IXIS Technical Exercise

Will Smialek

2024-05-16

Setup

```
knitr::opts_chunk$set(echo = TRUE, fig.align = 'center')

# Load libraries
pacman::p_load(tidyverse, openxlsx, lubridate)

# Set default theme for plotting
theme_set(theme_bw())

# Read Google Analytics datasets
adds_to_cart <- read.csv('DataAnalyst_Ecom_data_addsToCart.csv')
session_counts <- read.csv('DataAnalyst_Ecom_data_sessionCounts.csv')
```

Preprocessing

In order to create a data frame with month * device aggregation, the first step is to break the `dim_date` column into two parts (year and month).

```
session_counts <- session_counts |>

  # Change dim_date type from character to Date
  mutate(dim_date = as.Date(dim_date, format = '%m/%d/%y')) |>

  # Create columns for month and year
  mutate(dim_year = year(dim_date),
         dim_month = month(dim_date))

# Display preprocessed data
tibble(session_counts)
```

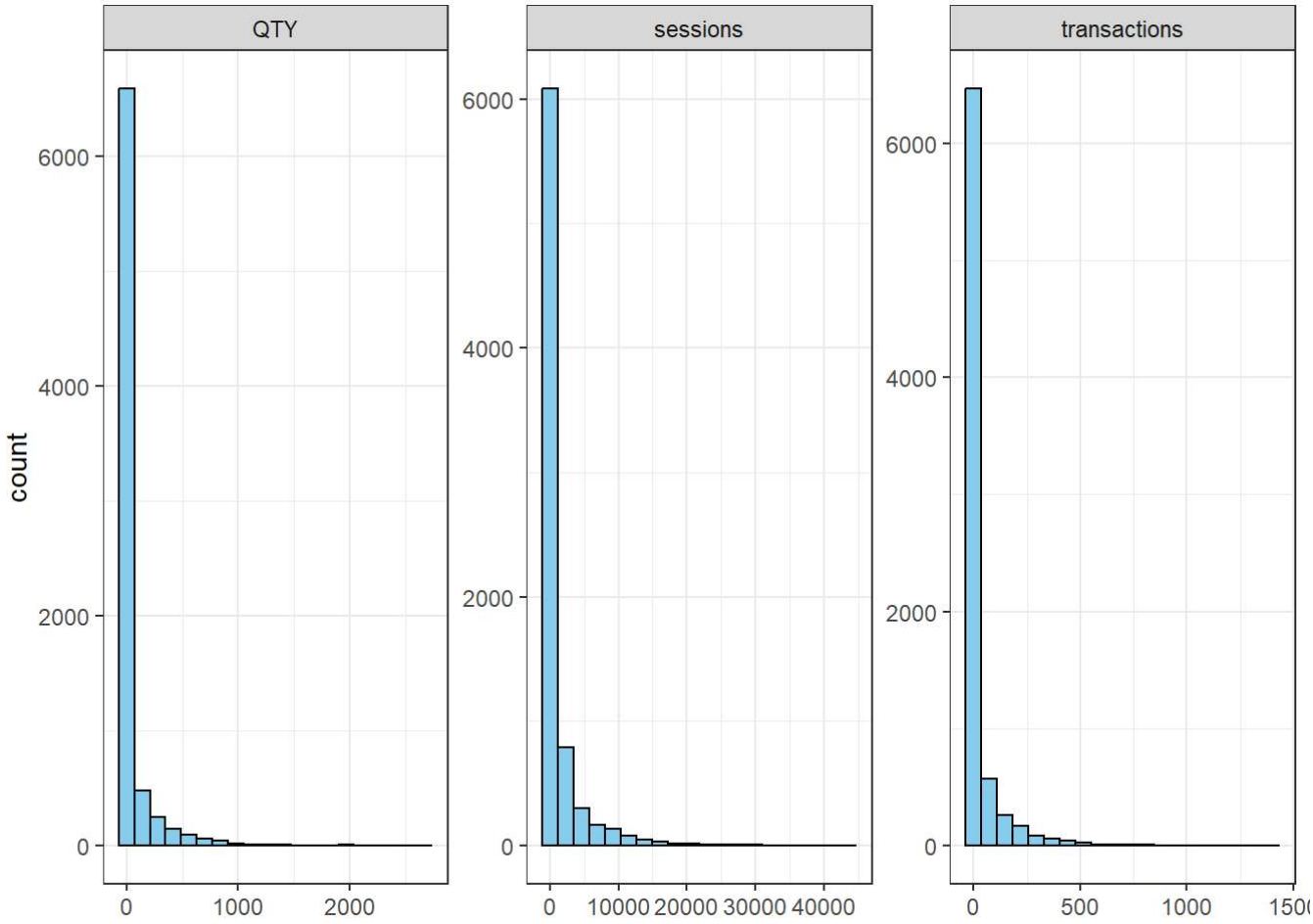
```
## # A tibble: 7,734 × 8
##    dim_browser       dim_deviceCategory dim_date   sessions transactions   QTY
##    <chr>             <chr>              <date>        <int>        <int> <int>
##  1 Safari            tablet             2012-07-01     2928          127   221
##  2 Internet Explorer desktop            2012-07-01     1106           28     0
##  3 Chrome            tablet             2012-07-01      474            3    13
##  4 Amazon Silk       tablet             2012-07-01      235            4     5
##  5 Internet Explorer mobile             2012-07-01      178            6    11
##  6 Internet Explorer tablet             2012-07-01      120            7     0
##  7 Android Browser   mobile             2012-07-01       10            0     0
##  8 error             desktop            2012-07-01        9            0     0
##  9 Edge              mobile             2012-07-01        5            0     0
## 10 Opera             mobile             2012-07-01        4            0     0
## # ℹ 7,724 more rows
## # ℹ 2 more variables: dim_year <dbl>, dim_month <dbl>
```

Before diving into aggregation and comparison, let's do some exploratory visualization.

```
session_counts |>

  # Reshape data to have one row for each feature-value pair
  pivot_longer(cols = sessions:QTY,
               names_to = 'Feature',
               values_to = 'Value') |>

  # Create histogram for each feature
  ggplot(aes(x = Value)) +
  geom_histogram(bins = 20, fill = 'skyblue', color = 'black') +
  facet_wrap(~ Feature, scales = 'free', nrow = 1) +
  labs(x = NULL)
```



These histograms are heavily skewed right, meaning there are lots of entries in `session_counts` with low values for quantity, sessions, and transactions, and a few very high values for each feature. In some data sets it may be beneficial to remove these outliers, but in this context we need not do so.

Aggregation

Now that `session_counts` has a year and month column, we can do month * device aggregation. First, let's explore the data a bit more.

```
table(session_counts$dim_deviceCategory)
```

```
##
## desktop  mobile  tablet
##    2672    3013    2049
```

This shows us that there are three categories of devices in our data frame. This means each month should have at most three device categories for each month. Now that we know what to expect, let's aggregate the data accordingly.

```
# Suppress friendly warning that appears when using `summarise` with more than one grouping
options(dplyr.summarise.inform = FALSE)

month_device_aggregated_data <- session_counts |>

  # Group by year, month, and device category
  group_by(dim_year, dim_month, dim_deviceCategory) |>

  # For each group take sum of sessions, transactions, and quantity
  summarise(sessions = sum(sessions),
            transactions = sum(transactions),
            QTY = sum(QTY)) |>

  # Calculate ECR for each group
  mutate(ECR = transactions / sessions)

# Display aggregated data
tibble(month_device_aggregated_data)
```

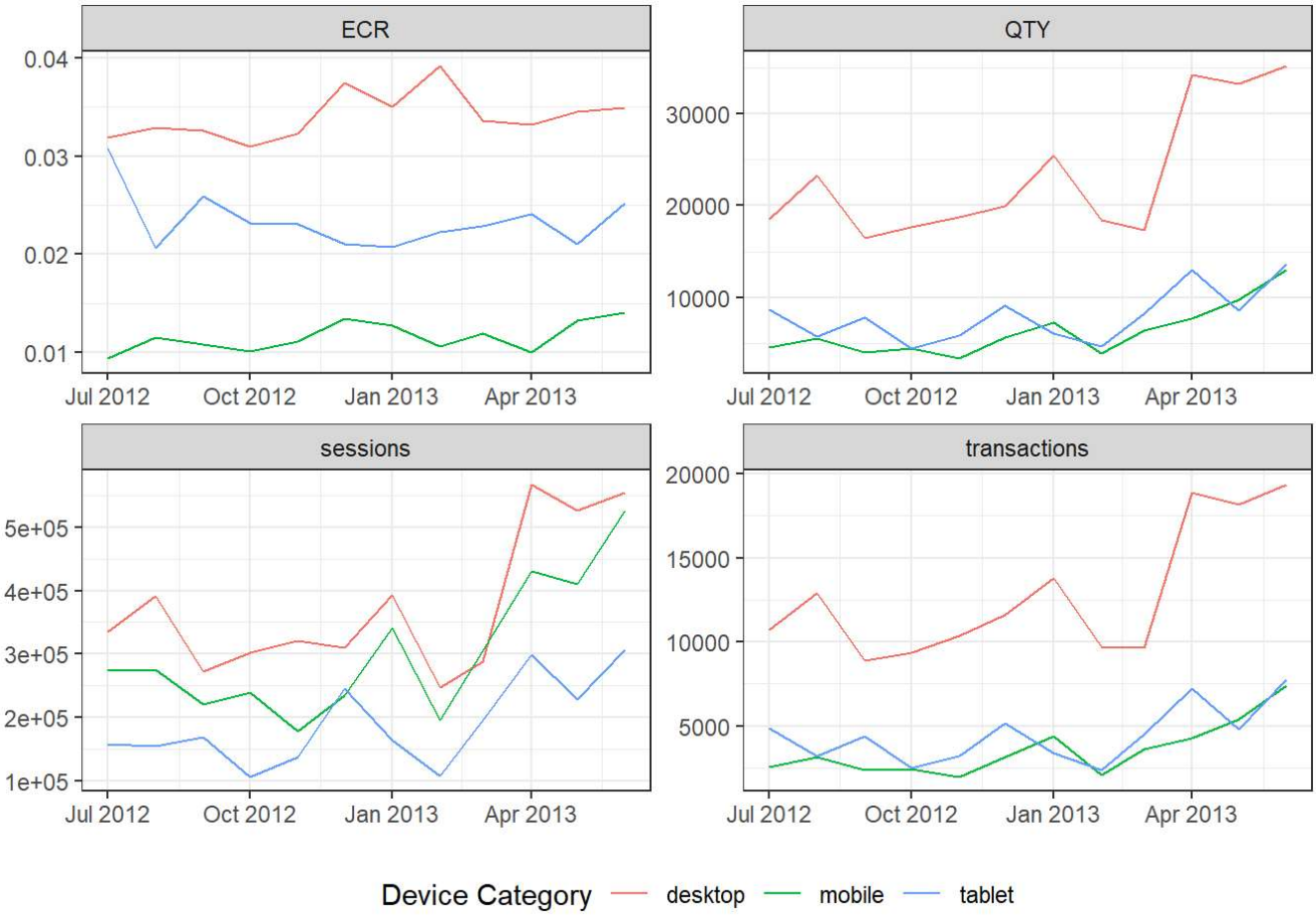
```
## # A tibble: 36 × 7
##    dim_year dim_month dim_deviceCategory sessions transactions   QTY     ECR
##       <dbl>     <dbl> <chr>                 <int>        <int> <int>   <dbl>
##  1     2012         7 desktop              335429        10701 18547 0.0319
##  2     2012         7 mobile               274443         2576  4557 0.00939
##  3     2012         7 tablet               158717         4884  8700 0.0308
##  4     2012         8 desktop              392079        12912 23316 0.0329
##  5     2012         8 mobile               275556         3165  5572 0.0115
##  6     2012         8 tablet               154858         3202  5760 0.0207
##  7     2012         9 desktop              272771         8898 16507 0.0326
##  8     2012         9 mobile               220689         2381  4050 0.0108
##  9     2012         9 tablet               169193         4379  7869 0.0259
## 10     2012        10 desktop              302682         9373 17675 0.0310
## # ℹ 26 more rows
```

Here we have 36 rows which contains the month * device aggregated data. Before moving on, let's visualize how each feature trends over time.

```
month_device_aggregated_data |>

  # Reshape data to have one row for each feature-value pair
  pivot_longer(cols = sessions:ECR,
               names_to = 'Feature',
               values_to = 'Value') |>

  # Create line plot for each feature. Group by device cateegory
  ggplot(aes(x = as.Date(paste(dim_year, dim_month, '1', sep = "/")),
             y = Value,
             color = dim_deviceCategory,
             group = dim_deviceCategory)) +
  geom_line() +
  facet_wrap(~ Feature, scales = 'free', nrow = 2) +
  labs(x = NULL, y = NULL, color = 'Device Category') +
  theme(legend.position = 'bottom')
```



This shows that quantity, sessions and transactions all seem to be trending upwards in the last few months; however, ECR seems to be at the same level throughout. Additionally, desktops seem to have the highest values for every feature.

Month Over Month Comparison

In order to compare the most recent two months, we must first determine which rows in the `session_counts` are the two most recent. Visually, it looks as if `session_counts` has been given to us sorted by date, ascending. However, we should write code that does not assume that the data has already been sorted.

```
groups(month_device_aggregated_data)
```

```
## [[1]]
## dim_year
##
## [[2]]
## dim_month
```

This shows us that `month_device_aggregated_data` currently has two groups, `dim_year` and `dim_month`. These are the exact groups we would like in order to summarize data by month. We will be able to include the `addsToCart` column from the second csv file as soon as we aggregate by month.

```
recent_two_months <- month_device_aggregated_data |>

  # Sum relevant data by month
  summarise(sessions = sum(sessions),
            transactions= sum(transactions),
            QTY = sum(QTY)) |>

  # Calculate ECR for each month
  mutate(ECR = transactions / sessions) |>

  # Perform a left join to include `addsToCart`
  left_join(adds_to_cart, by = c('dim_year', 'dim_month')) |>

  # Sort rows to have most recent months at the top
  arrange(desc(dim_year), desc(dim_month)) |>

  # After `summarise`, there is one grouping left
  # This grouping must be removed in order to slice properly
  ungroup() |>

  # Slice the top two rows
  slice(1:2)

# Display current data
tibble(recent_two_months)
```

```
## # A tibble: 2 × 7
##   dim_year dim_month sessions transactions   QTY    ECR addsToCart
##      <dbl>     <dbl>    <int>        <int> <int>  <dbl>      <int>
## 1     2013         6  1388834        34538 61891 0.0249     107970
## 2     2013         5  1164639        28389 51629 0.0244     136720
```

This shows we've successfully grabbed the most recent two months. Before we calculate direct comparisons between these two months, let's reorder them chronologically.

```
recent_two_months <- recent_two_months |>

  # Rearrange rows to be chronological
  arrange(dim_year, dim_month) |>

  # Create description column. This column will simply have which month it is
  # for our current two rows, but it will describe future rows differently
  mutate(description = paste(dim_month, dim_year, sep = '/')) |>

  # Reorder columns so that the description is first and
  # remove now-redundant date columns
  select(description, everything(), -dim_year, -dim_month)

# Display current data
tibble(recent_two_months)
```

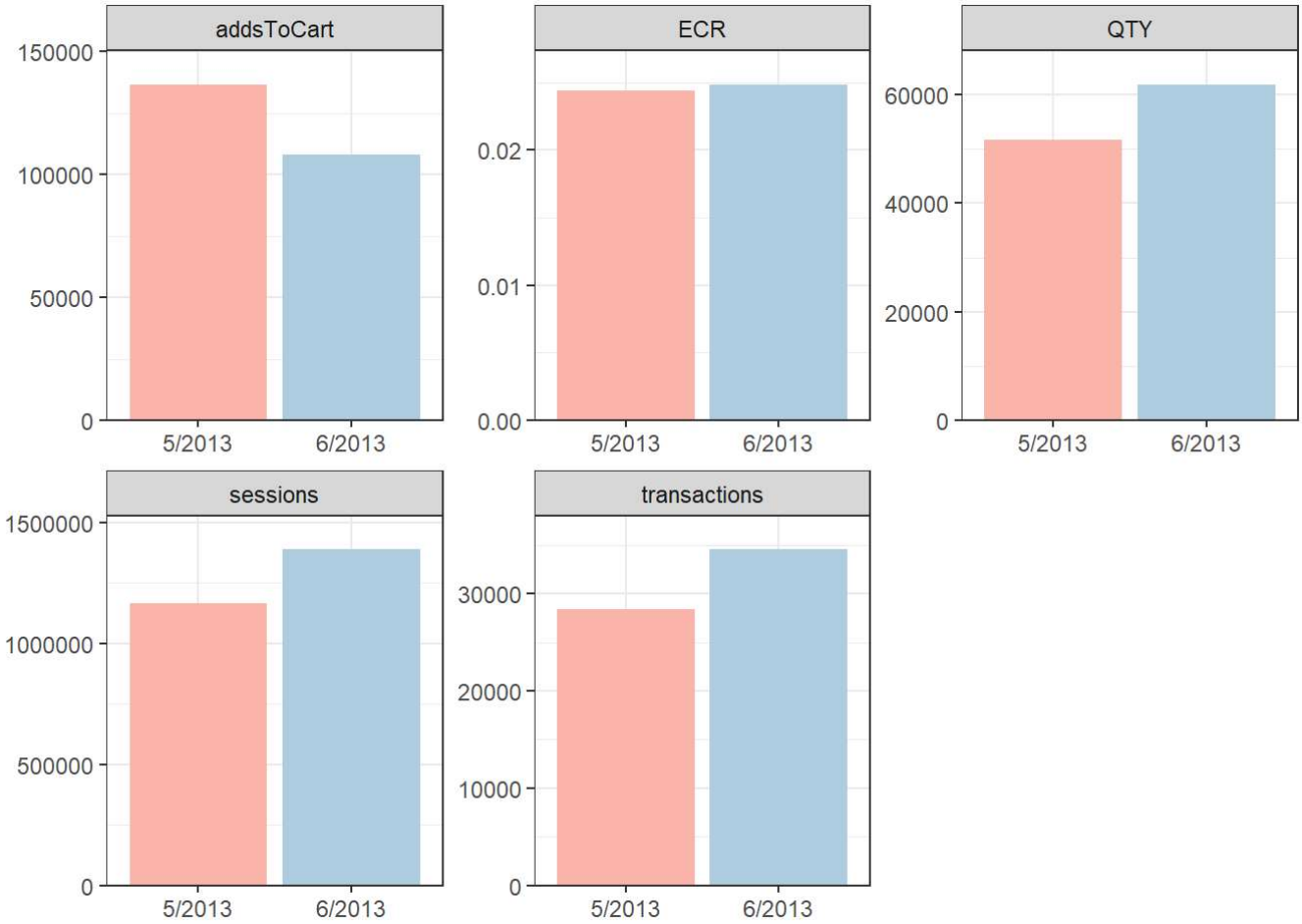
```
## # A tibble: 2 × 6
##   description sessions transactions   QTY    ECR addsToCart
##   <chr>          <int>        <int> <int>  <dbl>      <int>
## 1 5/2013       1164639        28389 51629 0.0244     136720
## 2 6/2013       1388834        34538 61891 0.0249     107970
```

We've combined the month and year columns for these two rows to serve as their description. Now we will be able to create new rows which describe diferent comparisons - absolute difference and relative difference. Before we do that, let's look at these two months side-by-side visually.

```
recent_two_months |>

  # Reshape data to have one row for each feature-value pair
  pivot_longer(cols = sessions:addsToCart,
               names_to = 'Feature',
               values_to = 'Value') |>

  # Create bar chart for each feature to show compare the last two months
  ggplot(aes(x = description, y = Value, fill = description)) +
  geom_bar(stat = 'identity', position = 'dodge') +
  facet_wrap(~ Feature, scales = 'free', nrow = 2) +
  scale_y_continuous(expand = expansion(mult = c(0, .1))) +
  scale_fill_brewer(palette = 'Pastel1') +
  labs(x = NULL, y = NULL) +
  theme(legend.position = 'none')
```



Here, we can see the difference between the most recent two months before doing any calculations. It seems that despite `addsToCart` decreasing significantly, the other features are increasing. Now we will actually perform these calculations. Absolute difference between a and b is written $|a - b|$. Relative difference between a and b is calculated as $\frac{b-a}{a}$.

```
# Create absolute difference row
absolute_difference <- recent_two_months |>
  summarise(
    description = 'absolute difference',

    # Calculate absolute difference for each numeric column
    across(where(is.numeric), ~ abs(.[1] - .[2]))
  )

# Create relative difference row
relative_difference <- recent_two_months |>
  summarise(
    description = 'relative difference',

    # Calculate relative difference for each numeric column
    across(where(is.numeric), ~ (.[2] - .[1]) / .[1])
  )

# Bind new rows to bottom of `recent_two_months`
month_over_month_comp <- rbind(recent_two_months, absolute_difference, relative_difference)

# Display month over month comparison
tibble(month_over_month_comp)
```

```
## # A tibble: 4 × 6
##   description            sessions transactions       QTY      ECR addsToCart
##   <chr>                     <dbl>        <dbl>     <dbl>    <dbl>      <dbl>
## 1 5/2013              1164639        28389     51629     0.0244   136720
## 2 6/2013              1388834        34538     61891     0.0249   107970
## 3 absolute difference  224195         6149     10262     0.000493  28750
## 4 relative difference       0.193        0.217     0.199 0.0202       -0.210
```

This shows the absolute and relative difference between the two most recent months. Note that while absolute difference has the same units as the first two rows, relative difference does not. The units of relative difference can be described as the *proportional change* between the two most recent months. We can take these results to mean that all features are trending upwards, with sessions, transactions, and quantity increasing by around 20% and ECR increasing by 2%. On the other hand, number of adds to cart decreased by more than 20%.

Create spreadsheet

Both output data frames have been created, `month_device_aggregated_data` and `month_over_month_comp`. All that's left to do is to create the Excel spreadsheet to store them.

```
wb <- createWorkbook()

# Add first sheet
addWorksheet(wb, 'Aggregated Data')
writeData(wb, 'Aggregated Data', month_device_aggregated_data)

# Add second sheet
addWorksheet(wb, 'Month Over Month Comparison')
writeData(wb, 'Month Over Month Comparison', month_over_month_comp)

# Add percentage style to apply to the relative difference row in the
# month over month comparison (for readability)
addStyle(wb,
         sheet = 'Month Over Month Comparison',
         rows = 5, # 4th row (+1 to account for header)
         cols = 1:ncol(month_over_month_comp),
         style = createStyle(numFmt = "0.00%"))

# For both sheets, set all column widths to be auto-calculated (for readability)
setColWidths(wb, sheet = 'Aggregated Data', cols = 1:ncol(month_device_aggregated_data), widths
= "auto")
setColWidths(wb, sheet = 'Month Over Month Comparison', cols = 1:ncol(month_over_month_comp), wi
dths = "auto")

# Write spreadsheet to `output.xlsx`
saveWorkbook(wb, 'output.xlsx', overwrite = TRUE)
```