E-Commerce Data Analytics

GarimaSood

February 5, 2018

```
library(ggplot2)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 3.4.4
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(plotly)
```

```
## Warning: package 'plotly' was built under R version 3.4.4
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
dataPath <- "C:/Users/garim/Documents/New Folder"
data <- read.csv(paste(dataPath,"data_science_analytics_2018_data.csv", sep = "/"))

#Preliminary data check

str(data)
```

```
## 'data.frame':    541909 obs. of  8 variables:
##  $ InvoiceNo  : Factor w/ 25900 levels "536365","536366",..: 25900 25900 25899 25898 25897 25
897 25896 25895 25894 25894 ...
##  $ StockCode  : Factor w/ 4070 levels "10002","10080",..: 310 3385 484 4067 1129 2039 2507 19
83 520 1056 ...
##  $ Description: Factor w/ 4224 levels "","? 4 PURPLE FLOCK DINNER CANDLES",..: 85 1712 3845 21
51 3844 4220 2510 2869 3527 578 ...
##  $ Quantity   : int  -5 -1 -5 -1 -12 -11 -80995 -4 -10 -12 ...
##  $ InvoiceDate: Factor w/ 23260 levels "1/10/2011 10:04",..: 9379 9379 9378 9369 9405 9405 94
00 9201 9200 9200 ...
##  $ UnitPrice  : num  1.25 1.25 10.95 224.69 1.95 ...
##  $ CustomerID : int  17315 17315 15311 15498 14397 14397 16446 17924 13599 13599 ...
##  $ Country    : Factor w/ 38 levels "Australia","Austria",..: 36 36 36 36 36 36 36 36 36 36
...
```

```
summary(data)
```

```
##    InvoiceNo        StockCode
##  573585 :  1114   85123A :  2313
##  581219 :   749   22423  :  2203
##  581492 :   731   85099B :  2159
##  580729 :   721   47566  :  1727
##  558475 :   705   20725  :  1639
##  579777 :   687   84879  :  1502
##  (Other):537202   (Other):530366
##                             Description        Quantity
##  WHITE HANGING HEART T-LIGHT HOLDER:  2369   Min.   :-80995.00
##  REGENCY CAKESTAND 3 TIER          :  2200   1st Qu.:     1.00
##  JUMBO BAG RED RETROSPOT           :  2159   Median :     3.00
##  PARTY BUNTING                     :  1727   Mean   :     9.55
##  LUNCH BAG RED RETROSPOT           :  1638   3rd Qu.:    10.00
##  ASSORTED COLOUR BIRD ORNAMENT     :  1501   Max.   : 80995.00
##  (Other)                           :530315
##            InvoiceDate       UnitPrice           CustomerID
##  10/31/2011 14:41:  1114   Min.   :-11062.06   Min.   :12346
##  12/8/2011 9:28  :   749   1st Qu.:     1.25   1st Qu.:13953
##  12/9/2011 10:03 :   731   Median :     2.08   Median :15152
##  12/5/2011 17:24 :   721   Mean   :     4.61   Mean   :15288
##  6/29/2011 15:58 :   705   3rd Qu.:     4.13   3rd Qu.:16791
##  11/30/2011 15:13:   687   Max.   : 38970.00   Max.   :18287
##  (Other)         :537202                       NA's   :135080
##            Country
##  United Kingdom:495478
##  Germany       :  9495
##  France        :  8557
##  EIRE          :  8196
##  Spain         :  2533
##  Netherlands   :  2371
##  (Other)       : 15279
```

```
length(unique(data$StockCode[data$Country=="United Kingdom"]))
```

```
## [1] 4065
```

```
length(unique(data$CustomerID[data$Country=="United Kingdom"]))
```

```
## [1] 3951
```

```
length(unique(data$InvoiceNo[data$Country=="United Kingdom"]))
```

```
## [1] 23494
```

```
data[data$Quantity > 80000 | data$Quantity< -80000,]
```

	InvoiceNo <fctr>	StockCo… <fctr>	Description <fctr>	Quantity <int>	InvoiceDate <fctr>	UnitP <c
7	C581484	23843	PAPER CRAFT , LITTLE BIRDIE	-80995	12/9/2011 9:27	2
10772	581483	23843	PAPER CRAFT , LITTLE BIRDIE	80995	12/9/2011 9:15	2

2 rows | 1-7 of 9 columns

```
data[data$UnitPrice > 35000 | data$UnitPrice< 0,]
```

	InvoiceNo <fctr>	StockCo… <fctr>	Description <fctr>	Quantity <int>	InvoiceDate <fctr>	UnitPrice <dbl>	CustomerID <int
5274	C556445	M	Manual	-1	6/10/2011 15:31	38970.00	1509
9289	A563187	B	Adjust bad debt	1	8/12/2011 14:52	-11062.06	*NA*
9290	A563186	B	Adjust bad debt	1	8/12/2011 14:51	-11062.06	*NA*

3 rows | 1-8 of 9 columns

```
countrysplit <- as.data.frame(table(data$Country))
countrysplit
```

Var1 <fctr>	Freq <int>
Australia	1259
Austria	401
Bahrain	19
Belgium	2069
Brazil	32
Canada	151
Channel Islands	758
Cyprus	622
Czech Republic	30
Denmark	389

1-10 of 38 rows Previous **1** 2 3 4 Next

Cleaning data for noise i.e. negative quantity orders/ adjustments, zero unit price orders and outliers in terms of quantity & price

```
data$cancel <- ifelse(grepl("C", data$InvoiceNo), 1,0)
data$post <- ifelse(data$StockCode =="POST"|data$StockCode=="DOT",1,0)
data$sample <- ifelse(data$StockCode =="S", 1,0)
data$charges <- ifelse(data$StockCode=="BANK CHARGES",1,0)
data$cust_trans <- ifelse(data$sample+data$charges+data$post==0,1,0)

data$sale <- data$Quantity*data$UnitPrice
```

Percentage of money spent on shipping

```
sum(data$sale[data$post==1])/sum(data$sale[data$cust_trans==1])
```

```
## [1] 0.02872555
```

Outliering based on 3 Standard deviations on unit price and quantity fields:

```
sd_quantity <- sqrt(var(data$Quantity))
mean_quantity <- mean(data$Quantity)
UL_quantity <- mean_quantity+3*sd_quantity
LL_quantity <- mean_quantity-3*sd_quantity

sd_UP <- sqrt(var(data$UnitPrice))
mean_UP <- mean(data$UnitPrice)
UL_UP <- mean_UP+3*sd_UP
LL_UP <- mean_UP-3*sd_UP

data_filt <- data[(data$Quantity>LL_quantity),]
data_filt <- data_filt[(data_filt$Quantity<UL_quantity),]
data_filt <- data_filt[(data_filt$UnitPrice<UL_UP),]
data_filt <- data_filt[(data_filt$UnitPrice>LL_UP),]
data_filt$Country1 <- ifelse(data_filt$Country=="United Kingdom", "UK", as.character(data_filt$C
ountry))


min(data_filt$UnitPrice)
```

```
## [1] 0
```

```
data_filt$sale = data_filt$Quantity*data_filt$UnitPrice
total_sale = sum(data_filt$sale)
```

Number of NAs in data

```
lapply(data_filt, function(x) sum(is.na(x)))
```

```
## $InvoiceNo
## [1] 0
##
## $StockCode
## [1] 0
##
## $Description
## [1] 0
##
## $Quantity
## [1] 0
##
## $InvoiceDate
## [1] 0
##
## $UnitPrice
## [1] 0
##
## $CustomerID
## [1] 134743
##
## $Country
## [1] 0
##
## $cancel
## [1] 0
##
## $post
## [1] 0
##
## $sample
## [1] 0
##
## $charges
## [1] 0
##
## $cust_trans
## [1] 0
##
## $sale
## [1] 0
##
## $Country1
## [1] 0
```

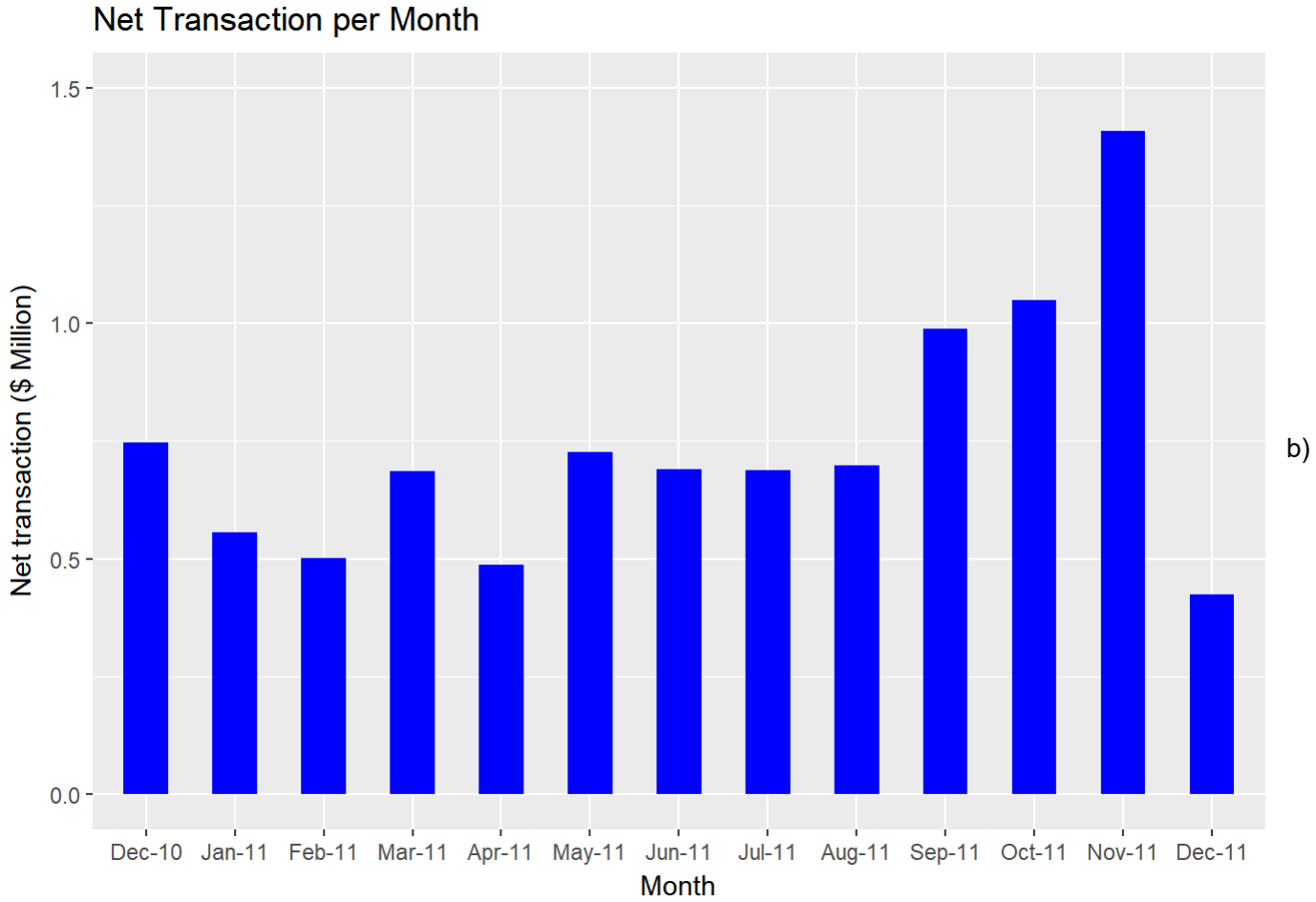
Extract month from datetime

```
data_filt$date = paste(substr(months(as.Date(data_filt$InvoiceDate, format = "%m/%d/%Y")),1,3),s
ubstr(as.Date(data_filt$InvoiceDate, format = "%m/%d/%Y"),3,4),sep = "-")
```

Net Transaction data analysis

a. Transactions by month

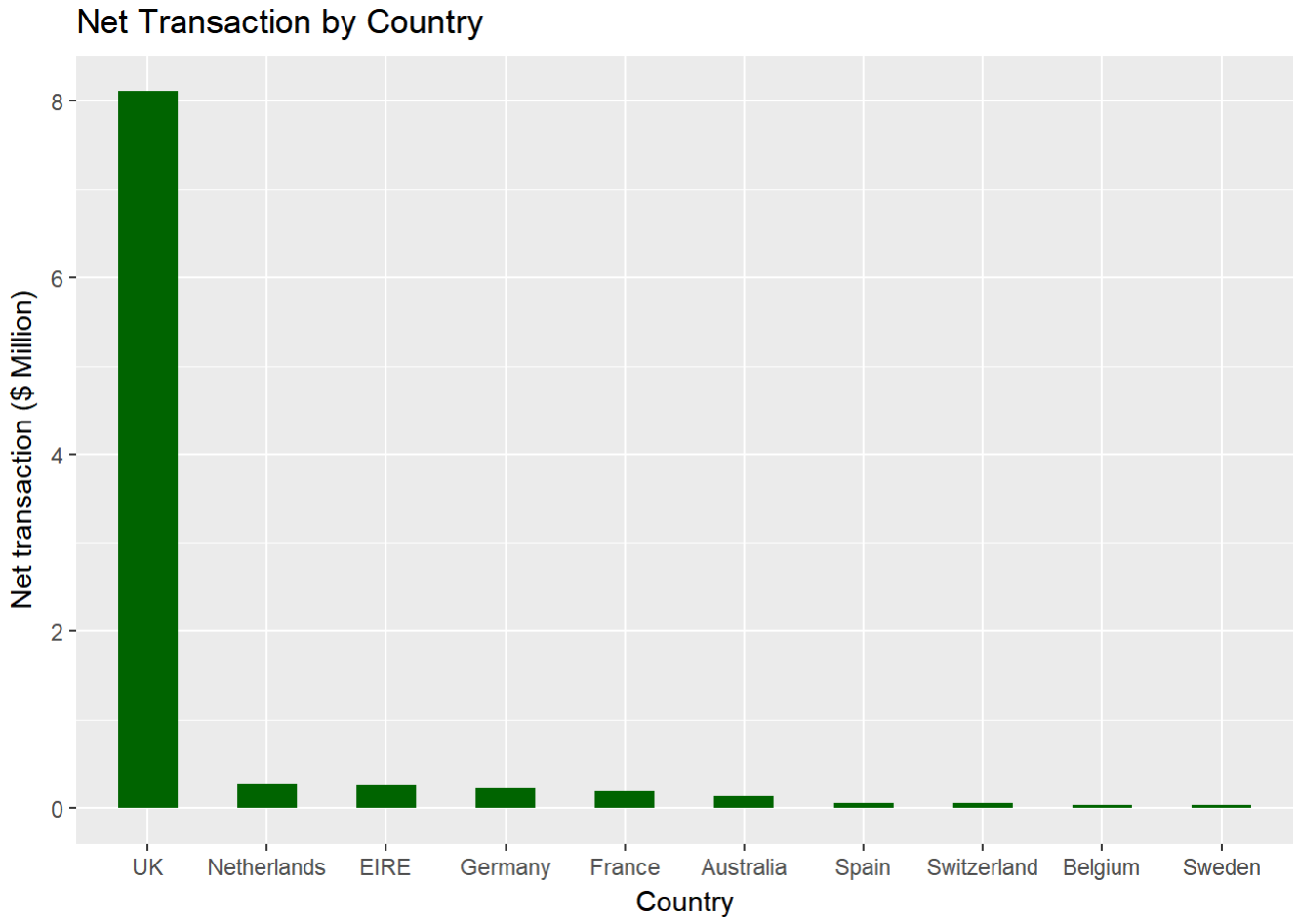
```
trans <- aggregate(data_filt$sale, by = list(data_filt$date),sum)
trans$x <- trans$x/1000000
size_date <- data.frame(table(data_filt$date))
trans_date <- merge(trans,size_date, by.x ="Group.1", by.y= "Var1")


order <- c("Dec-10","Jan-11", "Feb-11", "Mar-11", "Apr-11", "May-11", "Jun-11", "Jul-11", "Aug-1
1", "Sep-11", "Oct-11", "Nov-11", "Dec-11" )

ggplot(data = trans_date) +
  geom_col(mapping = aes(x = Group.1, y = x), width=0.5, fill = "blue")+  ylim(0,1.5) +
  labs(x="Month", y="Net transaction ($ Million)", title = "Net Transaction per Month")+
  scale_x_discrete(limits=order)
```



b)

Transactions by country (Top 10)

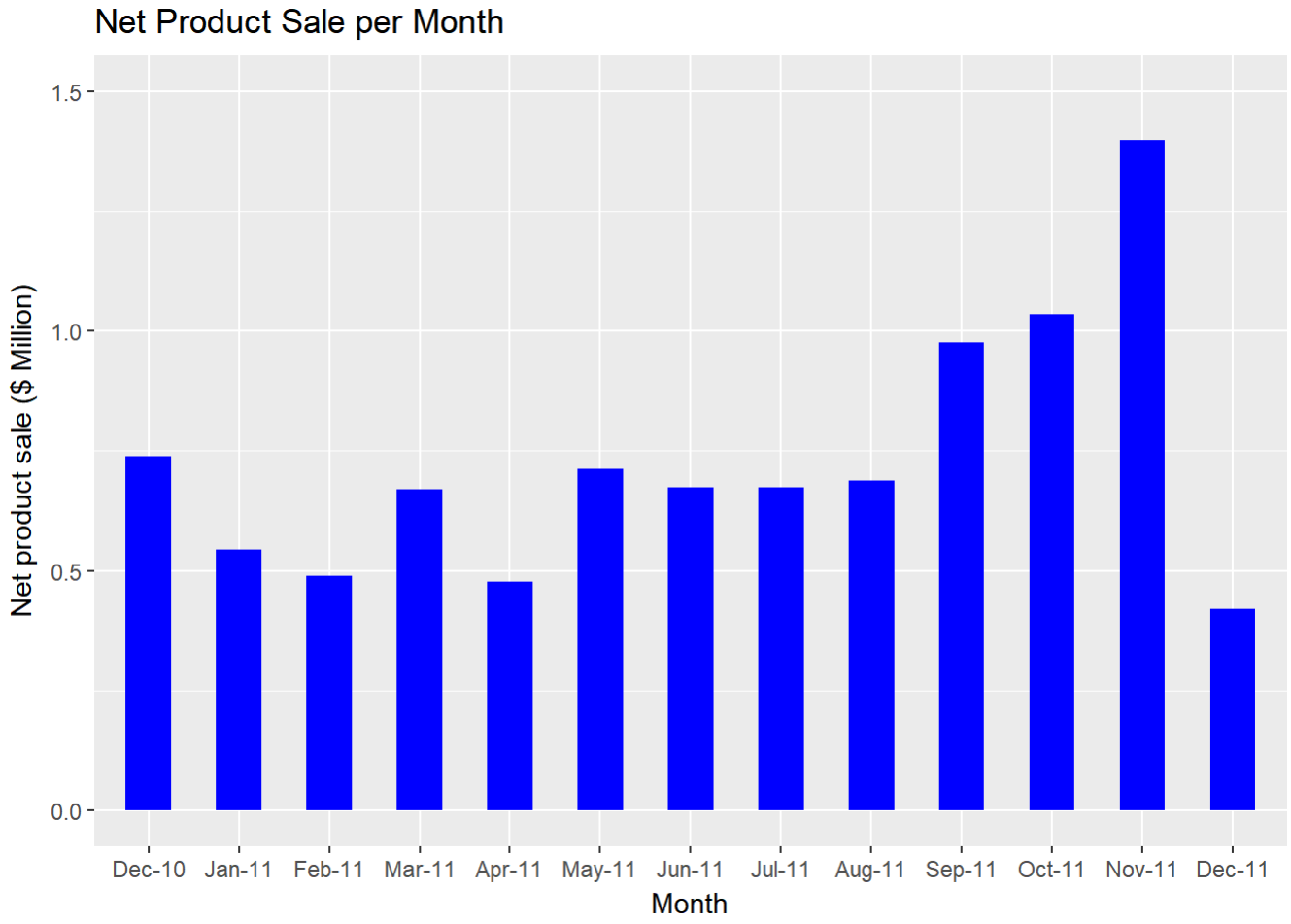
```
trans_country <- aggregate(data_filt$sale, by = list(data_filt$Country1),sum)
trans_country$x = trans_country$x/1000000
size_country <- data.frame(table(data_filt$Country1))
trans_country <- merge(trans_country,size_country, by.x ="Group.1", by.y= "Var1")
trans_country <- trans_country[order(-trans_country$x),]


ggplot(data = trans_country[1:10,]) +
  geom_col(mapping = aes(x = Group.1, y = x), fill = "dark green", width = 0.5)+
  labs(x="Country", y="Net transaction ($ Million)", title = "Net Transaction by Country") +
  scale_x_discrete(limits=trans_country$Group.1[1:10])
```



Sales data analysis (exclude postage, bank charges and free samples)

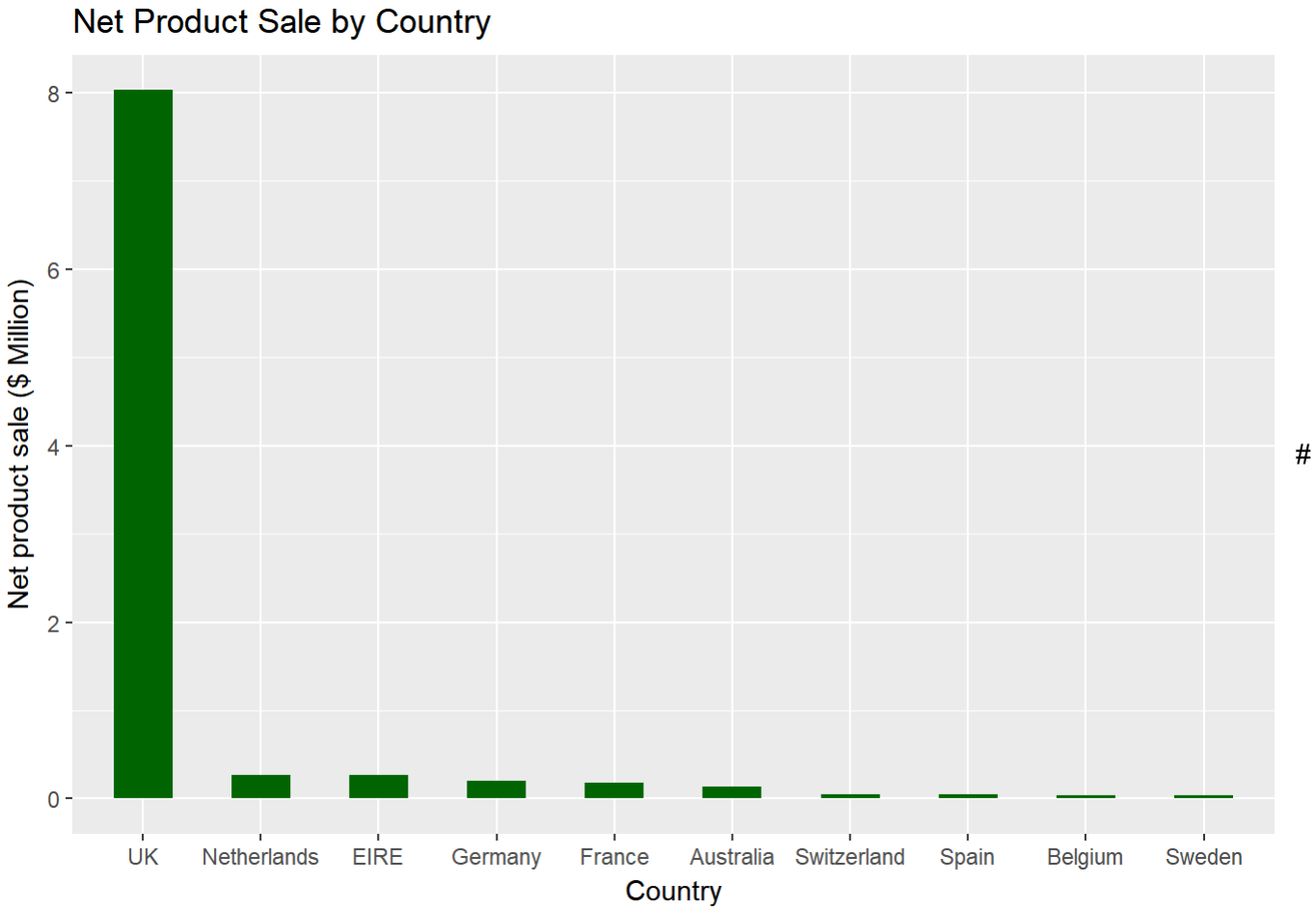
a. Sales by month

```
sales_data <- data_filt[data_filt$cust_trans==1,]

sales <- aggregate(sales_data$sale, by = list(sales_data$date),sum)
sales$x <- sales$x/1000000
size_date <- data.frame(table(sales_data$date))
sales_date <- merge(sales,size_date, by.x ="Group.1", by.y= "Var1")


ggplot(data = sales_date) +
  geom_col(mapping = aes(x = Group.1, y = x), width=0.5, fill = "blue")+  ylim(0,1.5) +
  labs(x="Month", y="Net product sale ($ Million)", title = "Net Product Sale per Month")+
  scale_x_discrete(limits=order)
```



b. Sales by Country

```
sales_country <- aggregate(sales_data$sale, by = list(sales_data$Country1),sum)
sales_country$x <- sales_country$x/1000000
size_country <- data.frame(table(sales_data$Country1))
sales_country <- merge(sales_country,size_country, by.x ="Group.1", by.y= "Var1")
sales_country <- sales_country[order(-sales_country$x),]


ggplot(data = sales_country[1:10,]) +
  geom_col(mapping = aes(x = Group.1, y = x), fill = "dark green", width = 0.5)+
  labs(x="Country", y="Net product sale ($ Million)", title = "Net Product Sale by Country")+
  scale_x_discrete(limits=sales_country$Group.1[1:10])
```



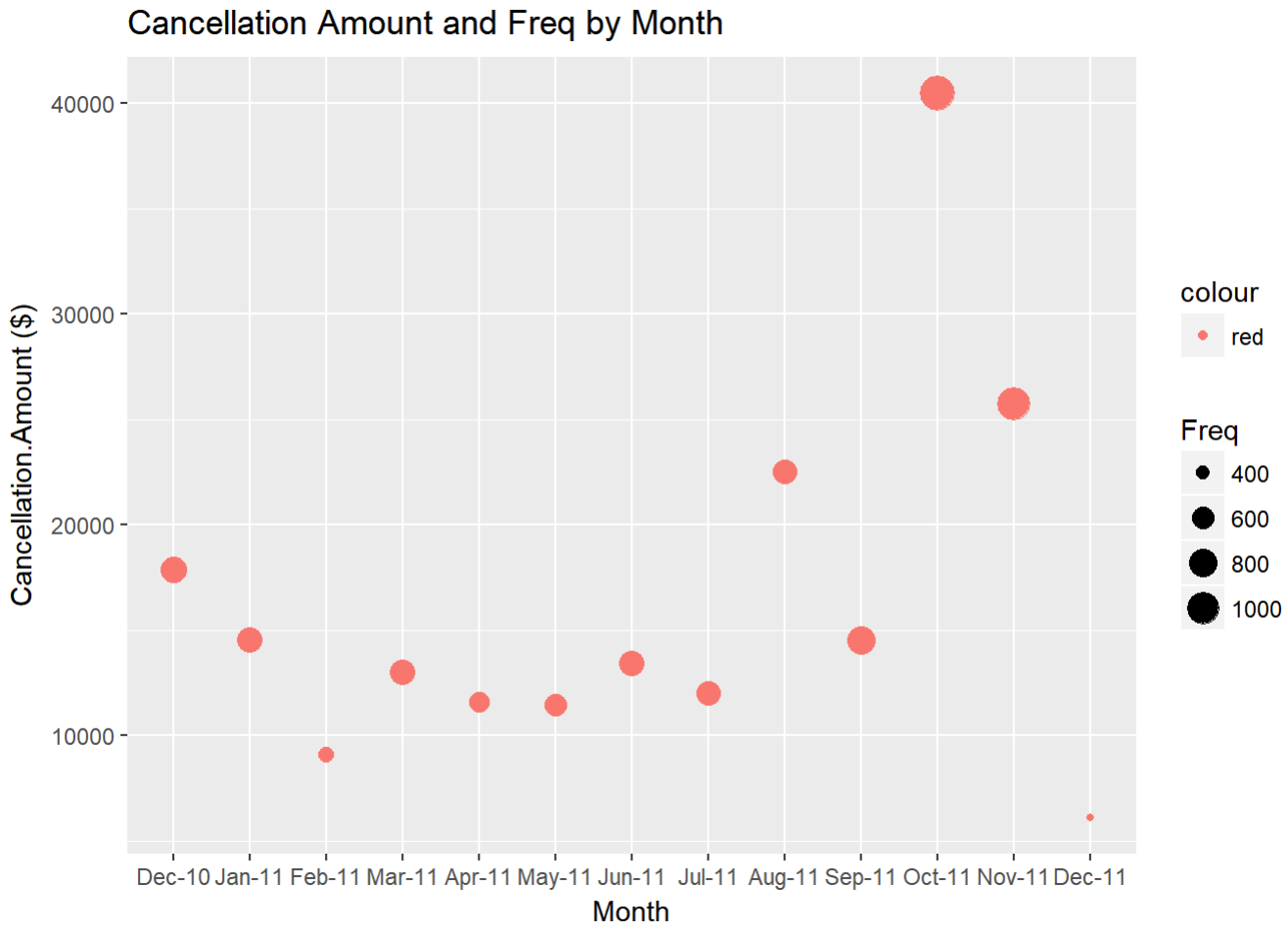
#

Cancellation data analysis

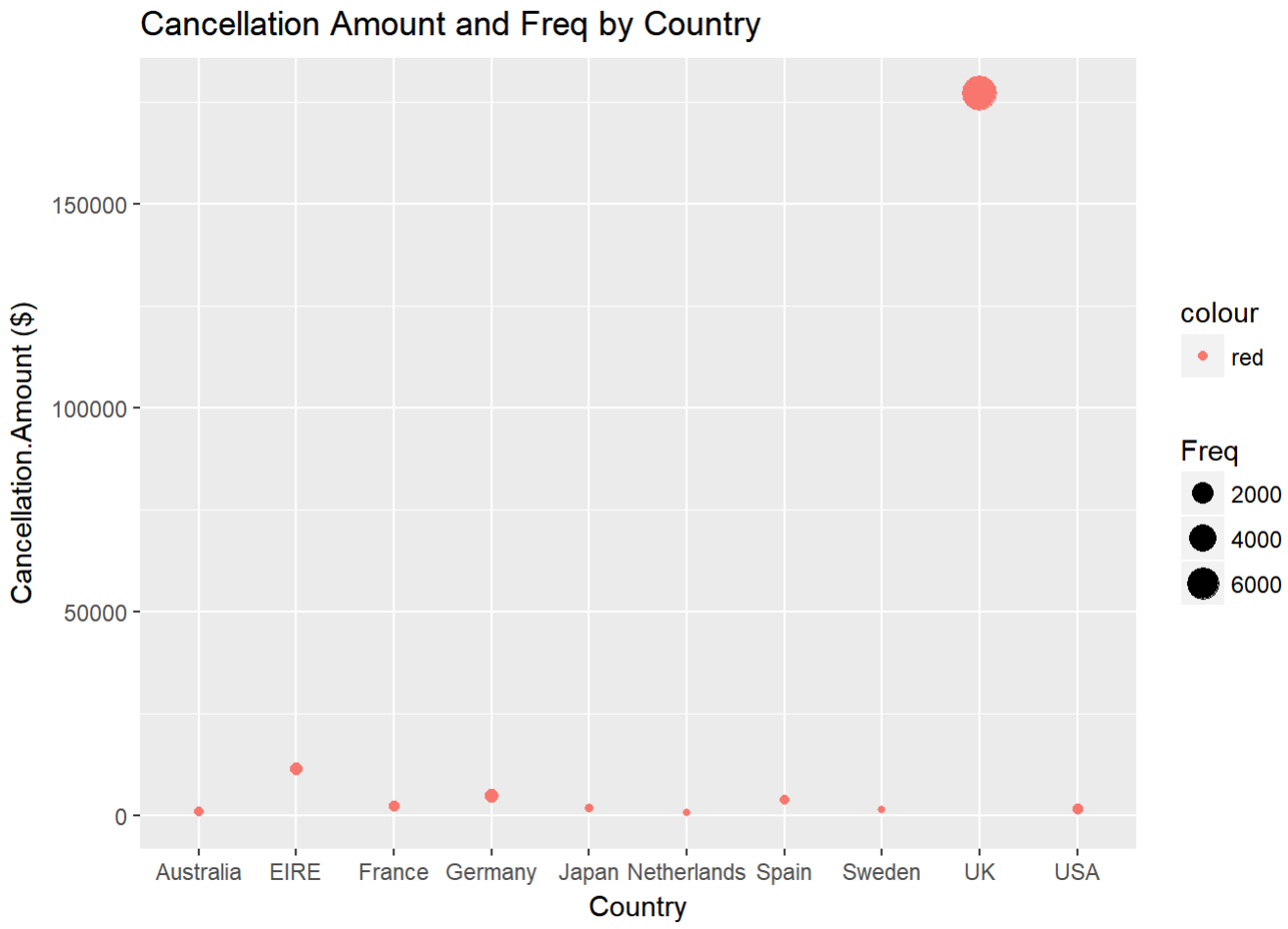
```
c_data_filt <- data_filt[(data_filt$cancel*data_filt$cust_trans)==1,]

cancellation <- aggregate(c_data_filt$sale*-1, by = list(c_data_filt$date),sum)
cancellation$Var1 = cancellation$Group.1
size_date <- data.frame(table(c_data_filt$date))

overall_cancel <- merge(cancellation[,c(2:3)],size_date, by.x ="Var1", by.y= "Var1")
```

a. Cancellations by month

```
ggplot(data = overall_cancel) +
  geom_point (mapping = aes(x = Var1, y = x, colour = "red", size = Freq))+ labs(x="Month", y="C
ancellation.Amount ($)", title = "Cancellation Amount and Freq by Month")+
  scale_x_discrete(limits=order)
```



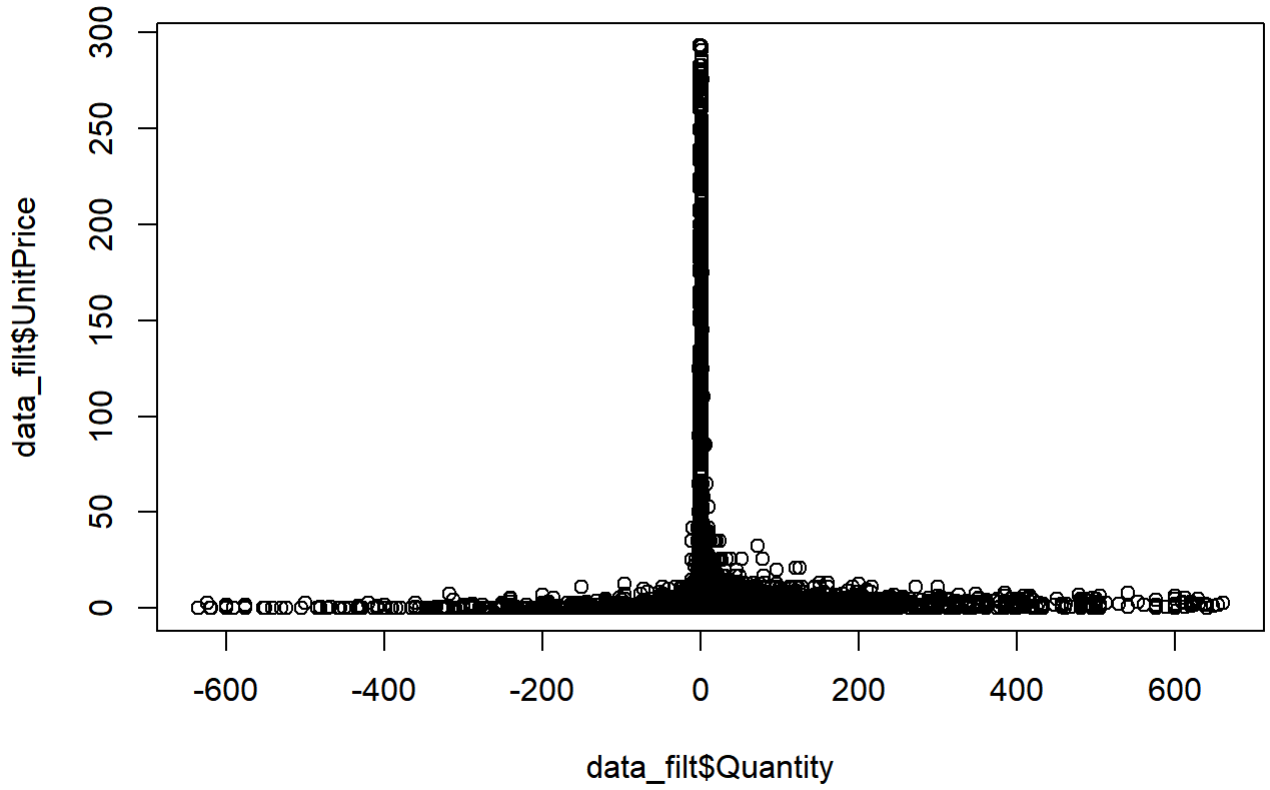

```
cancel_country <- aggregate(c_data_filt$sale*-1, by = list(c_data_filt$Country1),sum)
size_country <- data.frame(table(c_data_filt$Country1))
cancel_country <- merge(cancel_country,size_country, by.x ="Group.1", by.y= "Var1")
cancel_country <- cancel_country[order(-cancel_country$x),]


ggplot(data = cancel_country[1:10,]) +
  geom_point (mapping = aes(x = Group.1, y = x, colour = "red", size = Freq))+
  labs(x="Country", y="Cancellation.Amount ($)", title = "Cancellation Amount and Freq by Countr
y")
```




```
# + scale_x_discrete(limits=country)
```

Dependence between quantity ordered and unit price per product

```
plot(data_filt$Quantity, data_filt$UnitPrice)
```



Leading products in the market

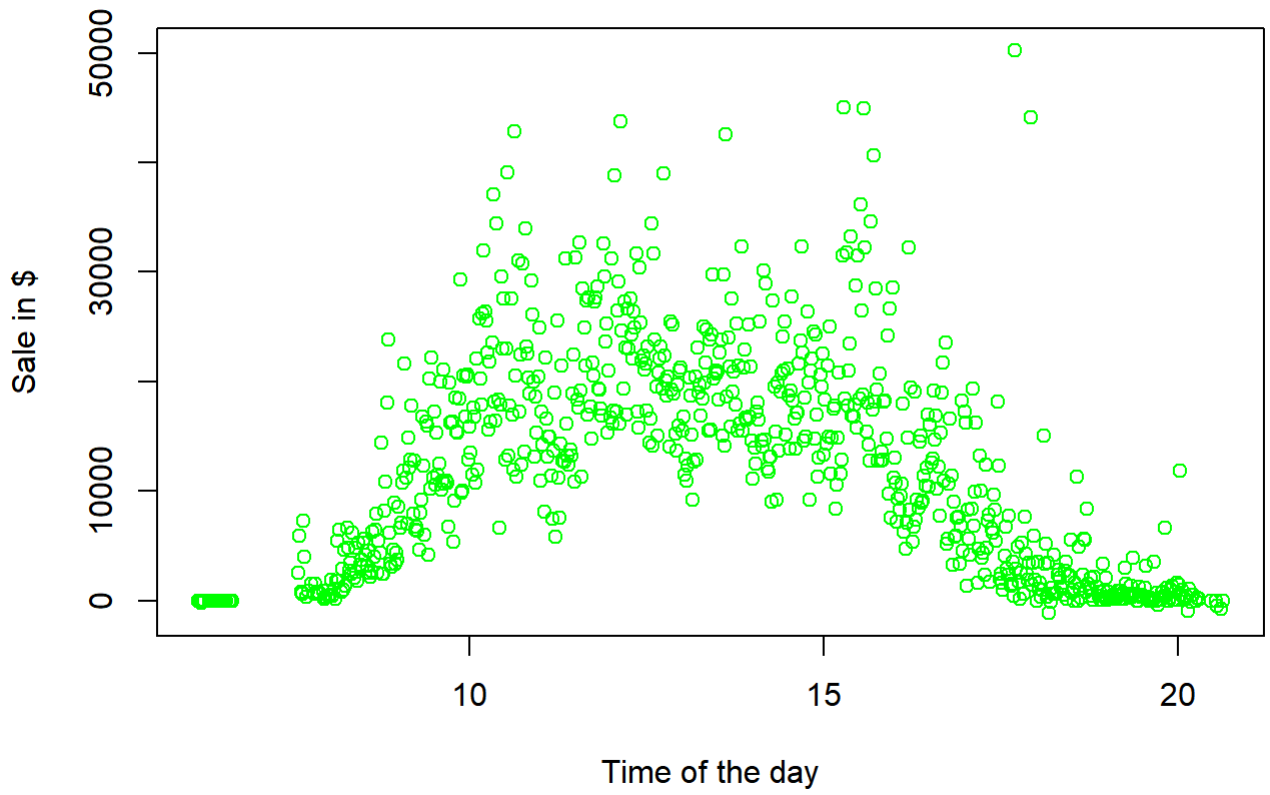
```
prod_sale_freq <- as.data.frame(table(data_filt$StockCode))
prod_sale <- aggregate(data_filt[,c(4,14)], by = list(data_filt$StockCode), sum)
unitP <- aggregate(data_filt[,6], by = list(data_filt$StockCode), mean)
write.csv(prod_sale,file = paste(dataPath,"prod_sale.csv", sep = "/"))
write.csv(unitP,file = paste(dataPath,"unitP.csv", sep = "/"))
```

Traffic and sale by time of the day

```
data_filt$hr_min <- hour(mdy_hm(data_filt$InvoiceDate))+ (minute(mdy_hm(data_filt$InvoiceDate))/
60)

time_split <- as.data.frame(table(data_filt$hr_min), rownames = TRUE)
write.csv(time_split, paste(dataPath, "time_split.csv", sep = "/"))
```

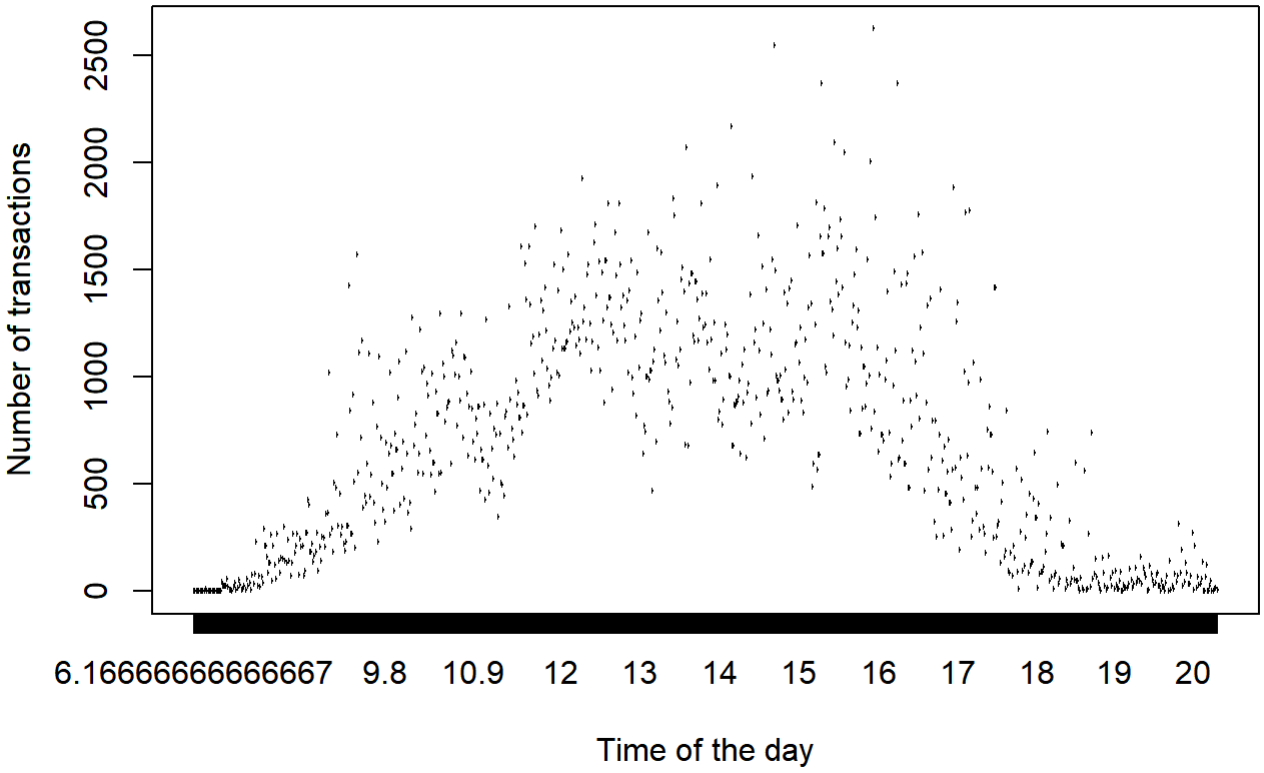
Traffic by time of the day

```
plot(aggregate(data_filt$sale, by = list(data_filt$hr_min), sum), type = 'p', col = "green", xla
b = "Time of the day", ylab = "Sale in $")
```



Sale by time of the day

```
plot(time_split, type = "p", xlab = "Time of the day", ylab = "Number of transactions")
```



#

Analysis of United Kingdom data

```
dat_uk <- data_filt[data_filt$Country1=="UK",]
nrow(dat_uk)
```

```
## [1] 494838
```

```
#dat_uk$keyw <- strsplit(as.character(dat_uk$Description)," ")
#word_list <- unique(unlist(dat_uk$keyw))
#write.csv(word_list, file = paste(dataPath,"desc_words.csv", sep = "/"))
#word_list <- VCorpus(DirSource(paste(dataPath,"texts", sep = "/")))
#word_list <-  tm_map(word_list, removeWords, stopwords("english"))
#word_list <-  tm_map(word_list, stripWhitespace)
#writeCorpus(word_list, path=paste(dataPath,"texts", sep = "/"))


shipping_cost <- sum(dat_uk$sale[dat_uk$post==1])
bank_charges <- sum(dat_uk$sale[dat_uk$charges ==1] )
cust_transactions <- sum(dat_uk$sale[dat_uk$cust_trans ==1])
canceled_trans <- sum(dat_uk$sale[(dat_uk$cancel ==1)*(dat_uk$cust_trans==1)])
```

Net transaction by month of the year

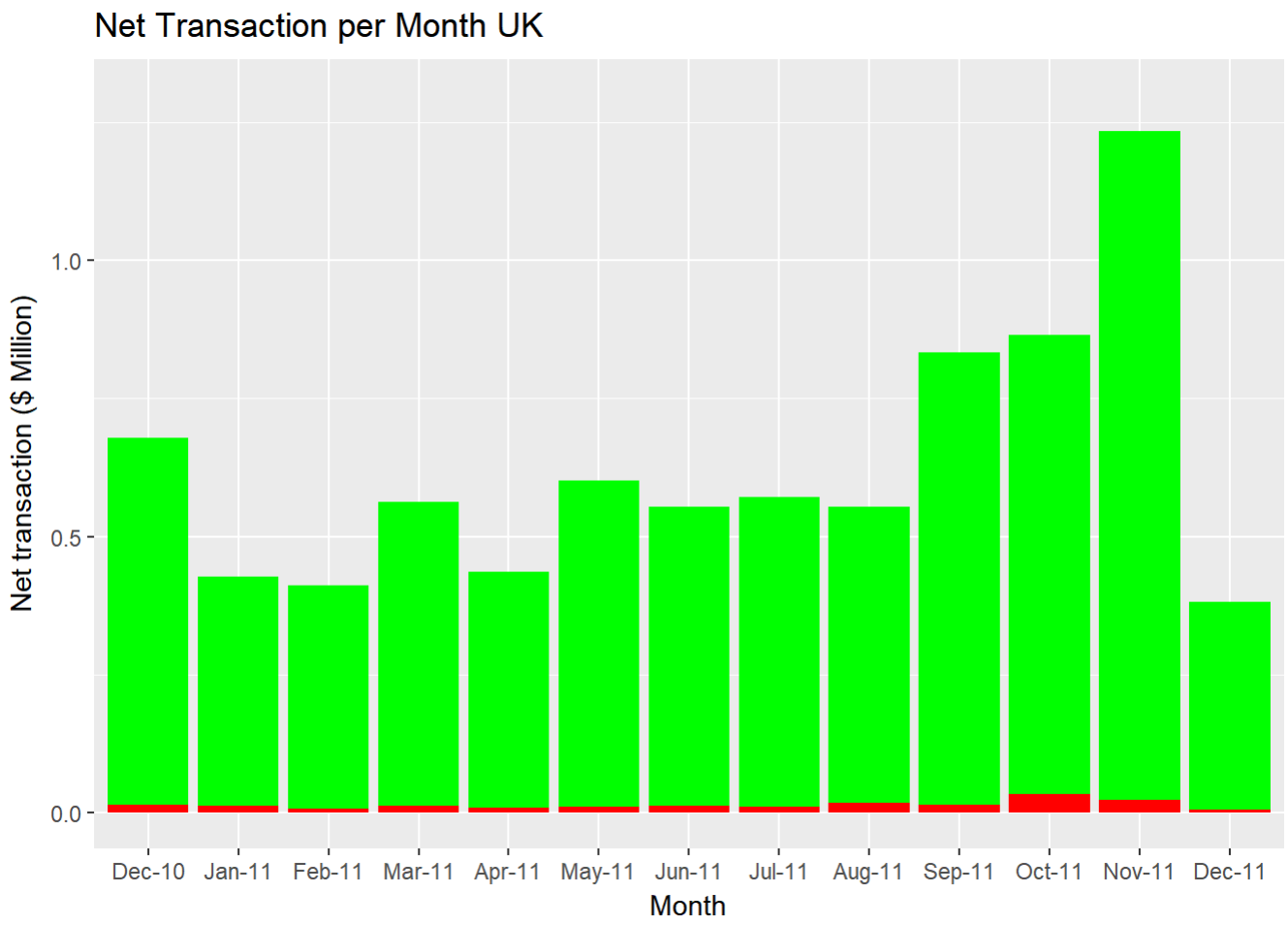
```
transuk <- aggregate(dat_uk$sale, by = list(dat_uk$date),sum)
transuk$x <- transuk$x/1000000
size_date <- data.frame(table(dat_uk$date))
transuk_date <- merge(transuk,size_date, by.x ="Group.1", by.y= "Var1")
```

Cancellations by month of the year

```
cancel_uk <- dat_uk[dat_uk$cancel==1,]
canceluk <- aggregate(cancel_uk$sale*-1, by = list(cancel_uk$date),sum)
canceluk$x <- canceluk$x/1000000
size_date <- data.frame(table(cancel_uk$date))
canceluk_date <- merge(canceluk,size_date, by.x ="Group.1", by.y= "Var1")
```

plot of Tranctions (green) and cancelation (red) amount

```
ggplot(data = transuk_date) +
  geom_col(mapping = aes(x = Group.1, y = x),  fill = "green")+  ylim(0,1.3) +
  labs(x="Month", y="Net transaction ($ Million)", title = "Net Transaction per Month UK", show.
legend= TRUE)+
  scale_x_discrete(limits=order) +
  geom_col(data = canceluk_date, mapping = aes(x = Group.1, y = x), fill = "red", show.legend =
TRUE)
```



Clustering of data for UK

1. Aggregate records by customer ID

```
datclus_uk <- aggregate(dat_uk[c(4,9,13,14)], by = list(dat_uk$CustomerID), sum)
up <- aggregate(dat_uk[c(6)], by = list(dat_uk$CustomerID), mean)
datclus_uk <- merge(datclus_uk, up, by = "Group.1")
```

2. Remove outliers (if any) because kmeans clustering technique does not ignore outliers and can bias the results

```
sd_qt <- sqrt(var(datclus_uk$Quantity))
mean_qt <- mean(datclus_uk$Quantity)
UL_qt <- mean_qt+3*sd_qt
LL_qt <- mean_qt-3*sd_qt
datclus_uk<- datclus_uk[datclus_uk$Quantity<UL_qt,]
datclus_uk<- datclus_uk[datclus_uk$Quantity>LL_qt,]
```

3. Create a variable “time_int” from InvoiceDate that provides the duration of association of the customer with the website in a year

```
ab<-split(dat_uk,dat_uk$CustomerID)
abx <- lapply(ab,function(x){
  max(as.Date(x$InvoiceDate,"%m/%d/%Y"))-min(as.Date(x$InvoiceDate,"%m/%d/%Y"))
})

test <- as.matrix(unlist(abx))
id <- as.data.frame(row.names(test))
abbd <- cbind(CustomerID = id , time_int=test )
abbd$Group.1 <- as.character(abbd$`row.names(test)`)
row.names(abbd)<- c(1:3934)
abbd <- abbd[,-1]

datclus_uk$Group.1 <- as.character(datclus_uk$Group.1)
datclus_uk1<- merge(datclus_uk,abbd, by.x= "Group.1", by.y= "Group.1")


scaled_uk <- scale(datclus_uk1[c(2,3,4,6,7)])
```

4. Splitting data into test and train

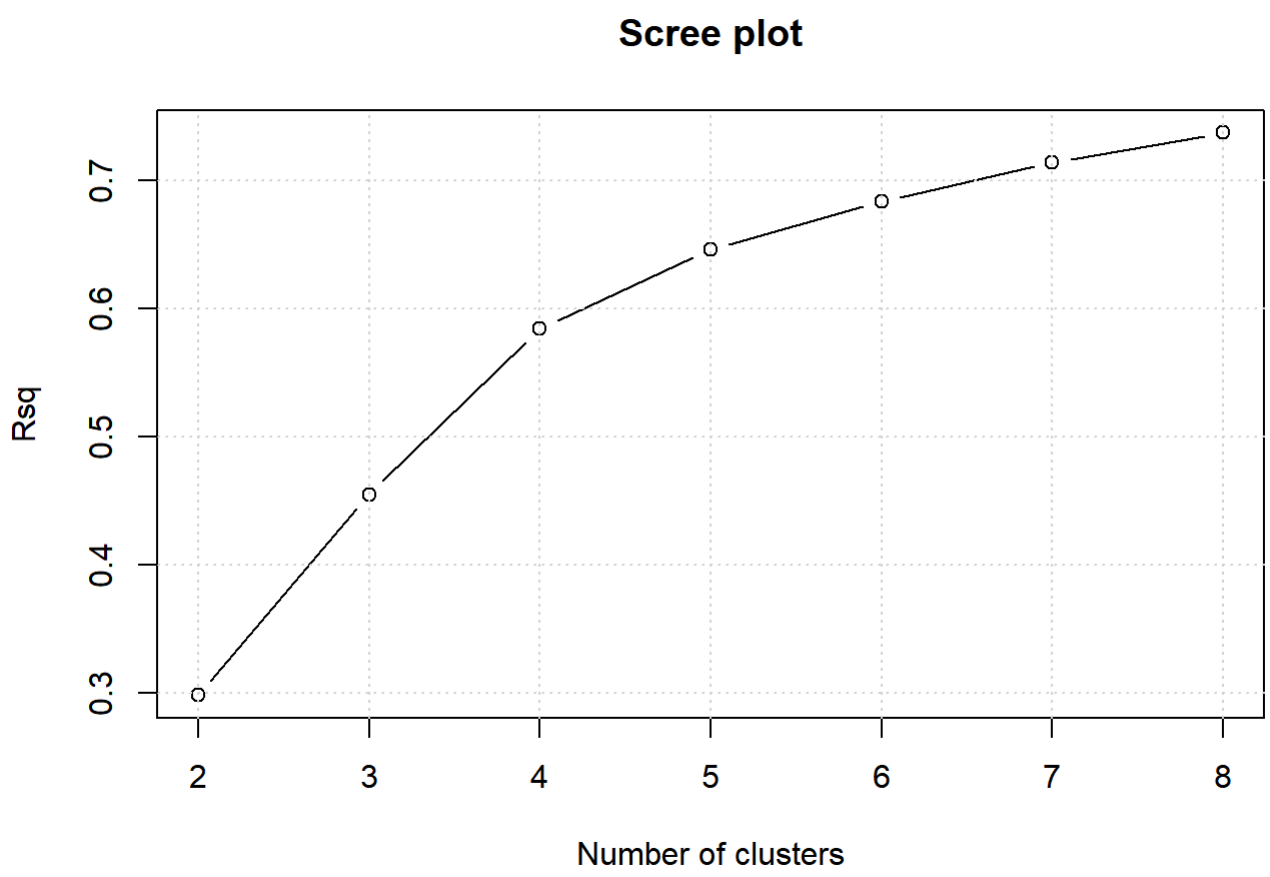
```
smp_size <- floor(0.70 * nrow(scaled_uk))

# set the seed to make partition reproductible
set.seed(123)
train_ind <- sample(seq_len(nrow(scaled_uk)), size = smp_size)

train <- scaled_uk[train_ind, ]
test <- scaled_uk[-train_ind, ]
```

5. Clustering of train data with continuos variables: Quantity, unit price, # cancellations, # transactions and duration of affiliation

```
Rsq = c()

for (i in 2:8){
  assign(paste0("kmean_uk",i), kmeans(train, i, nstart=100, iter.max = 20 ))
  assign(paste0("rsq",i),  eval(parse(text = paste0("kmean_uk",i,"$betweenss/kmean_uk",i,"$tots
s"))))
  Rsq = c(Rsq, eval(parse(text = paste0("rsq",i))))

  }
```

Scree plot to decide # clusters

```
plot(2:8, Rsq, type = "b", main = "Scree plot", xlab = "Number of clusters" ) + grid()
```




```
## integer(0)
```

Scree plot suggests 4 optimal clusters based on the elbow point

Aggregates based on 3 clusters

```
cbind(aggregate(datclus_uk1[train_ind,c(2,3,4,6,7,5)], by = list(kmean_uk3$cluster), mean), size
=as.matrix(kmean_uk3$size))
```

Group.1 <int>	Quantity <dbl>	cancel <dbl>	cust_trans <dbl>	UnitPrice <dbl>	time_int <dbl>	sale <dbl>	size <int>
1	3727.987	13.0897436	462.5449	2.831789	307.00641	6302.7781	156
2	1013.819	1.8088377	109.1335	3.116580	258.98847	1745.6837	1041
3	268.788	0.5133725	32.5773	3.697948	29.85845	454.3509	1533

3 rows

Aggregates based on 4 clusters

```
cbind(aggregate(datclus_uk1[train_ind,c(2,3,4,6,7,5)], by = list(kmean_uk4$cluster), mean), size
=as.matrix(kmean_uk4$size))
```

Group.1 <int>	Quantity <dbl>	cancel <dbl>	cust_trans <dbl>	UnitPrice <dbl>	time_int <dbl>	sale <dbl>	size <int>
1	9.166667	0.1666667	2.666667	82.138889	2.00000	358.9917	6
2	3727.987179	13.0897436	462.544872	2.831789	307.00641	6302.7781	156
3	1014.413127	1.8098456	108.960425	3.134250	259.75676	1745.8793	1036
4	271.834856	0.5182768	33.061358	3.376891	30.19582	458.8067	1532

4 rows

Aggregates based on 5 clusters

```
cbind(aggregate(datclus_uk1[train_ind, c(2,3,4,6,7,5)], by = list(kmean_uk5$cluster), mean), siz
e=as.matrix(kmean_uk5$size))
```

Group.1 <int>	Quantity <dbl>	cancel <dbl>	cust_trans <dbl>	UnitPrice <dbl>	time_int <dbl>	sale <dbl>	size <int>
1	2929.123016	5.0198413	307.702381	2.748812	292.32143	4788.3882	252
2	4034.025000	28.5500000	664.675000	3.101669	337.10000	7878.6798	40
3	273.391509	0.4952830	33.024259	3.375831	25.81267	458.1704	1484
4	784.531646	1.6550633	87.082278	3.202256	250.85021	1364.0403	948
5	9.166667	0.1666667	2.666667	82.138889	2.00000	358.9917	6

5 rows

I will go ahead with 4 cluster design for targeting based on their interpretability, practical use for marketing/ targeting, and cluster sizes

6. Check stability of clusters on test data using centers from train clusters

```
kmean_uk_test <- kmeans(test, centers = kmean_uk4$centers, nstart=100, iter.max = 20 )
Rsq_test <- kmean_uk_test$betweenss/kmean_uk_test$totss
c(Rsq_train = Rsq[3], Rsq_test = Rsq_test)
```

```
## Rsq_train  Rsq_test
## 0.5839450 0.6138064
```

R square values are close for both test and train data with 4 clusters and same centers

Compare the aggregates for the unscaled test data with train clusters: - Test clusters:

Group.1 <int>	Quantity <dbl>	UnitPrice <dbl>	cancel <dbl>	cust_trans <dbl>	time_int <dbl>	sale <dbl>	size <int>
1	53.33333	91.772222	0.0000000	1.333333	11.00000	718.6667	3
2	4276.78082	3.088488	12.7123288	300.273973	300.26027	7013.1437	73

Group.1 <int>	Quantity <dbl>	UnitPrice <dbl>	cancel <dbl>	cust_trans <dbl>	time_int <dbl>	sale <dbl>	size <int>
3	986.19002	3.064959	1.4489311	114.327791	265.52494	1706.6049	421
4	275.32938	3.321209	0.4658754	31.919881	28.78338	455.6842	674

4 rows

- Train clusters

Group.1 <int>	Quantity <dbl>	UnitPrice <dbl>	cancel <dbl>	cust_trans <dbl>	time_int <dbl>	sale <dbl>	size <int>
1	9.166667	82.138889	0.1666667	2.666667	2.00000	358.9917	6
2	3727.987179	2.831789	13.0897436	462.544872	307.00641	6302.7781	156
3	1014.413127	3.134250	1.8098456	108.960425	259.75676	1745.8793	1036
4	271.834856	3.376891	0.5182768	33.061358	30.19582	458.8067	1532

4 rows

```
round(prop.table(kmean_uk4$size*100),2)
```

```
## [1] 0.00 0.06 0.38 0.56
```

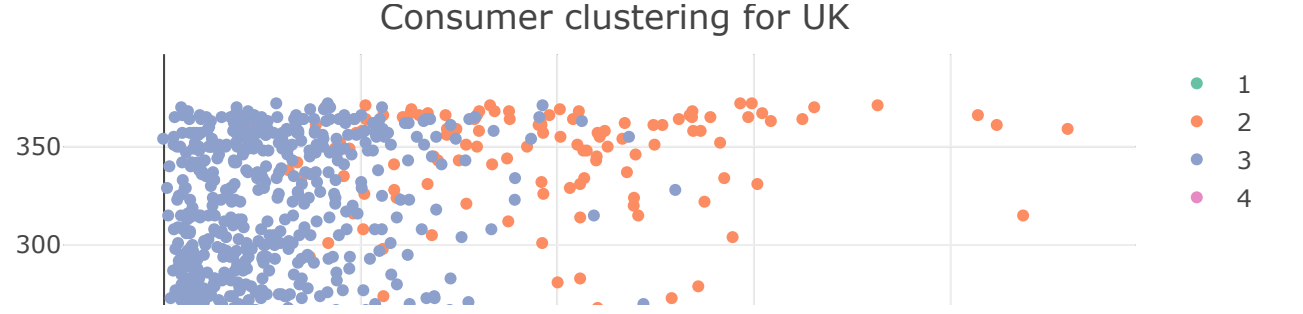
Interpretation of clusters:

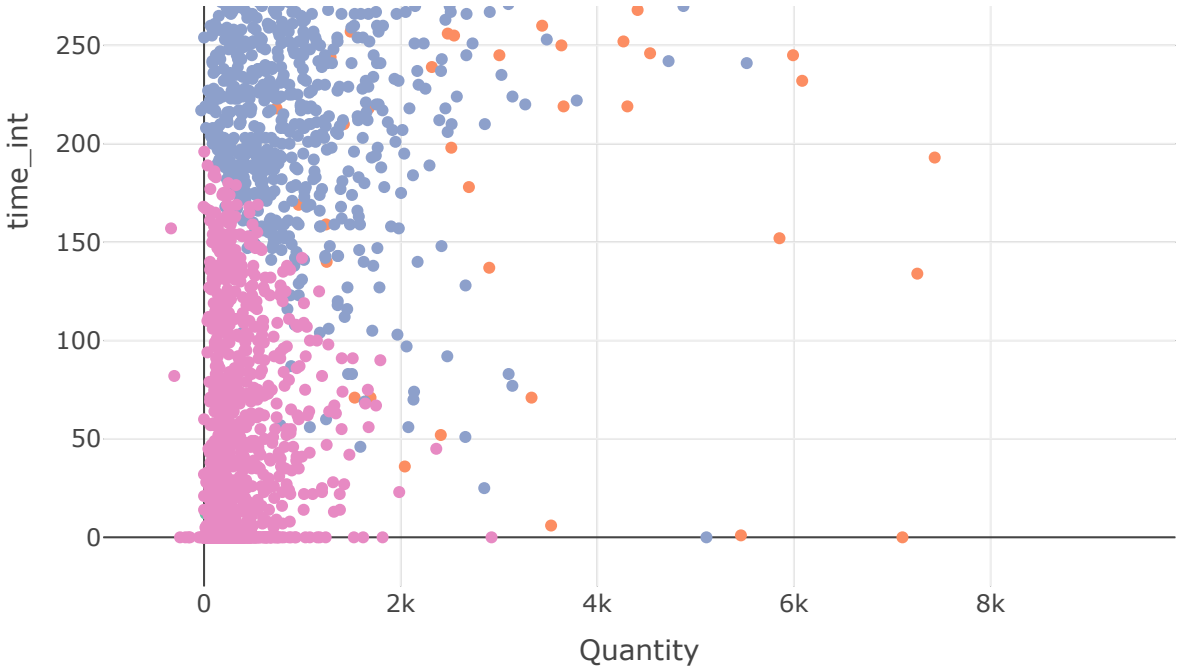
Cluster 1: “Outliers”- A minute proportion of customers who bought expensive products from the site once or twice

Cluster 2: “Old loyal customers (Revenue drivers)”: Buy a lot of products, frequent shoppers and large average cancellations| Among top 5% based on revenue

Cluster 3: “Old selective buyers” - Inclined towards buying less but frequently | Constitute more than a third of the customer base
Cluster 4: “New judicious shoppers” - Buy products in small quantity, less frequent shopping | Constitute more than half of the customer base

```
datclus_uk2 <- as.data.frame(cbind(datclus_uk1[train_ind,], kmean_uk3$cluster ))

plot_ly(datclus_uk2, x = ~Quantity, y = ~time_int, color = as.factor(kmean_uk4$cluster) ) %>%
add_markers() %>%
layout(title = 'Consumer clustering for UK', scene = list(xaxis = list(title = 'Quantity'),yaxis
 = list(title = 'Duration of shopping')))
```




time_int
250
200
150
100
50
0
0
2k
4k
6k
8k
Quantity