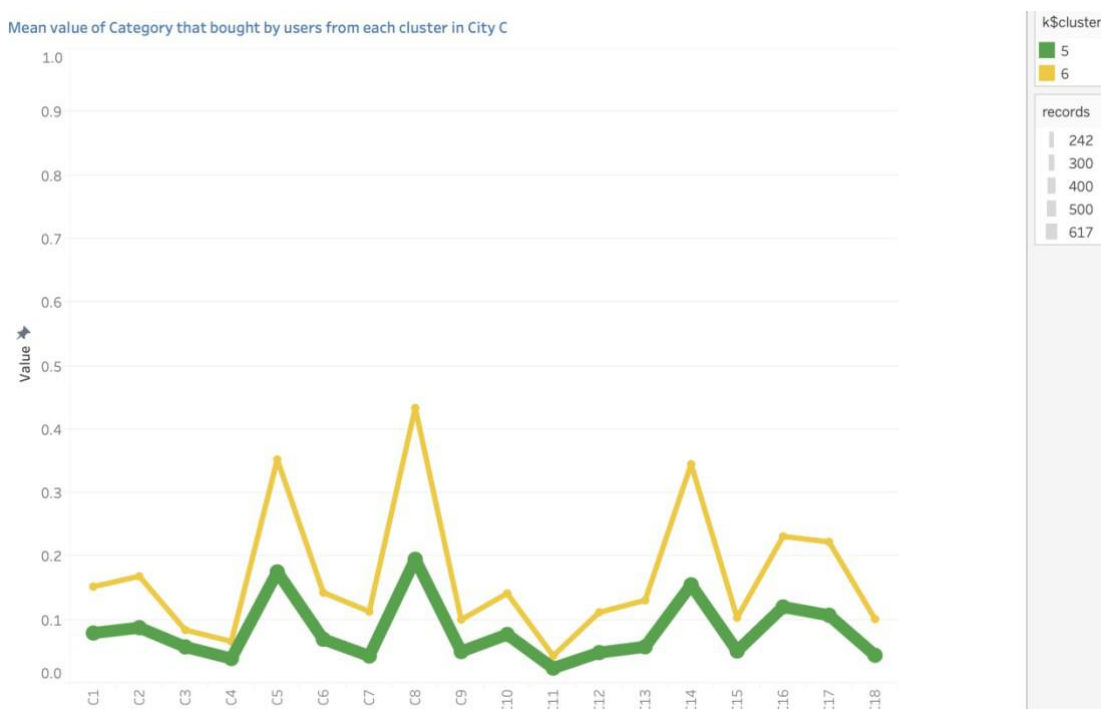
虽然里面还有一些集群看起来有点乱。点分布密集，大部分局限在较小范围内.. 所以我们认为去是好事。

最后我们在C市关注。根据EDA的讨论，C市的客户数量最多，但购买量较低，C市的集群结果是什么？再次，我们绘制WSS和簇线，并选择6作为簇号k。



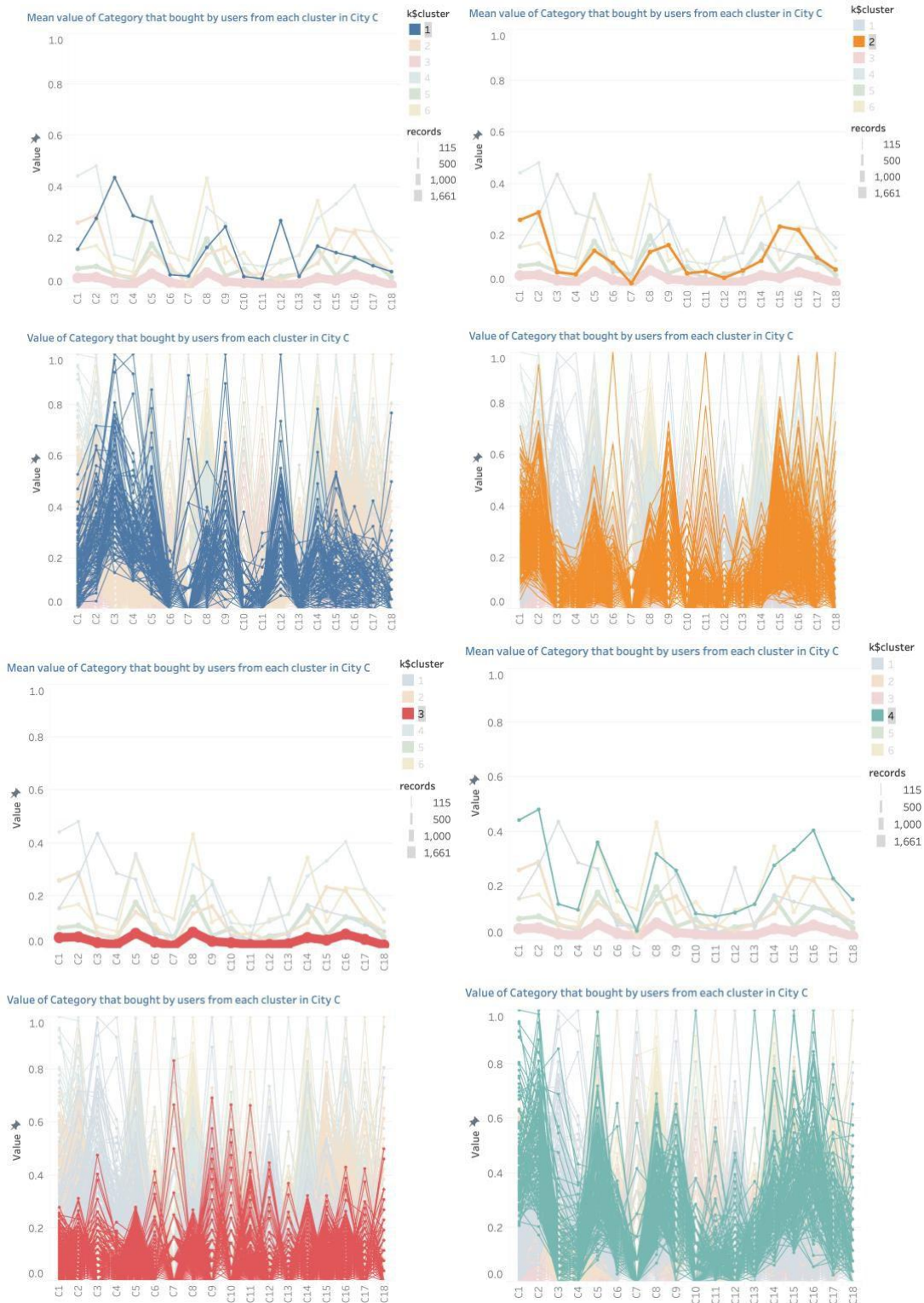


图中显示了每个类的平均水平，似乎集群5和集群6可以被认为是一组。让我们集中讨论集群5和6。



这两个集群中的人都喜欢购买5、8和14的产品。因此，我们认为他们是同一类型的客户。对于其他群体，第1组的人优先购买第3、4、5、9和12类产品，而第2组的人更喜欢购买第1、2、5、9、15和16类产品。此外，有大量的人属于集群3，他们很少购买东西。集群4中的用户对1、2、5、8、15和16类产品感兴趣。还有

我们的集群结果准确吗？







与城市A和B的情况相比，城市C有更多的人，所以向下钻取的图像有更多的线，这使得一些集群内部的一些类别的值很宽，比如集群6 但总的来说，集群还是有道理的..

既然我们已经将客户聚集成组，经理就可以有针对性地对他们实施不同的营销策略。

## (2) 捆绑销售

数据转化后，适合使用.. 因为我们想找出什么产品

可能是一起销售的，我们将计算所有可能的产品组合的销售数量。为了简化问题，在这个项目中，我们只考虑任意2种产品的组合。如果我们根据不同的城市来分析这个问题，那就更有意义了。所以我们首先关注A市。

```
# Calculate purchase of all combinations of 2 products in A
pair_product_A<-data.frame(matrix(0,0,length(all_product_A)))
names(pair_product_A)<-all_product_A
for (i in all_product_A) {
  purchase<-as.data.frame(t(colSums(product_A_transformed[c(9:length(product_A_transformed))][product_A_transformed[[i]]==1,]))
  row.names(purchase)<-i
  pair_product_A<-rbind(pair_product_A,purchase)
}
```

该算法创建一个3470x3470数据帧，以显示每个产品组合的销售数量。这意味着在BF期间，A市有3470种不同的产品被销售。下面是数据框架的一部分。

	P00069042	P00248942	P00087842	P00085442	P00193542	P00274942	P00251242
P00069042	63	19	4	14	15	19	20
P00248942	19	158	7	18	44	50	58
P00087842	4	7	22	14	7	3	9
P00085442	14	18	14	79	26	19	28
P00193542	15	44	7	26	159	53	64
P00274942	19	50	3	19	53	196	80
P00251242	20	58	9	28	64	80	248

作为一个例子，让我们看看[1,1]和[1,2]的数字。说明A市有63个客户购买了产品P00069042，有19人不仅购买了P00069042还购买了P00248942.. 它是一个对称矩阵。

但对于经理来说，他/她可能希望专注于那些销售最多的组合。因此，我们使用以下算法，这可能不是最好的方法，以获得具有最多销售的组合。我们首先删除对角线元素，然后对其他元素进行排序。

```
# Orderd top sales combinations in A
pair_product_A_diag<-pair_product_A
# drop element in diagonal (combinations of itself)
for (i in 1:nrow(pair_product_A_diag)){pair_product_A_diag[i,i]<-0}
# find the top 100 sales number (The number is 200 because in the matrix, each element is counted twice)
top_sells_A<-sort(as.matrix(pair_product_A_diag),decreasing = TRUE)[seq(1,200,2)]

> top_sells_A
[1] 157 151 147 143 135 135 134 134 133 133 133 133 132 132 131
[17] 129 129 128 128 128 128 128 127 127 127 126 125 125 125 125
[33] 124 123 123 123 123 123 122 122 122 121 121 121 121 121 121
[49] 121 121 121 120 120 120 120 120 120 119 119 119 119 119 118
[65] 118 118 118 118 118 118 118 118 117 117 117 117 117 116 116
[81] 116 116 116 115 115 115 115 114 114 114 114 114 114 114 114
[97] 114 113 113 113
```

现在我们已经有了结果，我们还需要找出每个组合中的产品。此外，我们也可能想知道这两种产品的销售号码，因为如果你想捆绑两个项目一起销售，你应该考虑他们的销售，无论是联合还是单独。因此，下面代码的逻辑是，对于等于top\_sells\_A中数字之一的组合的销售，我们找出了这种组合的产品ID。在知道产品ID后，我们再添加他们的个人频率。

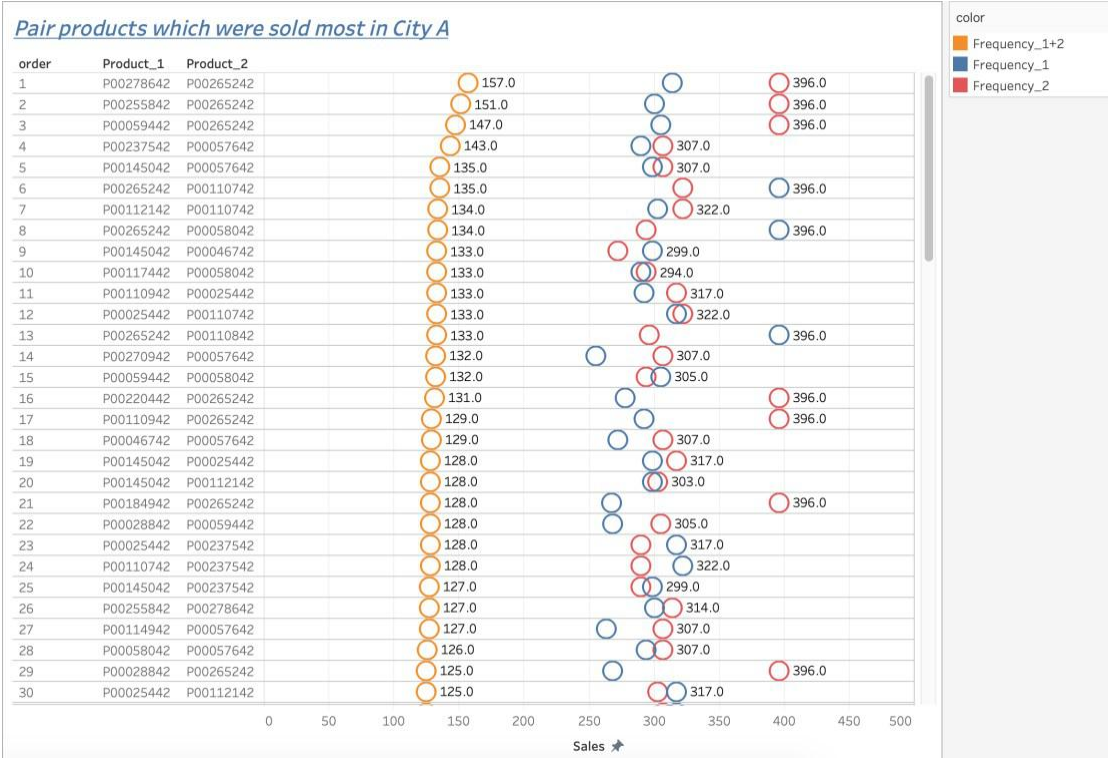
```
# Find out what products are for each top combinations
top_pairs_ID_A<-data.frame(matrix(0,0,2))
top_pairs_count_A<-data.frame(matrix(0,0,1))
for (j in 1:(nrow(pair_product_A_diag)-1)) {
  for (k in (j+1):nrow(pair_product_A_diag)) {
    if (pair_product_A_diag[j][k,] %in% top_sells_A) {
      top_pairs_ID_A<-rbind(top_pairs_ID_A,c(names(pair_product_A_diag)[j],row.names(pair_product_A_diag)[k]),stringsAsFactors=FALSE)
      top_pairs_count_A<-rbind(top_pairs_count_A,pair_product_A_diag[j][k,],stringsAsFactors=FALSE)
    }
  }
}
top_pairs_A<-data.frame(top_pairs_count_A,top_pairs_ID_A)
names(top_pairs_A)<-c('Frequency','Product_1','Product_2')
top_pairs_oredered_A<-top_pairs_A[order(top_pairs_A$Frequency,decreasing = TRUE),]

# Enrich top pairs dataset with their separete frequency
frequency1<-data.frame(f1=matrix(0,0,1))
frequency2<-data.frame(f2=matrix(0,0,1))
for (i in 1:nrow(top_pairs_oredered_A)){
  p1<-top_pairs_oredered_A[i,2]
  p2<-top_pairs_oredered_A[i,3]
  frequency1<-rbind(frequency1,pair_product_A[p1,p1])
  frequency2<-rbind(frequency2,pair_product_A[p2,p2])
}
bundle_A<-cbind(top_pairs_oredered_A,frequency1,frequency2)
names(bundle_A)[4:5]<-c('Frequency_1','Frequency_2')
```

现在我们有完整的结果数据框架。第一列频率是product\_1和product\_2组合的销售数量。它显示了不仅购买product\_1而且product\_2的客户数量。Frequency\_1和Frequency\_2代表个人

product\_1和product\_2的销售。 例如， 第一行显示314人购买P00278642， 396人购买P00265242， 157人购买P00278642和P00265242。 而不是仅仅给出这个数据框架， 我们使用

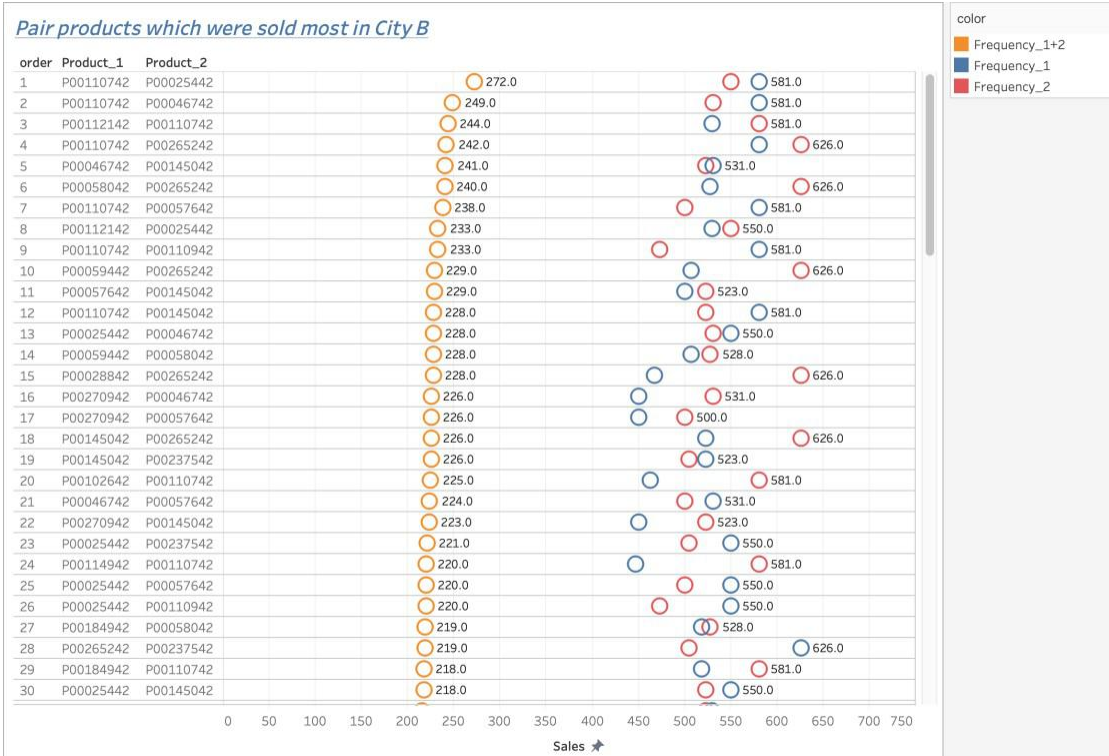
Frequency	Product_1	Product_2	Frequency_1	Frequency_2
157	P00278642	P00265242	314	396
151	P00255842	P00265242	300	396
147	P00059442	P00265242	305	396
143	P00237542	P00057642	290	307
135	P00145042	P00057642	299	307
135	P00265242	P00110742	396	322
134	P00112142	P00110742	303	322
134	P00265242	P00058042	396	294
133	P00145042	P00046742	299	272
133	P00117442	P00058042	290	294



更漂亮的方式来显示结果。

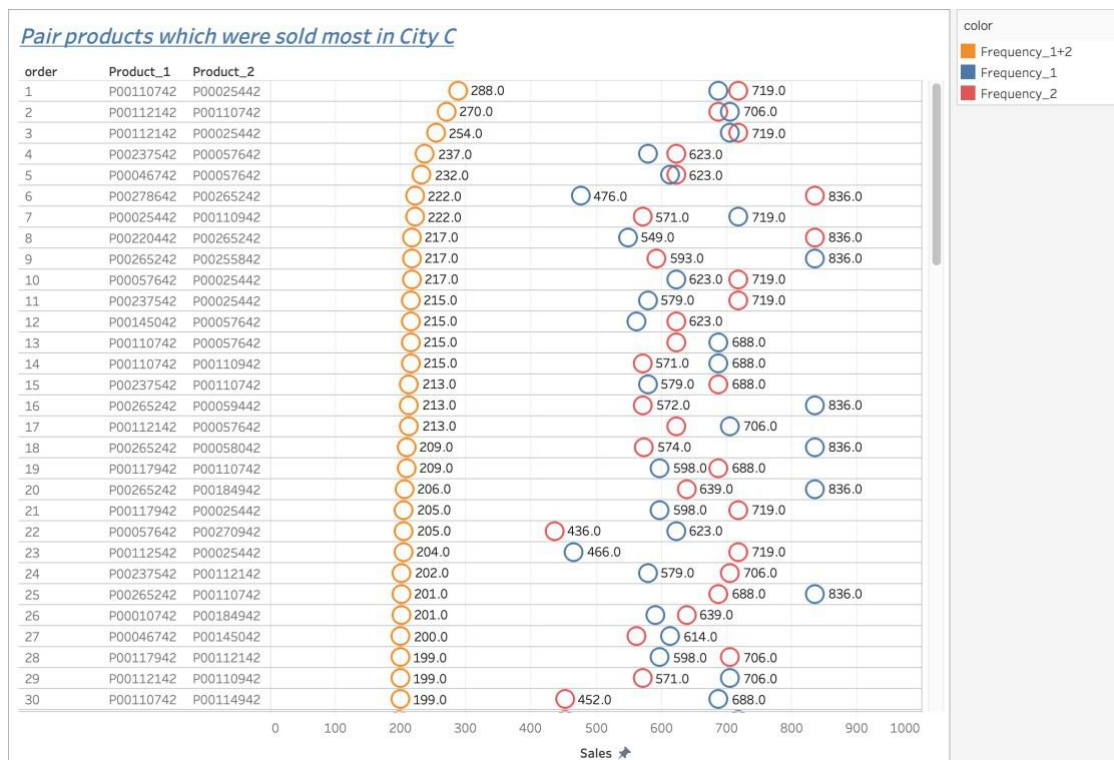
现在我们完成了城市A，然后我们遵循相同的过程，基于城市B和C.以下是城市B和C的结果。

Frequency	Product_1	Product_2	Frequency_1	Frequency_2
272	P00110742	P00025442	581	550
249	P00110742	P00046742	581	531
244	P00112142	P00110742	530	581
242	P00110742	P00265242	581	626
241	P00046742	P00145042	531	523
240	P00058042	P00265242	528	626
238	P00110742	P00057642	581	500
233	P00112142	P00025442	530	550
233	P00110742	P00110942	581	473
229	P00059442	P00265242	507	626



Frequency	Product_1	Product_2	Frequency_1	Frequency_2
288	P00110742	P00025442	688	719
270	P00112142	P00110742	706	688
254	P00112142	P00025442	706	719
237	P00237542	P00057642	579	623
232	P00046742	P00057642	614	623
222	P00278642	P00265242	476	836
222	P00025442	P00110942	719	571
217	P00220442	P00265242	549	836
217	P00265242	P00255842	836	593
217	P00057642	P00025442	623	719





在所有这些工作之后，我们想使用一些数学方法，并想知道这是否会提高结果的性能。因此，我们尝试使用相关系数矩阵，而不是简单地总结转换后的数据。这里是相关系

```
##### Try to use correlation coefficient matrix
corelation_A<-cor(product_A_transformed[9:3478])
```

	P00069042	P00248942	P00087842	P00085442	P00193542
P00069042	1.0000000000	0.106331683	0.074877481	0.140486366	0.060606821
P00248942	0.1063316832	1.000000000	0.068356309	0.061189092	0.148445449
P00087842	0.0748774810	0.068356309	1.000000000	0.311077808	0.067788743
P00085442	0.1404863660	0.061189092	0.311077808	1.000000000	0.140897393
P00193542	0.0606068205	0.148445449	0.067788743	0.140897393	1.000000000

数矩阵的一部分。

下一步基本上是做和以前一样的事情。我们使用这个相关系数矩阵在相同的过程中生成结果。

```

corelation_A<-as.data.frame(corelation_A)
# basically the same as before
corelation_A_diag<-corelation_A
# drop element in diagonal (combinations of itself)
for (i in 1:nrow(corelation_A_diag)){corelation_A_diag[i,i]<-0}
# find the top 25 sales number (The number is 50 because in the matrix, each element is counted twice)
top_sells_A_co<-sort(as.matrix(corelation_A_diag),decreasing = TRUE)[seq(1,50,2)]

# Find out what products are for each top combinations
top_pairs_ID_A_co<-data.frame(matrix(0,0,2))
top_pairs_count_A_co<-data.frame(matrix(0,0,1))
for (j in 1:(nrow(corelation_A_diag)-1)) {
  for (k in (j+1):nrow(corelation_A_diag)) {
    if (corelation_A_diag[j][k,] %in% top_sells_A_co) {
      top_pairs_ID_A_co<-rbind(top_pairs_ID_A_co,c(names(corelation_A_diag)[j],row.names(corelation_A_diag)[k]),stringsAsFactors=FALSE)
      top_pairs_count_A_co<-rbind(top_pairs_count_A_co,corelation_A_diag[j][k,],stringsAsFactors=FALSE)
    }
  }
}
top_pairs_A_co<-data.frame(top_pairs_count_A_co,top_pairs_ID_A_co)
names(top_pairs_A_co)<-c('Frequency','Product_1','Product_2')
top_pairs_oredered_A_co<-top_pairs_A_co[order(top_pairs_A_co$Frequency,decreasing = TRUE),]

# Enrich top pairs dataset with their separete frequency
frequency1<-data.frame(f1=matrix(0,0,1))
frequency2<-data.frame(f2=matrix(0,0,1))
for (i in 1:nrow(top_pairs_oredered_A_co)){
  p1<-top_pairs_oredered_A_co[i,2]
  p2<-top_pairs_oredered_A_co[i,3]
  frequency1<-rbind(frequency1,corelation_A[p1,p1])
  frequency2<-rbind(frequency2,corelation_A[p2,p2])
}
bundle_A_co<-cbind(top_pairs_oredered_A_co,frequency1,frequency2)
names(bundle_A_co)[4:5]<-c('Frequency_1','Frequency_2')

```

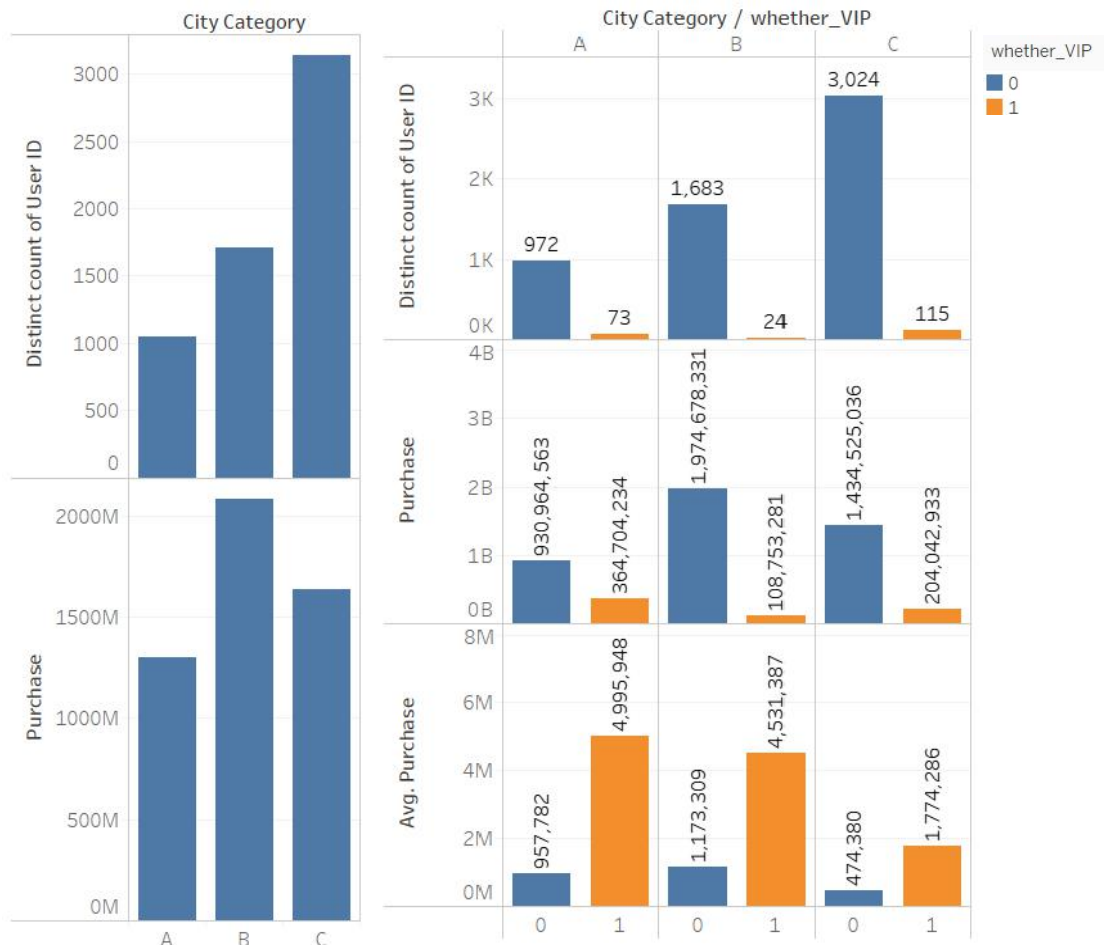
Frequency	Product_1	Product_2	Frequency_1	Frequency_2
1	P00065842	P00250142	1	1
1	P00255642	P00025842	1	1
1	P00051342	P00142042	1	1
1	P00070842	P00161742	1	1
1	P00005842	P00072142	1	1
1	P00005842	P00231042	1	1
1	P00005842	P00069742	1	1
1	P00005842	P00140342	1	1
1	P00005842	P00326342	1	1

很奇怪。有这个结果的原因是许多产品只买了一次。这使得它们的相关系数等于1。显然我们不想要这个结果。因此，我们决定不实现相关系数矩阵。

## 6. 讲故事

商业活动的每一个最终目标通常是赚更多的钱。这个项目也是。我们的想法是使用以数据为中心的方法来增加零售店的销售额。而这些数据过于匿名，分析结果只能对零售店有用。

首先，我们在EDA中发现了一个有趣的现象。



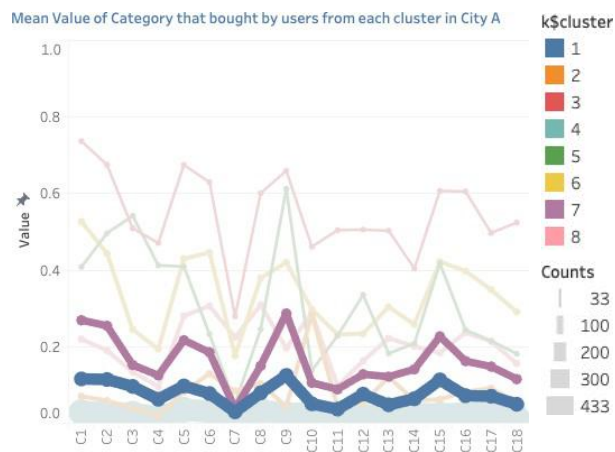
在所有这些交易中，C市拥有最多的客户，几乎是B市的两倍 但它的购买金额不是最高的。造成这种情况的原因是，一方面，C市的人们平均没有购买很多商品。相反，他们的购买金额最低。 另一方面，没有顾客在C市拥有极强的购买能力。虽然C市拥有最多的VIP，但他们只是C市的VIP。正如我们在直方图中所看到的，它们与A市或B市的VIP是不可比拟的。

为了实现我们的目标，我们认为也许我们可以把客户分成几组，并对他们实施不同的营销策略。

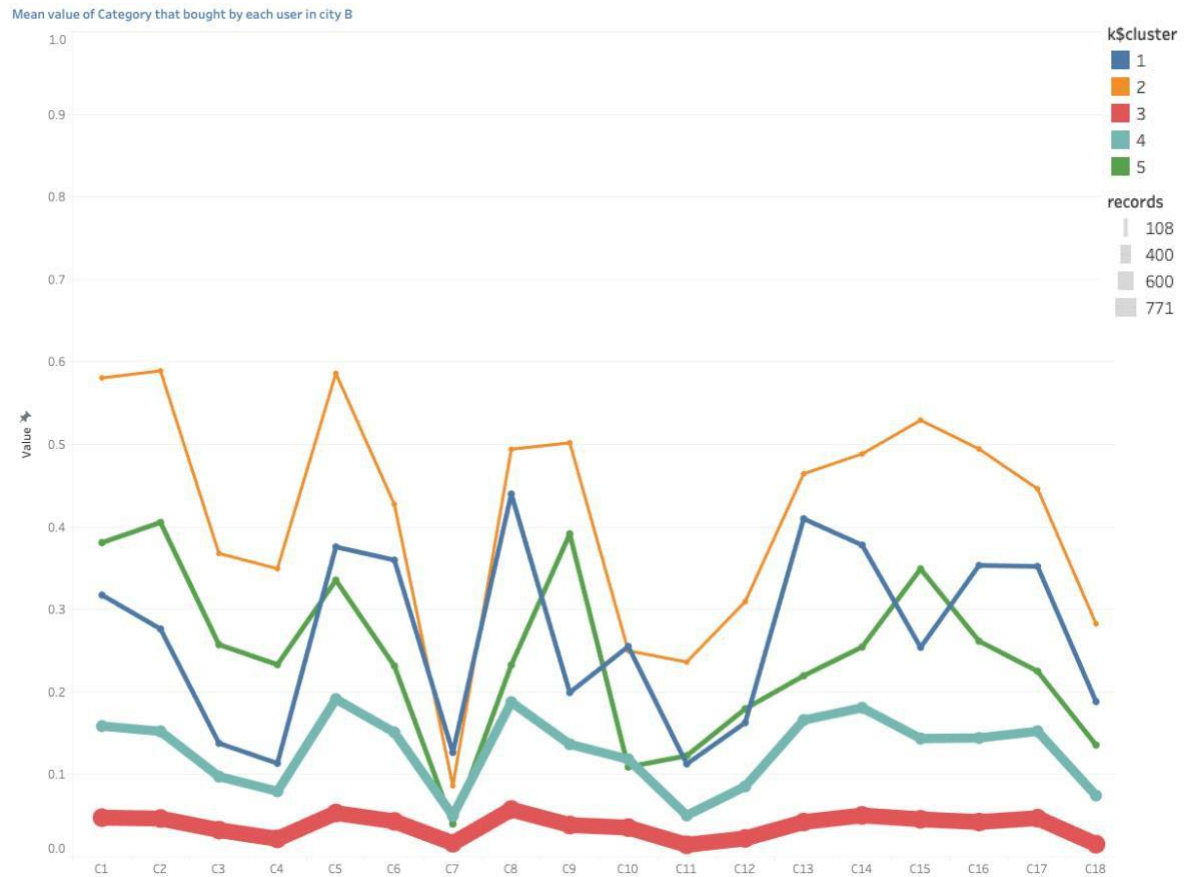
对于A市，最后我们考虑有6种类型的客户。每个图像代表一组客户。第一张图片（我们也可以称之为第一组）喜欢第1、2、5、6、8、9、15和16类产品。第二组人与第一组人有共同之处

组，但他们持有不同的分布，即类别1, 2, 5, 9, 15, 不包括6, 8和16从第一个。第三类对第10类有浓厚的兴趣，而第四类基本上什么也不买。第五组与第二组也略有不同，优先考虑第1、2、3、4、5、9和15类。最后一组喜欢5、6、8和10。此外，注意到每一行的宽度，大多数人很少购买。





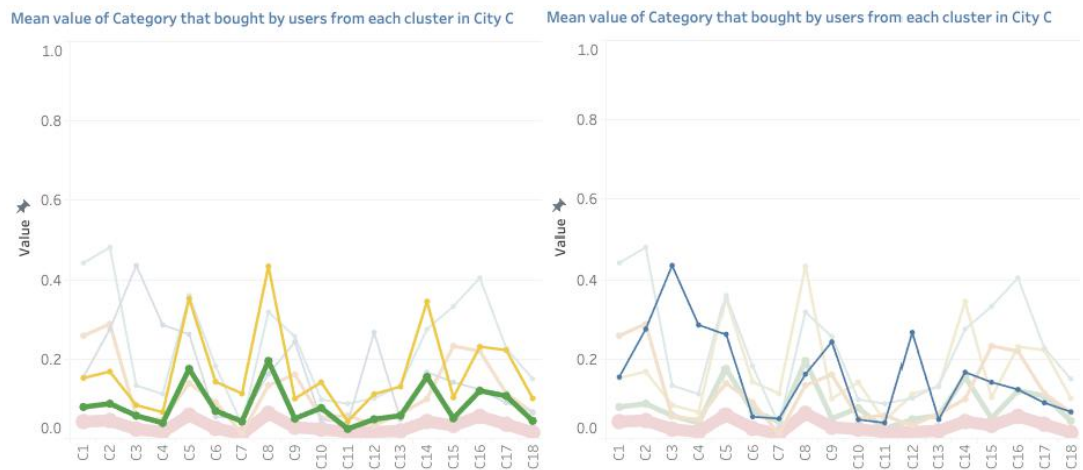
至于B市，我们已经产生了5组客户。但老实说，这些群体没有明显的模式。



然而，B市的大多数人买的东西很少。似乎第1、第2、第5、第6、第8、第9、第13、第14、第15、第16和第17类在B市的所有客户中都很受欢迎，但第5组的客户除外。

C市似乎是一个不寻常的城市，因为它的怪异表现我们之前谈到过。

它的集群结果会是什么样子？

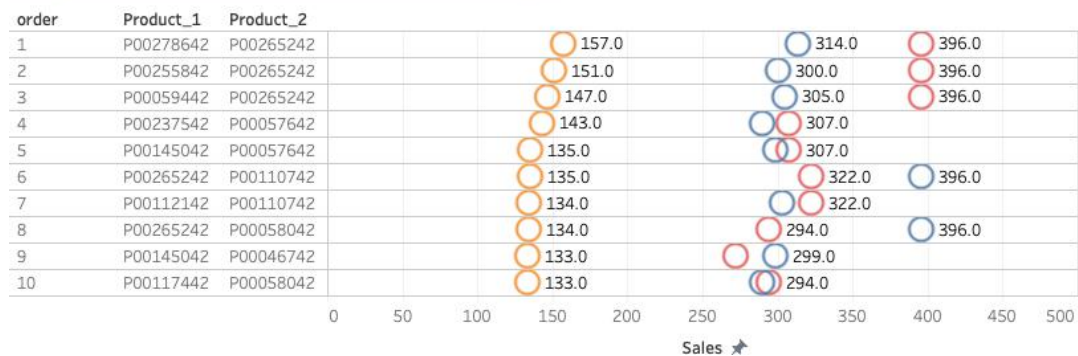




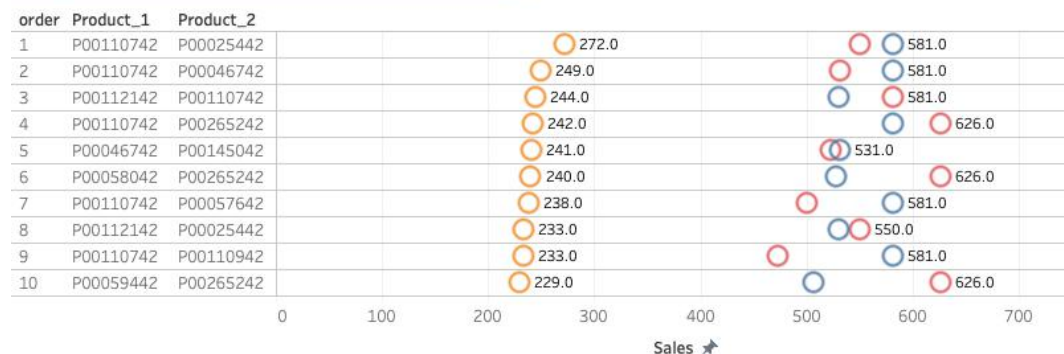
城市C的人被分成4组。一个常见的情况是，所有三个城市的大多数人都很少购买产品，因为这里的第4组线也有最大的宽度。因此，第一组购买了许多属于第5、8和14类的产品。相比之下，第二组更喜欢第三类。最后一组的人喜欢第一类、第二类、第五类、第八类、第九类、第十五类和第十六类，这似乎也是我们在A市看到的一种常见的客户类型。

在产品方面，我们为每个城市生产一个结果，说明哪些产品更有可能以捆绑销售。我们假设经理理想考虑捆绑销售的项目，这是流行的。因为如果没有人似乎购买该产品，无论它将与哪个产品捆绑，它是无用的。基于这一假设，我们为每个城市得出了一些结论。

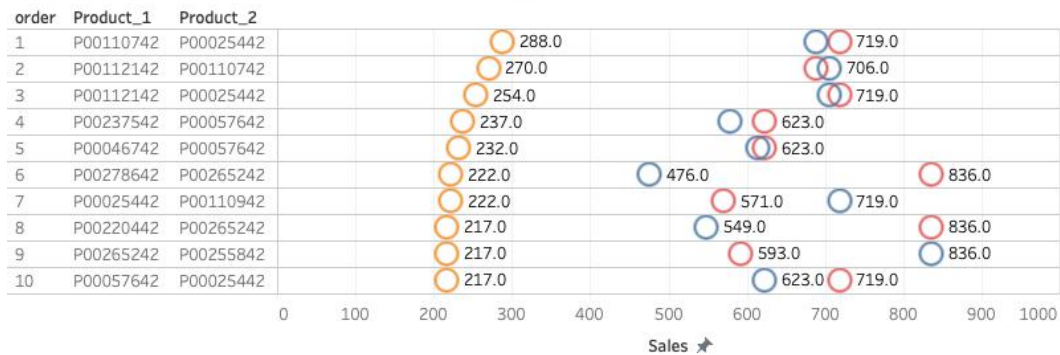
### Pair products which were sold most in City A



### Pair products which were sold most in City B



### Pair products which were sold most in City C



以上是最推荐实施捆绑销售的10对产品..但这个问题在实践中并不简单。我们还应该考虑库存情况，这些产品的单独销售等。我们只是给出分析的结果，经理最终会做出决定。我们也可以意识到，这些推荐的产品在三个城市中相当普遍。

## 7. 回应同行反馈.

在同行反馈中存在一些主要问题，我们将分别对它们作出反应。感谢所有给我们建议的同学。

### 7.1. 集群研究不够深入..

实际上，在我们的集群完成后，我们有了一些进一步的分析，表明我们的集群是合理的。我们认为了解客户的购买类别足以提高销售。没有必要像探索每个集群中的客户属性那样深入下去，因为零售商店很少能根据这些信息做事情。

### 7.2. 我们应该找到更多的数据来找出所有类别的类型。

我们找不到更多与这个数据集相关的数据，因为这个数据太匿名了。这些数据没有告诉我们它是什么零售店，这些数据来自哪里，以及这些类别是什么。所以我们什么也做不了。我们试过了，但没有解决这个问题。对于那些说我们应该寻找一个更可靠的来源的人，我们担心我们没有时间重新开始。

### 7.3. 数据应先归一化..

是的，我们在演讲后做了正常化。谢谢你的建议。

### 7.4. 我们应该用一些方法来找到本体簇数。

我们确实使用了一种方法来计算每个用户之间的距离。但它建议我们只做3个集群。我们认为这是没有意义的，因为这只是根据用户购买的数量将用户分成几组，我们希望集群数量足够大，以显示类别上的差异。

### 7.5. 对于捆绑销售产品，我们应该使用一个特定的数学公式，如相关系数矩阵。

起初，我们认为这是一个好主意，我们应该用一些具体的数学方法来分析。因此，我们尝试计算相关系数矩阵。但是我们意识到使用相关系数矩阵可能是不合适的。看看什么是部分使数据坦白。

### 7.6. 为什么C市没有贵宾？

实际上我们做错了。我们应该在把城市分成3组后找到VIP。更多

准确地说，他们不是真正的贵宾。我们只是认为他们是VIP，因为他们买的比普通用户多。我们用方格图对它们进行分类。

#### **7.7. 为什么职业不包括在集群中？**

因为我们认为零售店可以做的是基于商品。了解职业可能不会导致销售的改善。这是我们的考虑。

#### **7.8. 可以在分析中添加一个非黑色星期五的数据集。**

好建议！如果这些数据不那么匿名，我们肯定会这样做的。

#### **7.9. 预测人们会购买的产品。**

我们不能想出一个方法来做到这一点。

#### **7.10. 缺乏演示技巧和电源点的缺点。**

谢谢你的建议，下学期我们将提高我们的演示技巧和ppt制作技巧。

#### **7.11. 利用捆绑销售结果进行推荐。**

事实上，结果只是表明这些产品很可能一起销售。在实践中，是否捆绑销售两种产品也取决于它们的销售。我们认为有一个阈值来决定我们是否捆绑这些。因此，最好把这个问题留给经理。

### **8. 附录**

基本上，几乎所有的工作都是在R和Tableau完成的。我们使用R来转换数据、聚类和其他分析。在R.中使用了四个库，“reader”、“ggplot2”、“factiveextra”和“NBClust” 读取器用于数据输入，其他用于绘制WSS图，这是选择簇号k的辅助工具。我们更喜欢在Tableau上绘制图像，因为它更方便和强大。我们甚至从R输出结果，并使用Tableau可视化它。