Customer Segmentation Analysis

Nicholas Chavez

June 6th, 2025

Business Task: Using Online Retail data donated to UC Irvine on 11/5/2015 that recorded all transactions occurring between 01/12/2010 to 19/12/2011 for a UK-based registered non-store online retailer, I am prompted with the task of differentiating customer types to drive targeted marketing campaigns.

Setting up packages and loading data

```
library(tidyverse)
library(lubridate)
library(plotly)
setwd("C:/Users/Nicho/OneDrive/Desktop/Customer Segmentation")
sales_data <- read.csv("Online_Retail.csv")
```

Explore Data

```
head(sales_data)
```

```
  InvoiceNo StockCode                         Description Quantity
1    536365    85123A  WHITE HANGING HEART T-LIGHT HOLDER        6
2    536365     71053                 WHITE METAL LANTERN        6
3    536365    84406B      CREAM CUPID HEARTS COAT HANGER        8
4    536365    84029G KNITTED UNION FLAG HOT WATER BOTTLE        6
5    536365    84029E      RED WOOLLY HOTTIE WHITE HEART.        6
6    536365     22752        SET 7 BABUSHKA NESTING BOXES        2
     InvoiceDate UnitPrice CustomerID        Country
```

```
1 12/1/2010 8:26      2.55      17850 United Kingdom
2 12/1/2010 8:26      3.39      17850 United Kingdom
3 12/1/2010 8:26      2.75      17850 United Kingdom
4 12/1/2010 8:26      3.39      17850 United Kingdom
5 12/1/2010 8:26      3.39      17850 United Kingdom
6 12/1/2010 8:26      7.65      17850 United Kingdom
```

```
tibble(arrange(sales_data, Quantity))
```

```
# A tibble: 541,909 x 8
   InvoiceNo StockCode Description     Quantity InvoiceDate UnitPrice CustomerID
   <chr>     <chr>     <chr>              <int> <chr>           <dbl>      <int>
 1 C581484   23843     "PAPER CRAFT ,~   -80995 12/9/2011 ~      2.08      16446
 2 C541433   23166     "MEDIUM CERAMI~   -74215 1/18/2011 ~      1.04      12346
 3 556690    23005     "printing smud~    -9600 6/14/2011 ~      0            NA
 4 556691    23005     "printing smud~    -9600 6/14/2011 ~      0            NA
 5 C536757   84347     "ROTATING SILV~    -9360 12/2/2010 ~      0.03      15838
 6 556687    23003     "Printing smud~    -9058 6/14/2011 ~      0            NA
 7 546152    72140F    "throw away"       -5368 3/9/2011 1~      0            NA
 8 573596    79323W    "Unsaleable, d~    -4830 10/31/2011~      0            NA
 9 566768    16045     ""                 -3667 9/14/2011 ~      0            NA
10 565304    16259     ""                 -3167 9/2/2011 1~      0            NA
# i 541,899 more rows
# i 1 more variable: Country <chr>
```

Customer ID is a unique identifier that can be used to create a summary data for frequency, InvoiceDate can be used to determine how recent orders are, and the sum of the product of Quantity and Unit price per CustomerID can be used to determine monetary spending habits per customer. Also, Invoice data is in character format, using lubricate is necessary to covert the data to date format. Additionally, Quantity can be negative, based on description of data seems to be salvage/unsaleable merchandise, this will have to be removed in the cleaning process.

Data Cleaning and mutation:

In this cleaning, I dropped null values for CustomerIDs, removed negative values for Quantity, added the TotalPurchase column which is the sum of the products of Quantity and UnitPrice, and reformatted InvoiceDate from Character String to Date.

```r
cleaned_sales <- sales_data |>
  drop_na(CustomerID) |>
  filter(Quantity > 0) |>
  mutate(TotalPurchase = Quantity*UnitPrice) |>
  mutate(InvoiceDate = mdy_hm(InvoiceDate))
```

Data for Recency, Frequency, and Monetary Spending

In this section, I calculated new fields Recency, Frequency, and Monetary to help determine customer type. The first step was creating a reference date to compare dates to determine recency before using n_distinct() and sum() to create frequency and monetary.

```r
reference_date <- max(cleaned_sales$InvoiceDate) + days(1)
RFM_data <- cleaned_sales |>
  group_by(CustomerID) |>
  summarise(
    recency = as.numeric(reference_date - max(InvoiceDate), units = "days"),
    frequency = n_distinct(InvoiceNo),
    monetary = sum(TotalPurchase)
  )
```

Prepare for K-means

Standardize the data for comparability between different fields to prepare for K-means.

```r
Standardize <- function(x){
  (x - min(x))/(max(x)-min(x))
} # Creating a function to standardize data

# Standardize all values for comparability
rfm_stz <- RFM_data |>
  select(recency, frequency, monetary) |>
  mutate(recency = Standardize(recency)) |>
  mutate(frequency = Standardize(frequency)) |>
  mutate(monetary = Standardize(monetary))
```

check results

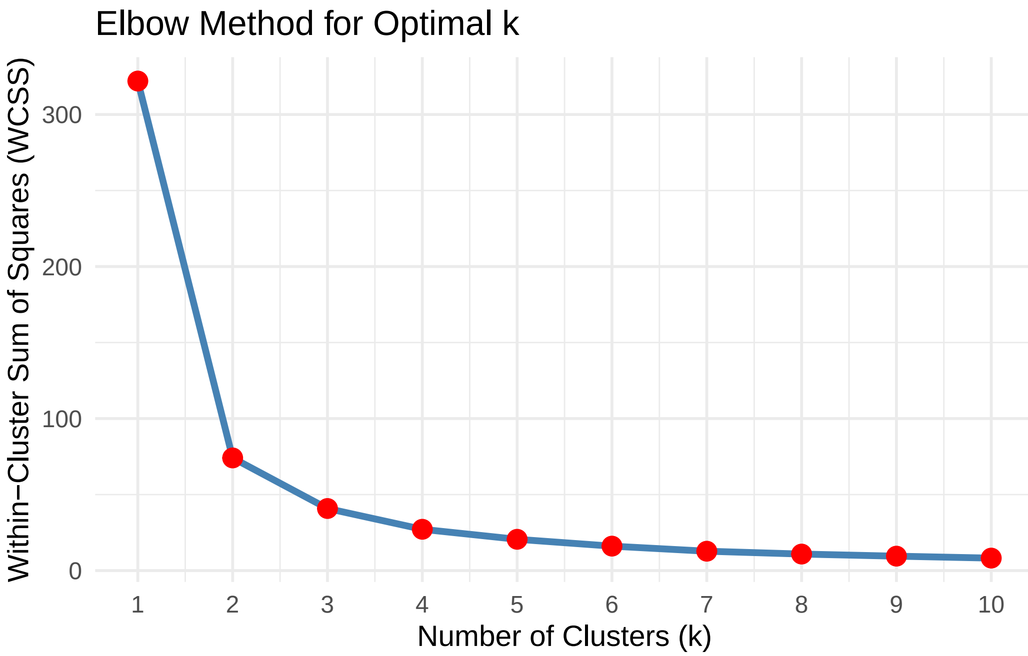
```r
summary(rfm_stz)
```

```
    recency          frequency           monetary
 Min.   :0.00000   Min.   :0.000000   Min.   :0.000000
 1st Qu.:0.04575   1st Qu.:0.000000   1st Qu.:0.001097
 Median :0.13423   Median :0.004785   Median :0.002407
 Mean   :0.24665   Mean   :0.015655   Mean   :0.007330
 3rd Qu.:0.37929   3rd Qu.:0.019139   3rd Qu.:0.005930
 Max.   :1.00000   Max.   :1.000000   Max.   :1.000000
```

I need to determine which the numbers of clusters to specify. For this I will be using total-within cluster sum of squares (WCSS), creating a tibble of the results, and then plot the WCSS. I then determine which k to proceed with based on the Elbow Method.

```r
set.seed(123)
wcss <- map_dbl(1:10, ~{
  kmeans(rfm_stz[, c("recency", "frequency", "monetary")], centers = .,
         nstart = 25)$tot.withinss
})
wcss_data <- tibble(
  k = 1:10,
  wcss = wcss
)

ggplot(wcss_data, aes(k, wcss)) +
  geom_line(color = "steelblue", linewidth = 1.2) +
  geom_point(color = "red", size = 3) +
  labs(
    title = "Elbow Method for Optimal k",
    x = "Number of Clusters (k)",
    y = "Within-Cluster Sum of Squares (WCSS)"
  ) +
  scale_x_continuous(breaks = 1:10) +
  theme_minimal()
```



Based on the Graph, 4 is where the elbow appears and appears flat.

K-Mean Clustering

Now I will run the K-means algorithim to group/segment the customers.

```
set.seed(123)
km_out <- kmeans(rfm_stz, 4, nstart = 42)

km_out$centers
```

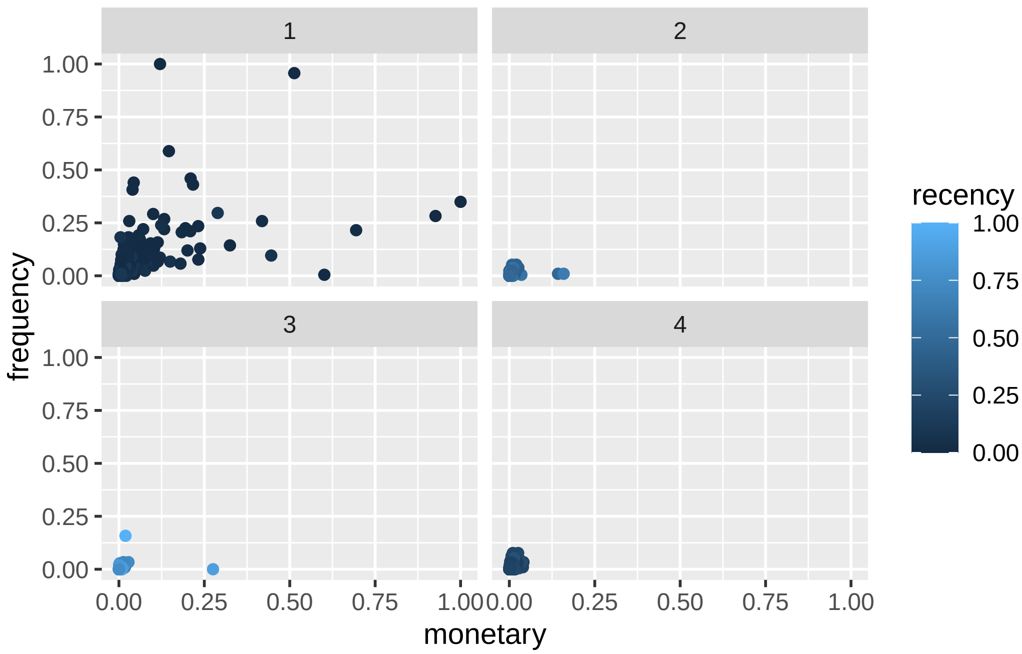
```
    recency   frequency    monetary
1 0.0500849 0.026288833 0.011859202
2 0.5119836 0.004180796 0.002595808
3 0.8264392 0.001671759 0.002019332
4 0.2155235 0.007792083 0.003598462
```

Adding the clusters to the data set

```r
clustered_sales <- rfm_stz |>
  mutate(cluster = factor(km_out$cluster))
```

Visualize the data

```r
ggplot(clustered_sales, aes(x = monetary, y =frequency))+
  geom_point(aes(color = recency)) +
  facet_wrap(~cluster)
```



Findings: Through K-mean clustering I was able to determine 4 separate groups of customers. The characteristics of each group is as follows:

- **Group 1:** Mostly recent with higher spread in spending and frequency.
- **Group 2:** Third most recent with lower levels of spending and frequency. Has a few higher spenders.
- **Group 3:** Least recent with lower levels of spending and frequency.
- **Group 4:** Second most recent with lower levels of spending and frequency.

Restructure Data to prepare for further visualization

Here I will be taking the RFM_data that contains customerIDs then include the newly determine clusters before renaming them for interpretability. I will define each group based on their attributes. Group 1 will be label High Value, Group 2 would be Churning, Group 3 will be Inactive, and Group 4 are Occasional. All are defined based on the characteristics above.

```
cleaned_clusters_sales <- RFM_data |>
  mutate(Cluster = clustered_sales$cluster) |>
  mutate(
    segment = case_when(
      Cluster == 1 ~ "High Value",
      Cluster == 2 ~ "Churning",
      Cluster == 3 ~ "Inactive",
      Cluster == 4 ~ "Occasional"
    )
  )
head(cleaned_clusters_sales)
```

```
# A tibble: 6 x 6
  CustomerID recency frequency monetary Cluster segment
       <int>   <dbl>     <int>    <dbl> <fct>   <chr>
1      12346  326.           1   77184. 3       Inactive
2      12347    2.87         7    4310  1       High Value
3      12348   76.0          4    1797. 4       Occasional
4      12349   19.1          1    1758. 1       High Value
5      12350  311.           1     334. 3       Inactive
6      12352   36.9          8    2506. 1       High Value
```

Verification

I was able to apply the clusters back into the dataset because the order was the same, however that method can be prone to error. So to verify I unstandardized the values to verify they were correct. This method of verification was able to work because the ID column was drop when standardizing the data, so reverting the calculation should give the data set with the clusters the same value as the dataset with the ID.

```
unstandardize <- function(standardized_value, x){
  standardized_value*(max(x)-min(x)) + min(x)
}
check <- clustered_sales |>
```

```r
  mutate(
   recency = unstandardize(recency, RFM_data$recency),
   frequency = unstandardize(frequency, RFM_data$frequency),
   monetary = unstandardize(monetary, RFM_data$monetary)
  )
head(cleaned_clusters_sales)
```

```
# A tibble: 6 x 6
  CustomerID recency frequency monetary Cluster segment
       <int>   <dbl>     <int>    <dbl> <fct>   <chr>
1      12346  326.           1   77184. 3       Inactive
2      12347    2.87         7    4310  1       High Value
3      12348   76.0          4    1797. 4       Occasional
4      12349   19.1          1    1758. 1       High Value
5      12350  311.           1     334. 3       Inactive
6      12352   36.9          8    2506. 1       High Value
```

```r
head(check)
```

```
# A tibble: 6 x 4
  recency frequency monetary cluster
    <dbl>     <dbl>    <dbl> <fct>
1  326.           1   77184. 3
2    2.87         7    4310  1
3   76.0          4    1797. 4
4   19.1          1    1758. 1
5  311.           1     334. 3
6   36.9          8    2506. 1
```

```r
tail(cleaned_clusters_sales)
```

```
# A tibble: 6 x 6
  CustomerID recency frequency monetary Cluster segment
       <int>   <dbl>     <int>    <dbl> <fct>   <chr>
1      18278   74.0          1    174.  4       Occasional
2      18280  278.           1    181.  3       Inactive
3      18281  181.           1     80.8 2       Churning
4      18282    8.05         2    178.  1       High Value
5      18283    4.03        16   2095.  1       High Value
6      18287   43.1          3   1837.  1       High Value
```

```r
tail(check)
```

```
# A tibble: 6 x 4
  recency frequency monetary cluster
    <dbl>     <dbl>    <dbl> <fct>
1   74.0          1    174.  4
2  278.           1    181.  3
3  181.           1     80.8 2
4    8.05         2    178.  1
5    4.03        16   2095.  1
6   43.1          3   1837.  1
```

When looking at the table I looked to see if the recency,frequency, monetary, and cluster values matched. So this checks out and the data is associate with the correct customerID's. Since the data is now confirmed to be credible I can export the data to be used in other visualization tools.

```r
write.csv(cleaned_clusters_sales, "Segmented_data_Irvine.csv",
          row.names = FALSE)
```

My choice to create an CSV file before closing the analysis is because the current dataset "Segmented_data_Irvine.csv" can be used for Target Marketing. Utilizing the CustomerID section and Segement will allow the marketing team to isolate customers into their respective groups and contact them with promotions and rewards.

Customer Lifetime values

CLV = frequency * lifespan * average_spending

frequency = n_distinct(InvoiceNo) lifespan = max(InvoiceDate) - min(InvoiceDate) average_spending = mean(TotalPurchases)

```r
CLV_data <- cleaned_sales |>
            group_by(CustomerID) |>
            summarize(
              frequency = n_distinct(InvoiceNo),
              lifespan = as.numeric(max(InvoiceDate) - min(InvoiceDate), units = "days"),
              avg_spending = mean(TotalPurchase)) |>
            mutate(CLV = frequency * lifespan * avg_spending) |>
            left_join(cleaned_clusters_sales |> select("CustomerID", "segment"), by = "Custon
head(CLV_data)
```

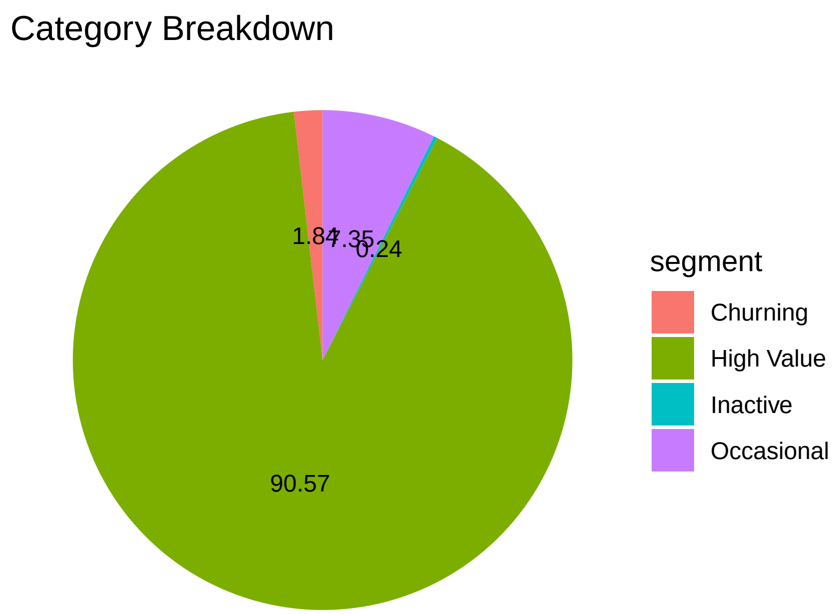
```
# A tibble: 6 x 6
  CustomerID frequency lifespan avg_spending    CLV segment
       <int>     <int>    <dbl>        <dbl>  <dbl> <chr>
1      12346         1        0       77184.      0 Inactive
2      12347         7      365.          23.7 60512. High Value
3      12348         4      283.          58.0 65571. Occasional
4      12349         1        0           24.1      0 High Value
5      12350         1        0           19.7      0 Inactive
6      12352         8      260.          29.5 61345. High Value
```

```r
grouped_CLV <- CLV_data |>
  mutate( CLV_portion = (CLV / sum(CLV)) * 100 ) |>
  group_by(segment) |>
  summarize(
    avg_frequency = mean(frequency),
    avg_spending = mean(avg_spending),
    percent_of_lifetimevalue = sum(CLV_portion),
    total_value = sum(CLV)
  )
```

```r
head(grouped_CLV)
```

```
# A tibble: 4 x 5
  segment    avg_frequency avg_spending percent_of_lifetimevalue total_value
  <chr>              <dbl>        <dbl>                    <dbl>       <dbl>
1 Churning            1.87         64.7                    1.84     4266402.
2 High Value          6.49         57.6                   90.6    209751372.
3 Inactive            1.35        192.                     0.241     557453.
4 Occasional          2.63         35.0                    7.35    17018868.
```

```r
ggplot(grouped_CLV, aes(x = "", y = percent_of_lifetimevalue, fill = segment) ) +
  geom_bar(stat = "identity", width = 1) + # Create the bar
  coord_polar("y", start = 0) +            # Wrap the bar into a circle
  theme_void() +                           # Remove background grid/axes
  labs(title = "Category Breakdown") +
  geom_text(aes(label = round(percent_of_lifetimevalue,2)),
            position = position_stack(vjust = 0.5),
            color = "black",
            size = 3)
```



Though the final calculation of customer lifetime value, and the relative percentages, we can see that 90.57% of customer lifetime value come from the high value customers, 7.35% come from occasional customers, 1.84% come from Churning customers, and 0.24% come from inactive customers.