Big Data: E-commerce analysis

Group 2

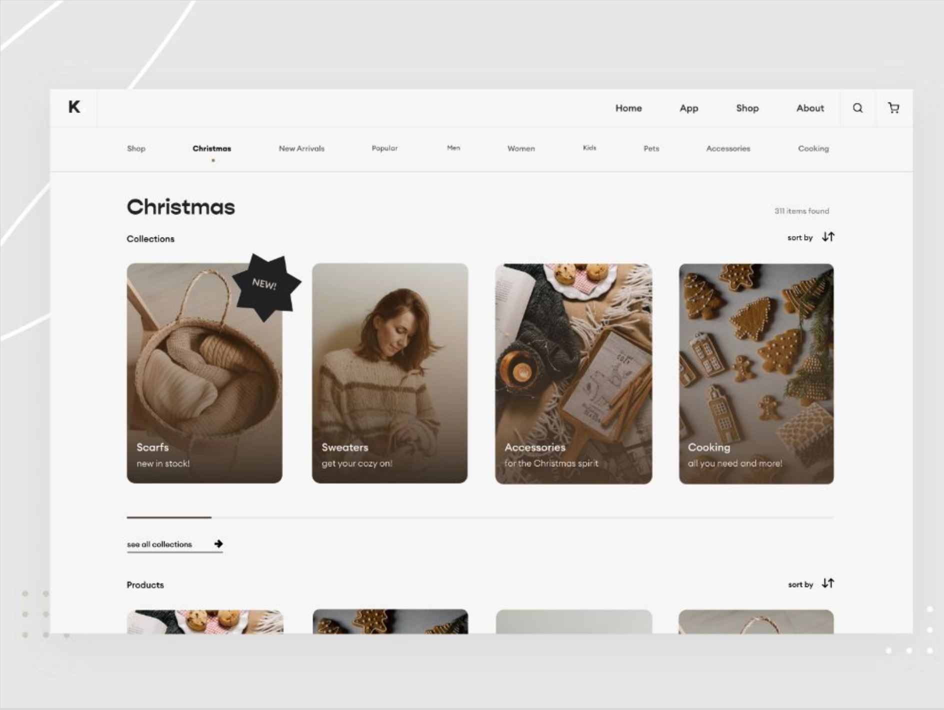
Table of contents

01

Dataset Introduction

UK e-commerce website

1 year
of sales transaction data



Sells gifts and homeware products

Converting it to a Parquet file

Saving it in a Parquet file format

```{r}
#OPEN SAVED PARQUET FILE HERE
transactions_parquet <- arrow::open_dataset(
  sources = "parquet_folder/part-0.parquet",
  format = "parquet")

# Collect the data into a data frame
transaction_df <- transactions_parquet |>
  dplyr::collect()
```

Read the parquet file to Spark

```
library(sparklyr)
library(dplyr)

# Connect to a local instance of Spark
sc <- spark_connect(master = "local", version = "3.4.0")

# Copy the transactions R data frame to Spark memory and create the
R reference transaction_ref
transaction_ref <- copy_to(sc, transaction_df)
head(transaction_ref, 4)
```

Data Cleaning

Removing rows w/empty values or negative quantities

```{r}
transaction_clean <- transaction_ref |>
  filter(!is.na(CustomerNo) & (Quantity > 0 & Quantity < 1000) & !is.null(CustomerNo))

```

Source: SQL [?? x 8]   Database: spark_connection

TransactionNo <chr>	Date <chr>	ProductNo <chr>	ProductName <chr>	Price <dbl>	Quantity <int>	CustomerNo <int>	Country <chr>
C581484	12/9/2019	23843	Paper Craft Little Birdie	6.19	-80995	16446	United Kingdom
C581490	12/9/2019	22178	Victorian Glass Hanging T-Light	6.19	-12	14397	United Kingdom
C581490	12/9/2019	23144	Zinc T-Light Holder Stars Small	6.04	-11	14397	United Kingdom
C581568	12/9/2019	21258	Victorian Sewing Box Large	6.19	-5	15311	United Kingdom
C581569	12/9/2019	84978	Hanging Heart Jar T-Light Holder	6.19	-1	17315	United Kingdom
C581569	12/9/2019	20979	36 Pencils Tube Red Retrospot	6.19	-5	17315	United Kingdom
C581228	12/8/2019	22423	Regency Cakestand 3 Tier	6.19	-6	16019	United Kingdom
C581228	12/8/2019	23210	White Rocking Horse Hand Painted	6.19	-12	16019	United Kingdom
C581228	12/8/2019	82494L	Wooden Frame Antique White	6.19	-6	16019	United Kingdom
C581228	12/8/2019	22781	Gumball Magazine Rack	6.19	-24	16019	United Kingdom

1-10 of 1,000 rows   Previous 1 2 3 4 5 6 ... 100 Next

Data Cleaning

Removing anomalies in sales

```{r}
#remove these
transaction_ref |>
  filter(Quantity >10000)
```

Source: SQL [?? x 8] Database: spark_connection

TransactionNo <chr>	Date <chr>	ProductNo <chr>	ProductName <chr>	Price <dbl>	Quantity <int>	CustomerNo <int>	Country <chr>
581483	12/9/2019	23843	Paper Craft Little Birdie	12.38	80995	16446	United Kingdom
578841	11/25/2019	84826	Asstd Design 3d Paper Stickers	6.19	12540	13256	United Kingdom
541431	1/18/2019	23166	Medium Ceramic Top Storage Jar	11.32	74215	12346	United Kingdom

3 rows

Data Cleaning

Converting dates into a standard date format

```
2.2 Converting date to date format
```{r}
library(dplyr)

transaction_clean <- transaction_clean |>
    mutate(
    month = substring_index(Date, "/", 1), #anyone else can't do substring?
    day = substring_index(substring_index(Date, "/", -2), "/", 1),
    year = substring_index(Date, "/", -1)
  )
# Add leading zeros to month and day
transaction_clean <- transaction_clean |>
  mutate(
    month = lpad(month, 2, "0"),
    day = lpad(day, 2, "0")
  )

# Combine the formatted values to create the "yyyy/mm/dd" date
transaction_clean <- transaction_clean |>
  mutate(FormattedDate = concat(year, "-", month, "-", day)) |>
  select(-month, -day, -year, -Date)

#convert to date format from chr
transaction_clean <-transaction_clean |>
  mutate(FormattedDate = to_date(FormattedDate))

transaction_clean
```
```

Source: SQL [?? x 8] Database: spark_connection

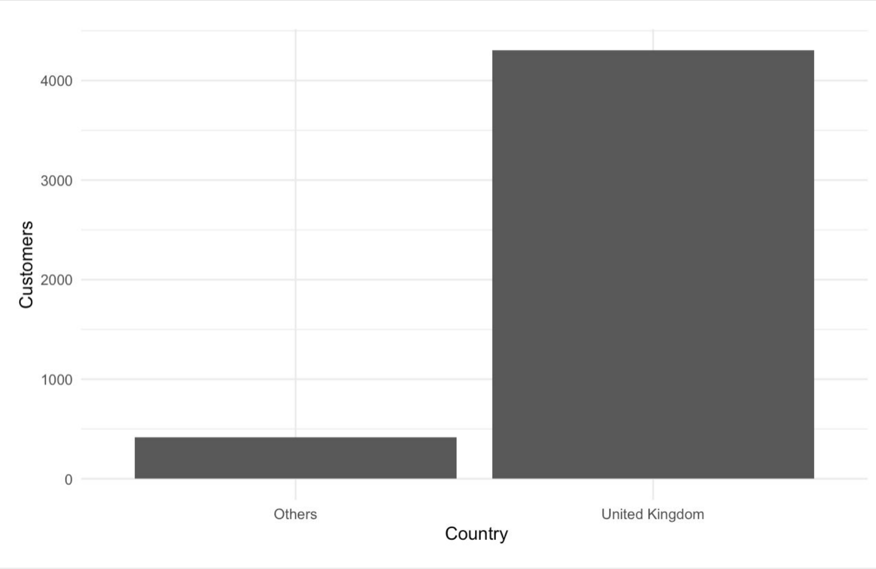
Price <dbl>	Quantity <int>	CustomerNo <int>	Country <chr>	FormattedDate <date>
21.47	12	17490	United Kingdom	2019-12-09
10.65	36	13069	United Kingdom	2019-12-09
11.53	12	13069	United Kingdom	2019-12-09
10.65	12	13069	United Kingdom	2019-12-09
11.94	6	13069	United Kingdom	2019-12-09
10.65	24	13069	United Kingdom	2019-12-09
11.53	18	13069	United Kingdom	2019-12-09
12.25	12	13069	United Kingdom	2019-12-09
10.65	12	13069	United Kingdom	2019-12-09
10.55	24	13069	United Kingdom	2019-12-09

1-10 of 1,000 rows | 4-8 of 8 columns Previous 1 2 3 4 5 6 ... 100 Next

02

Customer EDA

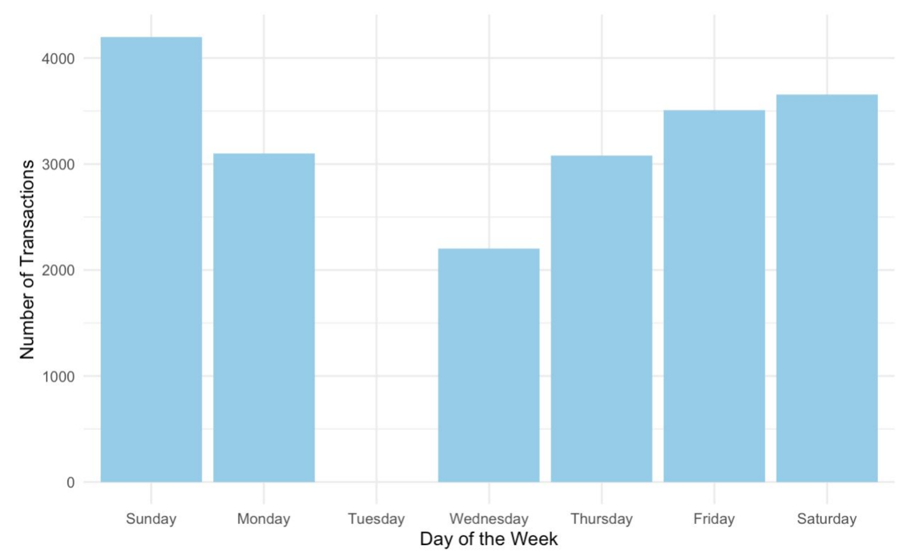
Geographical analysis




```
library(dbplot)
library(ggplot2)
country_plot <- transaction_clean |>
  mutate(Country = ifelse(Country == "United Kingdom", Country, "Others"))

dbplot_bar(country_plot, x = Country, Customers = n_distinct(CustomerNo)) +
  theme_minimal()
```

Seasonality - Day of the week

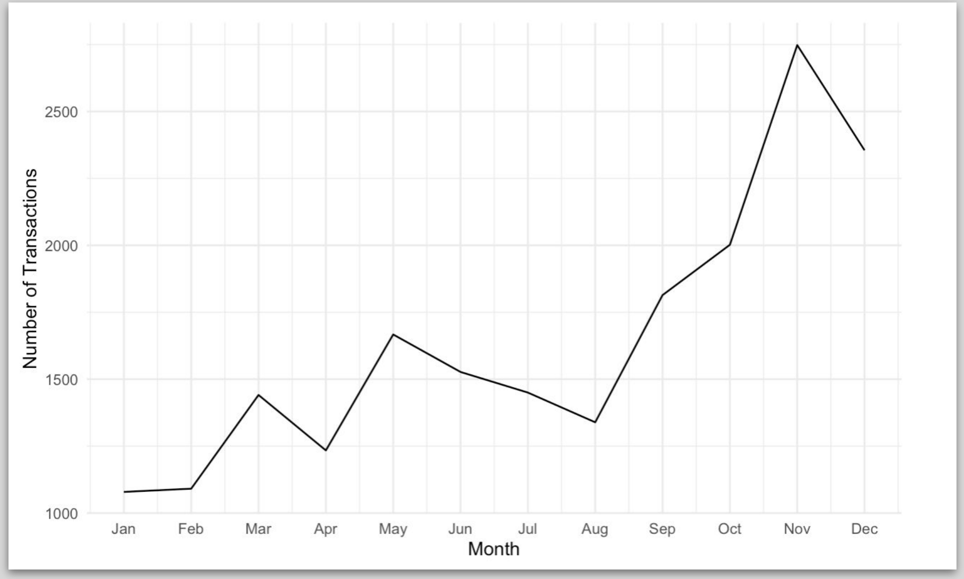



```
lm(formula = number_of_transactions ~ day_of_week, data =

Residuals:
      1       2       3       4       5       6       7
  526.5   175.3  1622.2   593.8   441.5  -619.0 -2740.3

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2498.00    1274.99   1.959    0.107
day_of_week    80.75     285.10   0.283    0.788
```

Seasonality - Month




```
lm(formula = Number_of_transactions ~ month, data = table1)

Residuals:
    Min      1Q  Median      3Q     Max 
-488.66 -114.86  -33.13  126.63  556.20 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   856.61     168.97   5.070 0.000485 ***
month         121.38      22.96   5.287 0.000354 ***
---
```

Feature engineering - RFM

Recency

Number of days since the last transaction. The higher it is, the longer it has been since the customer's last purchase.

Frequency

Total number of unique transactions per customer.

Monetary

Total amount the customer has spent in all their purchases with the business

```
ref_customer <- transaction_clean |>
  group_by(CustomerNo) |>
  summarise(
    Recency = as.numeric(datediff(max(FormattedDate),
                                  to_date("2019-12-09"))*(-1)),
    Frequency = n_distinct(TransactionNo),
    Monetary = sum(Price * Quantity),
```

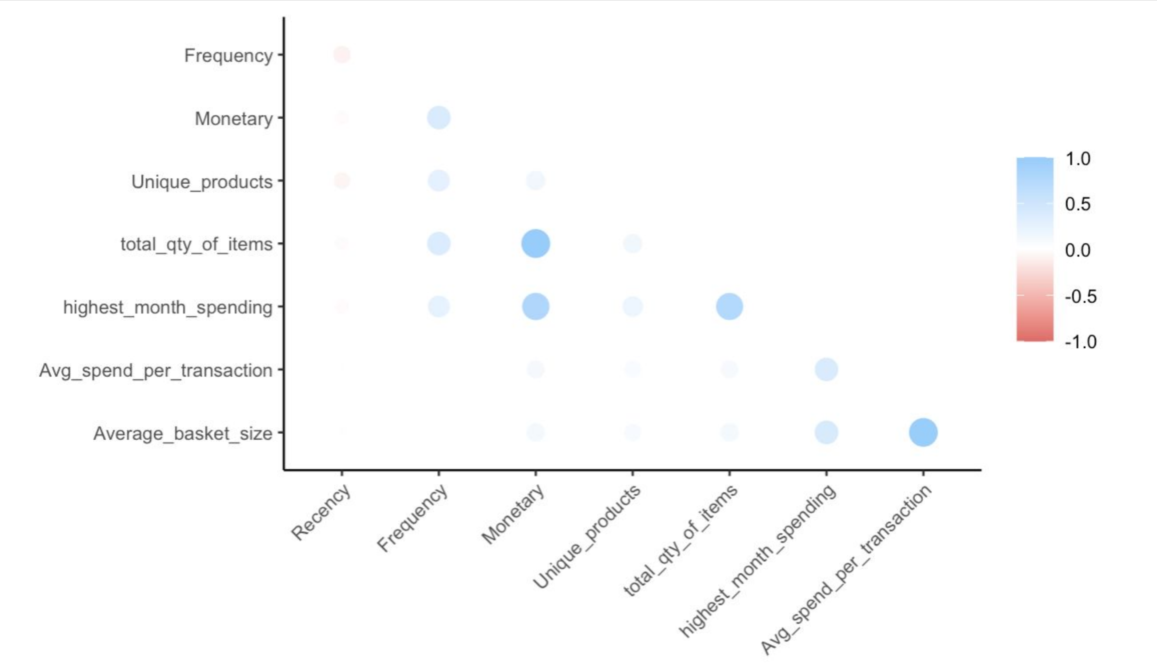
Source: SQL [?? x 12] Database: spark_connection

CustomerNo <int>	Recency <dbl>	Frequency <dbl>	Monetary <dbl>
15907	3	3	4766.57
16495	3	4	3277.35
14503	3	6	20398.81
16076	3	10	16260.48
17861	3	8	20308.57
18154	3	2	2363.58
18180	4	6	17879.32
15156	1	3	5097.06
13612	4	1	5521.15
17579	4	3	2537.87

Feature engineering - Additional columns

Column name	Description
Unique_products	Number of unique products bought
total_qty_of_items	Total quantity of products bought
month_with_max_spending	A number from 1-12, where 1 = January, 2 = February etc. Represents the month where the customer spent the most.
highest_month_spending	The monetary value during the month where the customer spent the most
Avg_spend_per_transaction	Monetary/Frequency
Average_basket_size	This is the average number of items purchased in a transaction

Feature selection - correlation analysis



03

Customer Modelling

Prepare data for modelling

Standardization

Due to the different units of measure and ranges of the variables.

```
#standardised values

rfm_stats <- ref_customer |>
  summarize(
    r_mean = mean(Recency),
    r_sd = sd(Recency),
    f_mean = mean(Frequency),
    f_sd = sd(Frequency),
    m_mean = mean(Monetary),
    m_sd = sd(Monetary)
  ) |> collect() #bring back to local r


ref_customer <- ref_customer |>
  mutate(R_standardized = (Recency - !!rfm_stats$r_mean) / !!rfm_stats$r_sd,
         F_standardized = (Frequency - !!rfm_stats$f_mean) / !!rfm_stats$f_sd,
         M_standardized = (Monetary - !!rfm_stats$m_mean) / !!rfm_stats$m_sd)

ref_customer |>
  sdf_describe(cols = c("R_standardized","F_standardized","M_standardized"))
```

Model 1: Customer loyalty

Dependent Variable: Logistic Duration (Proxy)

Duration : last date of purchase - first day of purchase

High loyalty customers: 1 (above median duration)
Low loyalty customers: 0 (below median duration)

**Independent Variables:
Monetary, Recency, Frequency**

How do these variables predict loyalty and which measure it most important?

Model 1: Customer loyalty

```
fit1_logistic<- ref_customer_split_train |>
  ml_logistic_regression(formula = logistic_duration ~ M_standardized + F_standardized + R_standardized)


fit1_logistic$summary$area_under_roc()

fit1_logistic |> tidy()
```

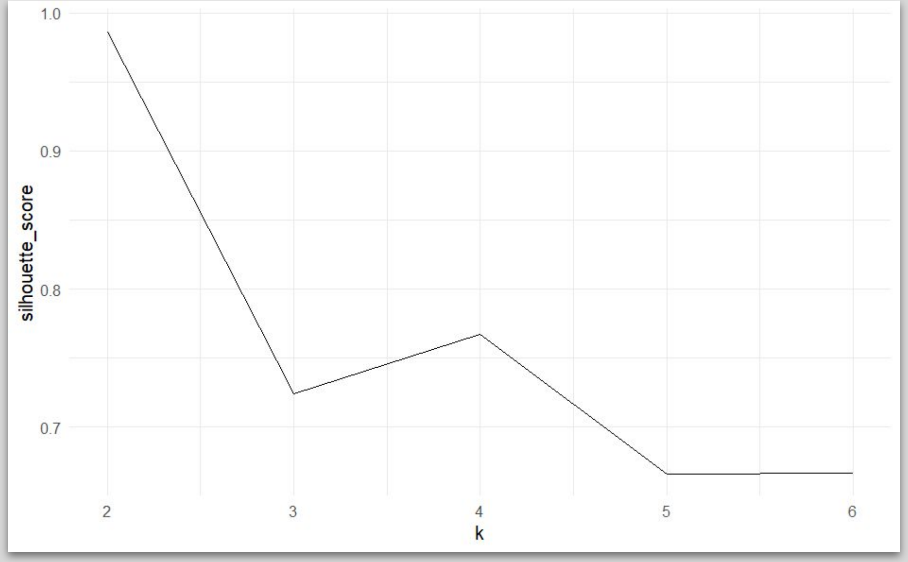
```
[1] 0.9362978
```

features <chr>	coefficients <dbl>
(Intercept)	1.7294993
M_standardized	-0.1302451
F_standardized	8.5105619
R_standardized	-0.6653812

Model 2: Customer Clustering

K-means clustering: uses multiple iterations to segment the unlabeled data points into “K” different clusters with similar properties.

Step 1: Selecting the best value of K based on the silhouette scores.
We chose K = 4



Model 2: Customer Clustering

Step 2: created a pipeline to generate 4 customer clusters based on the three standardized RFM variables and create a model.

```r
kmeans_pipeline <- ml_pipeline(sc) |>
  ft_vector_assembler(
    input_cols = c("Recency","Frequency","Monetary"),
    output_col = "features"
  ) |>
  ft_standard_scaler(
    input_col = "features",
    output_col = "features_stdz",
    with_mean = TRUE
  ) |>
  ml_kmeans(
    features_col = "features_stdz",
    prediction_col = "prediction",
    k=4,
    max_iter = 100,
    init_mode = "random",
    seed = 2001
  )

fitted_model <- ml_fit(kmeans_pipeline, ref_customer)


predictions <- ml_transform(fitted_model, ref_customer) |>
  collect()
```

Model 2: Customer Clustering

Step 3: calculated the mean of each cluster to numerically classify them into different groups

cluster <chr>	mean_recency <dbl>	mean_frequency <dbl>	mean_monetary <dbl>
0	66.08283	3.580787	9006.521
1	29.37500	77.187500	657592.798
2	35.02281	19.745247	58923.938
3	267.67701	1.551313	3574.195

Interpretation of the clusters

Cluster 0:

Relatively recent, but frequency and monetary low- could be newer customers

Cluster 1:

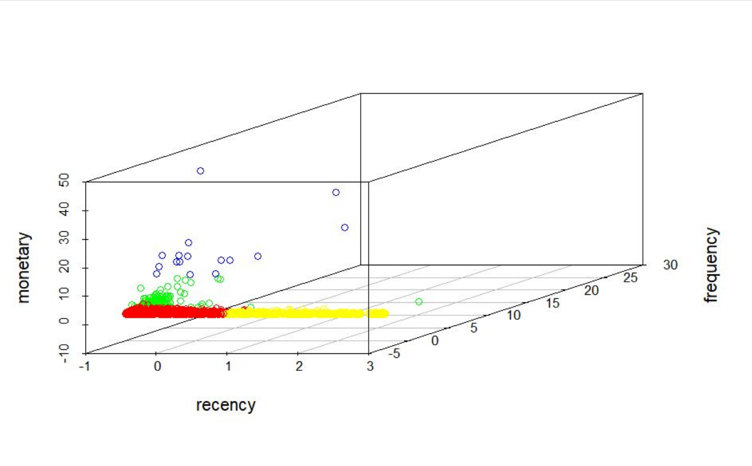
High frequency and Monetary and most recent customers - they are the most loyal and valuable customers

Cluster 2:

Relatively recent, but lower Monetary and Frequency than cluster 2: Potential to become part of most valuable.

Cluster 3:

Not very recent, with frequency close to 1 and low monetary value: lost customers or non recurring



04

Product EDA

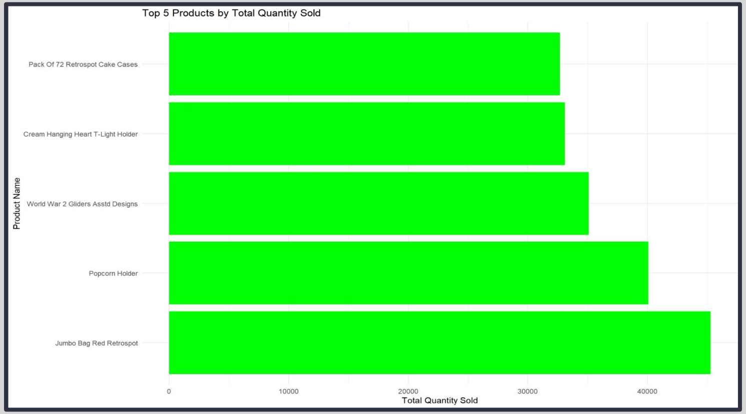
Ranking products - Top and Bottom 5

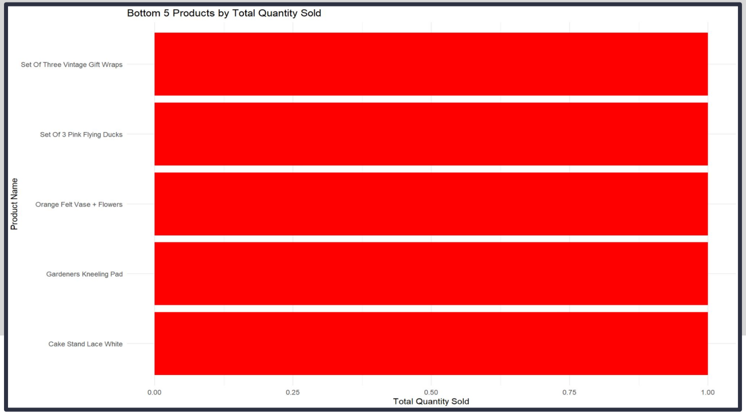
Grouping by product name and number, summarising by sum of quantity and collecting back to R environment for analysis

Insights: popular items are sold in **quantities up to 40k,** and the least popular items are only sold once

Code:

```
#group the products and sum the quantity for every product, then arrange in descending order based on quantity
ranked_products <- transaction_clean |>
  filter(Quantity >= 0 & !is.na(CustomerNo)) |>
  group_by(ProductName,ProductNo) |>
  summarise(Quantity_sold = sum(Quantity)) |>
  arrange(desc(Quantity_sold)) |>
  collect()

#choose the top 5 rows
top_5_products <- head(ranked_products,5)
#choose the bottom 5 rows
bottom_5_products <- tail(ranked_products, 5)

top_5_products
bottom_5_products
```





Quantity sold per month (Top 5)

Focused on the top 5 products and UK
Calculation of average price for each product
Grouped by month, country and product no/product name

ProductNo <chr>	ProductName <chr>	Country <chr>	month_sold <int>	Quantity <dbl>	Avg_Price <dbl>
22197	Popcorn Holder	United Kingdom	4	1653	11.241515
85099B	Jumbo Bag Red Retrospot	United Kingdom	12	3093	6.448385
21212	Pack Of 72 Retrospot Cake Cases	United Kingdom	11	2035	9.910551
22197	Popcorn Holder	United Kingdom	9	2842	11.333818
84879	Assorted Colour Bird Ornament	United Kingdom	2	1619	12.010833
85099B	Jumbo Bag Red Retrospot	United Kingdom	8	4768	6.313774
85099B	Jumbo Bag Red Retrospot	United Kingdom	7	3031	6.460230
21212	Pack Of 72 Retrospot Cake Cases	United Kingdom	2	1709	11.019885
84077	World War 2 Gliders Asstd Designs	United Kingdom	8	2067	10.575000
84077	World War 2 Gliders Asstd Designs	United Kingdom	6	1546	10.595161
85099B	Jumbo Bag Red Retrospot	United Kingdom	5	3303	6.429887
85099B	Jumbo Bag Red Retrospot	United Kingdom	2	2824	6.423097
22197	Popcorn Holder	United Kingdom	11	8075	9.606229
85099B	Jumbo Bag Red Retrospot	United Kingdom	9	4053	6.332621
22197	Popcorn Holder	United Kingdom	6	1989	11.311158
84077	World War 2 Gliders Asstd Designs	United Kingdom	5	3029	10.550769
84077	World War 2 Gliders Asstd Designs	United Kingdom	2	877	10.580588
21212	Pack Of 72 Retrospot Cake Cases	United Kingdom	9	2334	11.070294
22197	Popcorn Holder	United Kingdom	1	1680	11.432647
84879	Assorted Colour Bird Ornament	United Kingdom	1	1539	12.072740
84879	Assorted Colour Bird Ornament	United Kingdom	12	3469	10.250833
21212	Pack Of 72 Retrospot Cake Cases	United Kingdom	10	1239	11.049255

1-22 of 60 rows Previous 1 2 3 Next

Code:

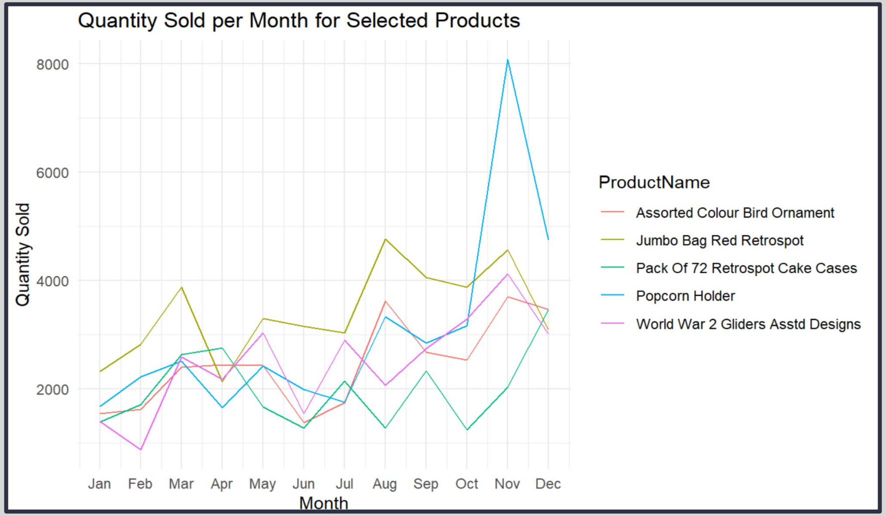
```
products_to_analyse = c("22197","84077","85099B","84879","21212")
country_to_analyse = "United Kingdom"
##########################################################################################
####

#This dataframe groups the data by month, product and countrry, and the values for Price
is aggregated to get the Average price since one item can have multiple different
prices, and quantity is aggregated with the sum function.
different_months <- transaction_clean |>
  filter(ProductNo %in% products_to_analyse,
         Country == country_to_analyse) |>
  mutate(month_sold = Month(FormattedDate)) |>
  group_by(ProductNo, ProductName, Country, month_sold) |>
  summarise(Quantity = sum(Quantity),
            Avg_Price = mean(Price)) |>
  collect()

#View the collected Spark dataframe
different_months
```

Quantity sold per month (Top 5)

Using ggplot2 to visualise a line graph showing the quantity sold per month for the top 5 products

Using scale_x_continuous for continuous data (months)

Insights: popcorn holders are extremely popular during November!



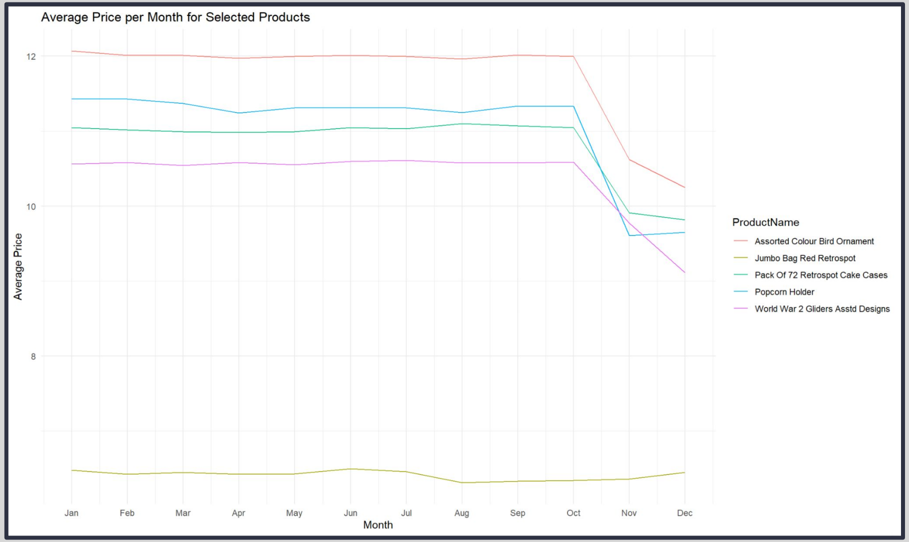
Code:

```
#Visualising the quantity sold every month for the top 5 products
ggplot(different_months, aes(x = month_sold, y = Quantity, color = ProductName)) +
  geom_line() +
  scale_x_continuous(breaks = 1:12, labels = month.abb[1:12]) +  # Set numeric months
and labels
  labs(x = "Month", y = "Quantity Sold") +
  ggtitle("Quantity Sold per Month for Selected Products") +
  theme(legend.position = "top") +
  theme_minimal()
```

Change in average prices of products every month

Using ggplot2 to visualise a line graph on the change in average prices for the top 5 products

Insights: all top products except jumbo bag red retrosport faced a decline during october, november, december (could be from discounts/clearing stocks)



Code:

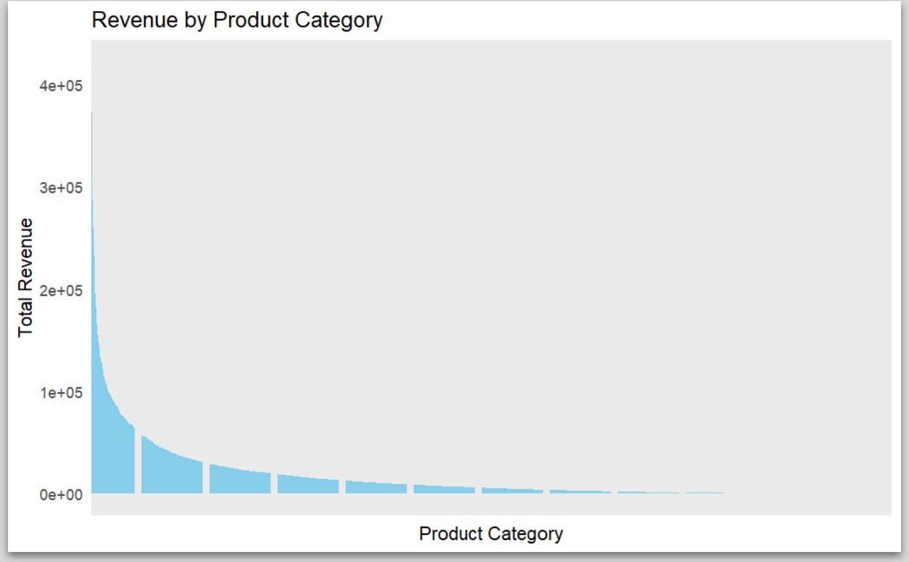
```
#Visualising the change in average prices of the products every month
ggplot(different_months, aes(x = month_sold, y = Avg_Price, color = ProductName)) +
  geom_line() +
  scale_x_continuous(breaks = 1:12, labels = month.abb[1:12]) +  # Set numeric months
and labels
  labs(x = "Month", y = "Average Price") +
  ggtitle("Average Price per Month for Selected Products") +
  theme(legend.position = "top") +
  theme_minimal()
```

05

Product Modelling

Understanding products' contribution

- As an ecommerce company, you would like to understand what contributes most to your revenue
- Using the dplyr package, we calculated the total revenue that each category collected over the period of the dataset
- With that visual analysis, we found out that a s**mall proportion of the product categories brought about majority of the revenue**



Feature engineering

Recency

Number of days since the last transaction. The higher it is, the longer it has been since the product has been sold.

Frequency

Total number of unique transactions per product.

Monetary

Total amount the product has generated through its sales

Average Transaction Quantity

The average volume of units sold for each product given any transaction.

Average Price

The average price that the product was sold for over the period.

Feature engineering

```
product_rfm <- transaction_clean |>
  dplyr::mutate(Revenue = Quantity * Price) |>
  group_by(ProductNo, ProductName) |>
  summarize(
    Price = mean(Price),
    Frequency = n(),
    AvgTransactionQuantity = mean(Quantity),
    Recency = as.numeric(datediff(max(FormattedDate), to_date("2019-12-31"))*(-1)),
    Monetary = sum(Revenue)
  ) |>
  arrange(desc(Monetary), desc(Frequency)) |> collect()
```

ProductNo	ProductName	Price	Frequency	AvgTransactionQuantity	Recency	Monetary	isTopProduct
22197	Popcorn Holder	10.807761	1416	28.304379	22	421648.36	1
85123A	Cream Hanging Heart T-Light Holder	13.078984	2333	14.181740	22	421536.89	1
84879	Assorted Colour Bird Ornament	11.638126	1489	21.768301	22	373419.54	1
84077	World War 2 Gliders Asstd Designs	10.269848	526	66.690114	22	360785.63	1
21212	Pack Of 72 Retrospot Cake Cases	10.770898	1370	23.850365	22	350497.83	1

Finding our “Top Products”

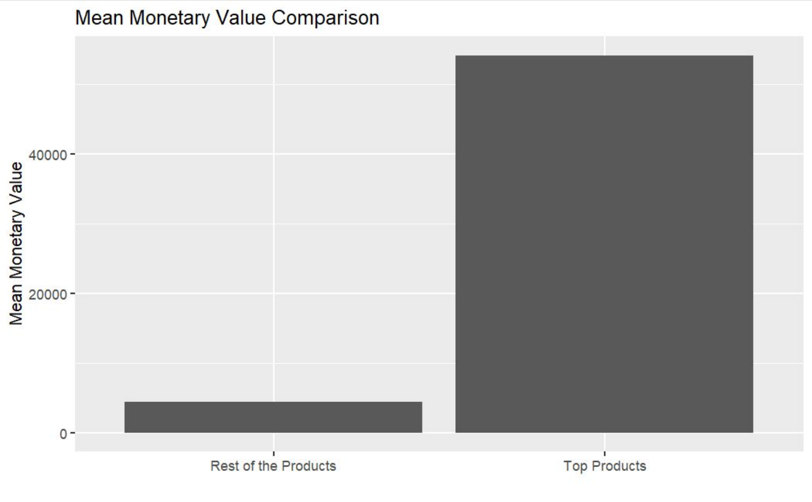
- One of our hypotheses was the Pareto Principle where 80% of outcomes are due to 20% of inputs.

- We hypothesize that 80% of our revenue was due to 20% of our products.

- We found that the **top 25% of our products contributed to 80% of our revenue**

- However, through KS-testing of **our revenue distribution by product category did not found it to be statistically significant for enough for a theoretical Pareto Distribution**.

- Nonetheless, a the top 24.8% products had a **statistically significant higher mean monetary value** through t-tests.



Conducting logistic regression

```
top_products_logistic_regression <- product_rfm_spark|>
  ml_generalized_linear_regression(
    formula = isTopProduct ~ Price + AvgTransactionQuantity + Frequency,
family = "binomial") |> tidy()
```

	term	estimate	std.error	statistic	p.value
1	(Intercept)	-7.95329042	0.320145578	-24.842731	0.000000e+00
2	Price	0.01340290	0.003125843	4.287772	1.804742e-05
3	AvgTransactionQuantity	0.18526359	0.010384815	17.839856	0.000000e+00
4	Frequency	0.03104502	0.001317810	23.558029	0.000000e+00

Conducting logistic regression

```
#creating training and testing validation
product_analysis_split <- product_rfm_spark|>
  sdf_random_split(training = 0.8, testing = 0.2, seed = 123)

top_product_analysis_train <- product_analysis_split$training

top_product_analysis_test <- product_analysis_split$testing

lr_fit <- top_product_analysis_train|> ml_logistic_regression(formula = 
isTopProduct ~ Price + AvgTransactionQuantity + Frequency)

lr_fit_summary<- ml_evaluate(lr_fit, dataset = top_product_analysis_test
)
lr_fit_summary$area_under_roc()
```
```

```
[1] 0.9801339
```
```

06

Conclusion

Conclusion

Utilized many models to interpret the e-commerce dataset in more detail

High-value products model	Customer segment model
Found the highest selling products	There are 4 key groups of customers
Product Strategy Focus on more popular products	Targeted Marketing for each customer segment

Project Roadmap

Task	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Introduction												
Apache file format												
Data wrangling												
Visualization & EDA												
Modeling												
ML Pipeline												

Thank you!

Do you have any questions?