Statics Dashboards Using ggplot

Maya Reese Farmer

October 21, 2021

Introduction I'm going to be constructing 2 static dashboards using the Sales Conversion Optimization dataset from kaggle.com
*this data was anonymously submitted

The sales dataset contains 1143 observations and has 11 columns with data on:

- ad_id: an unique ID for each ad.
- xyzcampaignid: an ID associated with each ad campaign of XYZ company.
- fbcampaignid: an ID associated with how Facebook tracks each campaign.
- age: age of the person to whom the ad is shown.
- gender: gender of the person to whim the add is shown
- interest: a code specifying the category to which the person's interest belongs (interests are as mentioned in the person's Facebook public profile).
- Impressions: the number of times the ad was shown.
- Clicks: number of clicks on for that ad.
- Spent: Amount paid by company xyz to Facebook, to show that ad.
- Total conversion: Total number of people who enquired about the product after seeing the ad.
- Approved conversion: Total number of people who bought the product after seeing the ad.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.5     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
## importing dataset into r
marketsale <- read_csv("KAG_conversion_data.csv")
```

```
## Rows: 1143 Columns: 11

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): age, gender
## dbl (9): ad_id, xyz_campaign_id, fb_campaign_id, interest, Impressions, Clic...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
## view data
head(marketsale,10)
```

```
## # A tibble: 10 x 11
##     ad_id xyz_campaign_id fb_campaign_id age   gender interest Impressions
##     <dbl>           <dbl>          <dbl> <chr> <chr>     <dbl>       <dbl>
##  1 708746             916         103916 30-34 M            15        7350
##  2 708749             916         103917 30-34 M            16       17861
##  3 708771             916         103920 30-34 M            20         693
##  4 708815             916         103928 30-34 M            28        4259
##  5 708818             916         103928 30-34 M            28        4133
##  6 708820             916         103929 30-34 M            29        1915
##  7 708889             916         103940 30-34 M            15       15615
##  8 708895             916         103941 30-34 M            16       10951
##  9 708953             916         103951 30-34 M            27        2355
## 10 708958             916         103952 30-34 M            28        9502
## # ... with 4 more variables: Clicks <dbl>, Spent <dbl>, Total_Conversion <dbl>,
## #   Approved_Conversion <dbl>
```

Data Visualization First I want to make one static dashboard that breaks down how the company's ads performed on Facebook.

```r
## create new data frame for first figure
library(dplyr)

## find the total amount spent for each ad campaign and
## total number of adds deployed by FB
marketsale %>%
    group_by(xyz_campaign_id) %>%
    summarise(sum(Spent))
```

```
## # A tibble: 3 x 2
##   xyz_campaign_id `sum(Spent)`
##             <dbl>        <dbl>
## 1             916         150.
## 2             936        2893.
## 3            1178       55662.
```

```
marketsale %>%
    count(xyz_campaign_id)
```

```
## # A tibble: 3 x 2
##   xyz_campaign_id     n
##             <dbl> <int>
## 1             916    54
## 2             936   464
## 3            1178   625
```

```
## create data frame
ad.spend = c(150, 2893, 55662)
ad.count = c(54, 464, 625)
ad.number = c("916", "936", "1178")
ad.info = tibble(ad.number, ad.spend, ad.count)
ad.info
```

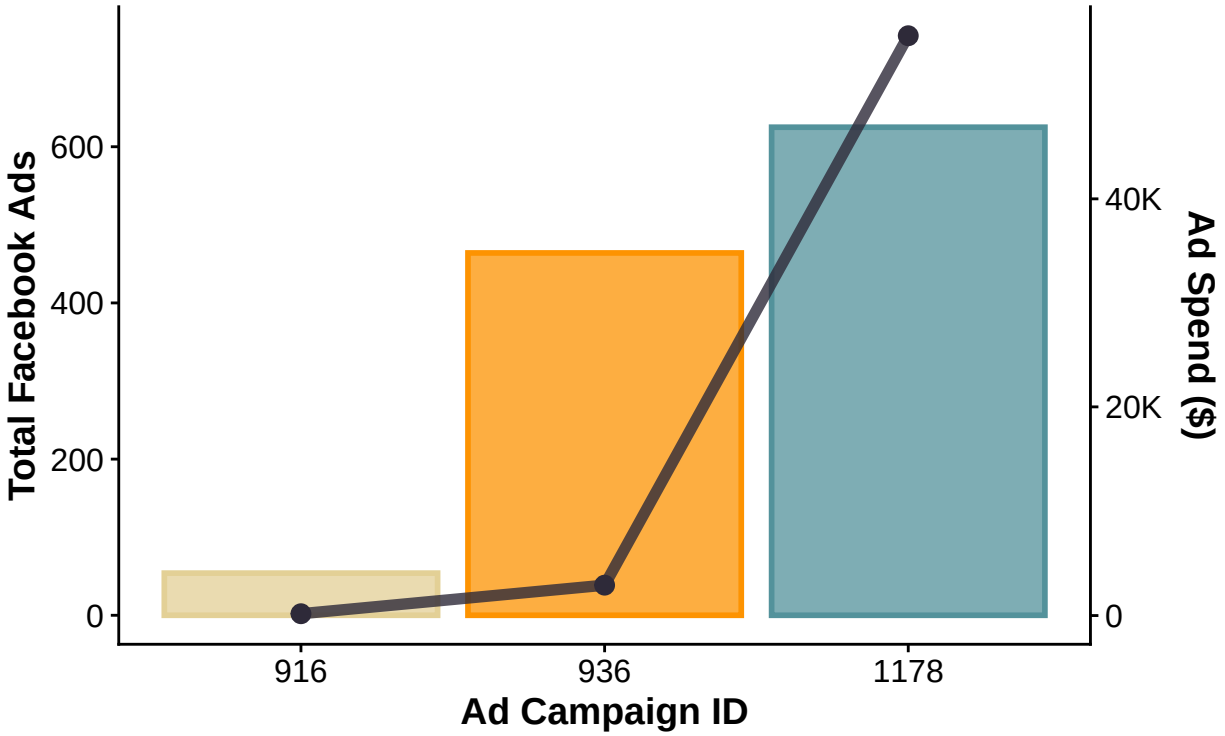
```
## # A tibble: 3 x 3
##   ad.number ad.spend ad.count
##   <chr>        <dbl>    <dbl>
## 1 916            150       54
## 2 936           2893      464
## 3 1178         55662      625
```

```
## create first figure using ad.info data
library(ggplot2)
library(cowplot)
library(scales)
```

```
##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

```
ad.breakdown = ggplot(ad.info) + geom_bar(aes(x = reorder(ad.number,
    ad.count), y = ad.count, fill = reorder(ad.number, ad.count),
    color = reorder(ad.number, ad.count)), stat = "identity",
    size = 1) + geom_line(aes(x = reorder(ad.number, ad.spend),
    y = ad.spend/75, group = 1), size = 2, color = "#2E2A39",
    stat = "identity", alpha = 0.8) + geom_point(aes(x = reorder(ad.number,
    ad.spend), y = ad.spend/75), color = "#2e2a39", size = 3) +
    scale_y_continuous(name = "Total Facebook Ads", sec.axis = sec_axis(~. *
        75, name = "Ad Spend ($)", labels = label_number_si())) +
    theme_cowplot(12) + scale_color_manual(values = c("#E3D096",
    "#FD9301", "#53929B")) + scale_fill_manual(values = alpha(c("#E3D096",
```

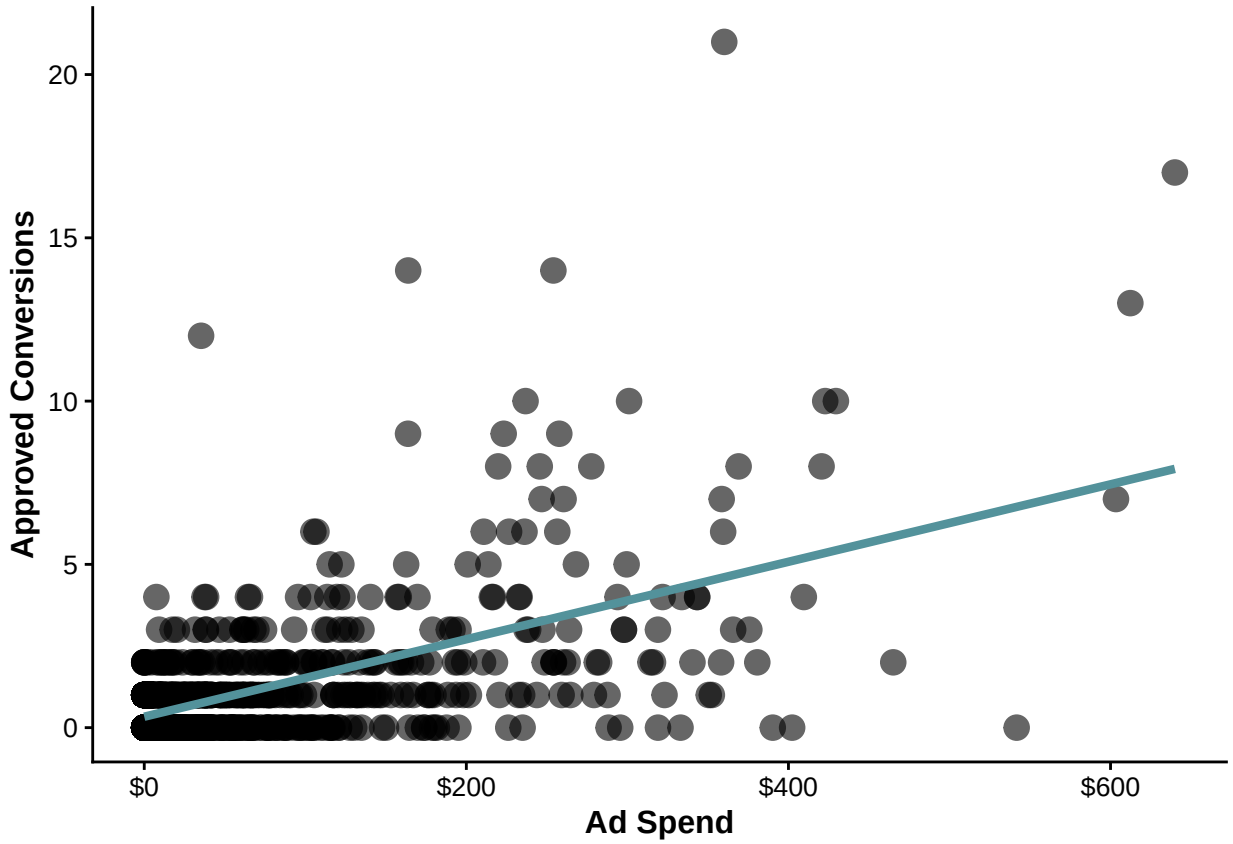
```
    "#FD9301", "#53929B"), 0.75)) + labs(x = "Ad Campaign ID",
    caption = "The bars represent the total number of Facebook ads deployed\n for each ad campaign. The
    theme(legend.position = "none", axis.title = element_text(size = 14,
        face = "bold"), axis.text = element_text(size = 12),
        axis.title.y.right = element_text(margin = margin(l = 10)),
        plot.caption = element_text(hjust = 0.5, size = 11))
ad.breakdown
```



The bars represent the total number of Facebook ads deployed for each ad campaign. The line and points represent the total amount spent on each ad campaign.

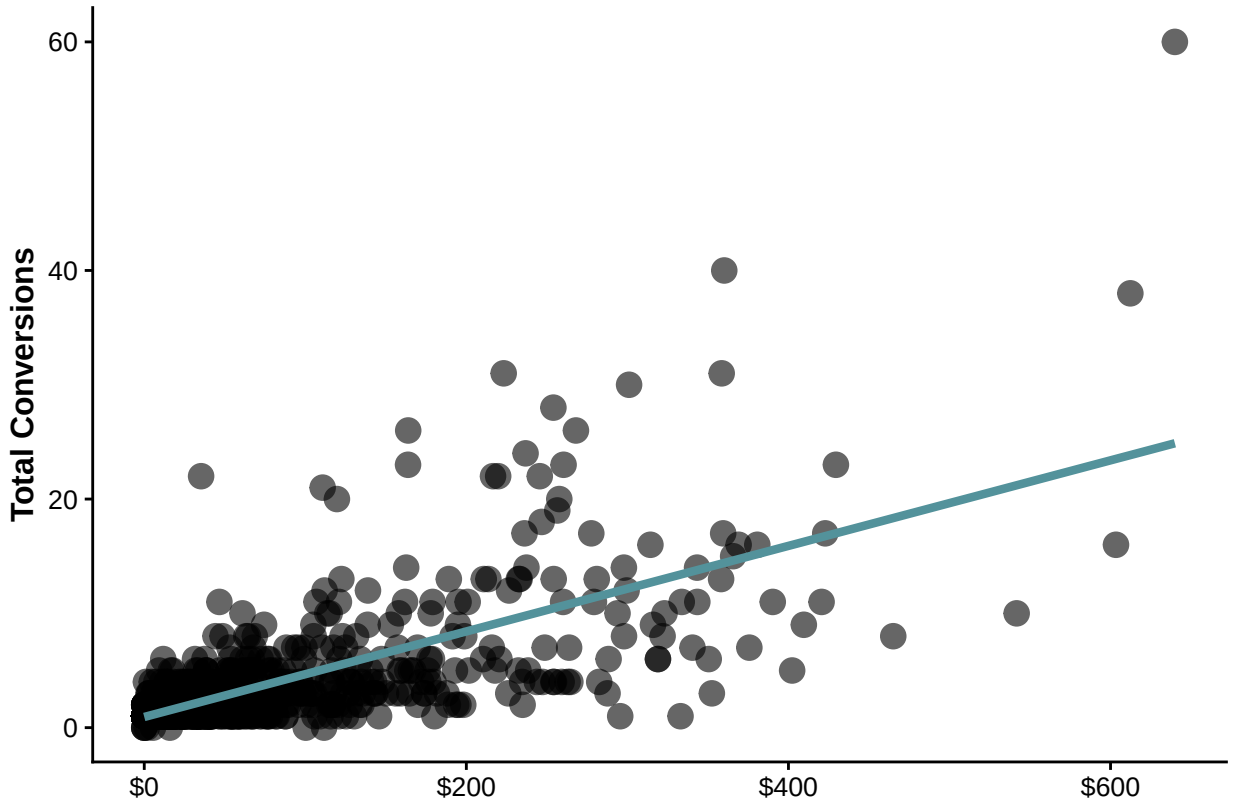
```
## create graph comparing ad spend to approved conversions
library(scales)

approve = ggplot(marketsale, aes(Spent, Approved_Conversion)) +
    geom_point(size = 4, alpha = 0.6) + geom_smooth(method = lm,
    se = FALSE, color = "#53929B", size = 1.5) + labs(x = "Ad Spend",
    y = "Approved Conversions") + theme_cowplot(12) + theme(axis.title = element_text(face = "bold",
    size = 12)) + scale_x_continuous(labels = dollar_format())
approve
```

```
## `geom_smooth()` using formula 'y ~ x'
```



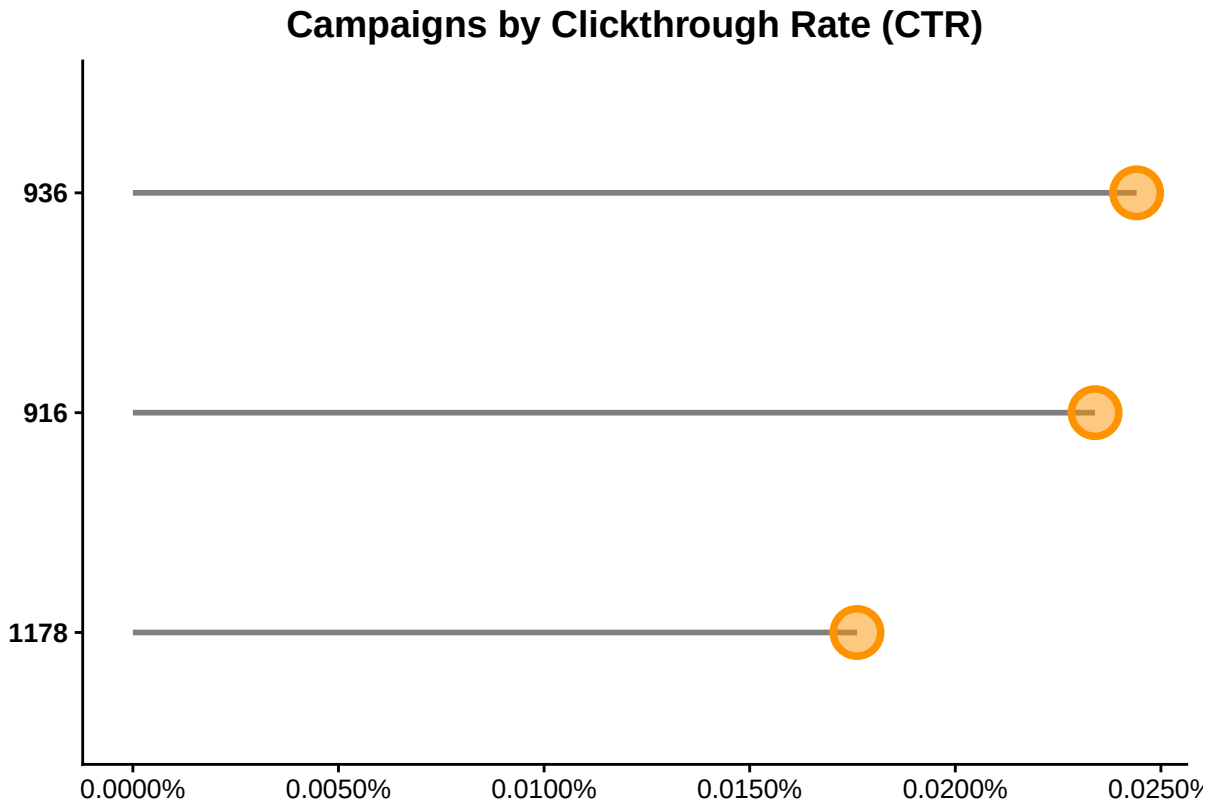

```
## create graph comparing ad spend to total conversions
total.con = ggplot(marketsale, aes(Spent, Total_Conversion)) +
    geom_point(size = 4, alpha = 0.6) + geom_smooth(method = lm,
    se = FALSE, color = "#53929B", size = 1.5) + labs(x = "",
    y = "Total Conversions") + theme_cowplot(12) + theme(axis.title = element_text(face = "bold",
    size = 12)) + scale_x_continuous(labels = dollar_format())
total.con
```

```
## `geom_smooth()` using formula 'y ~ x'
```

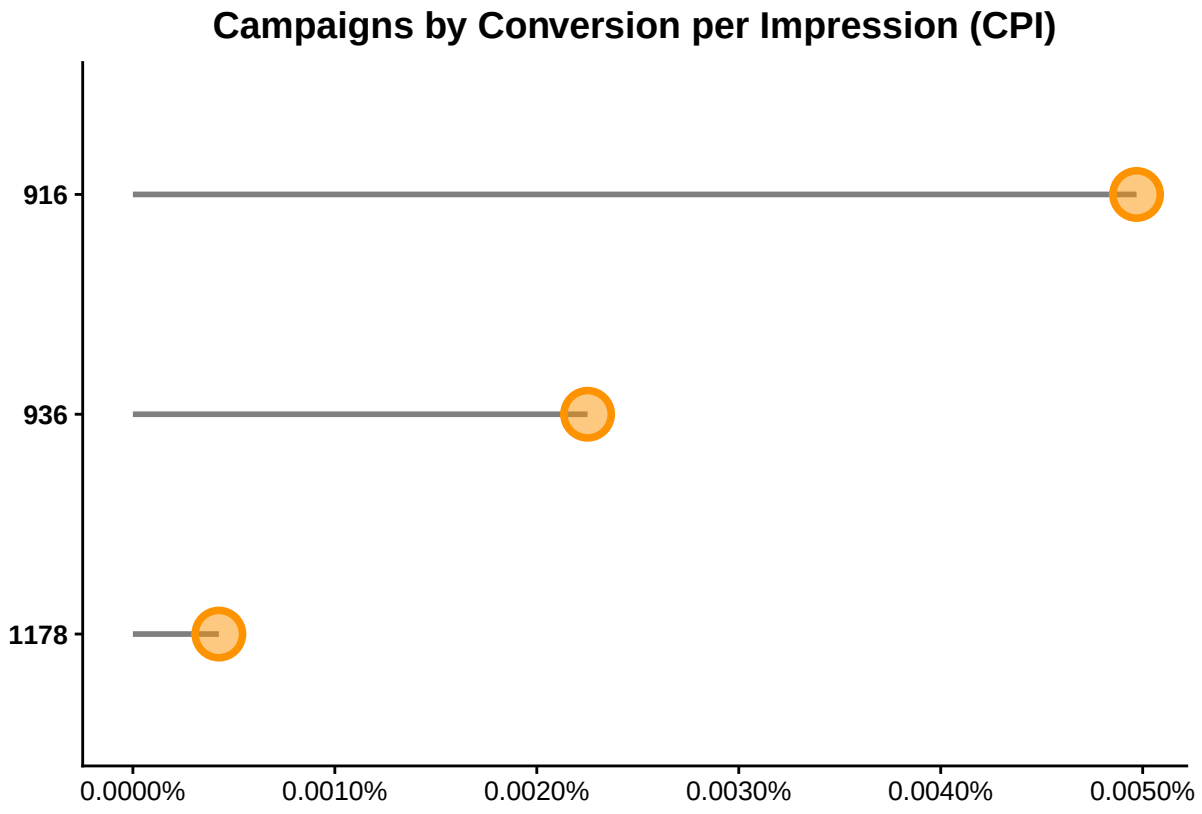



```
## create plots that display the clickthrough rate (CTR)
## and conversions per impression (CPI) for each ad
## campaign
ctr.plot = marketsale %>%
    group_by(xyz_campaign_id) %>%
    summarise(imp.sum = sum(Impressions), click.sum = sum(Clicks)) %>%
    mutate(CTR = click.sum/imp.sum) %>%
    ggplot(aes(x = reorder(xyz_campaign_id, CTR), y = CTR)) +
    geom_segment(aes(x = reorder(xyz_campaign_id, CTR), xend = reorder(xyz_campaign_id,
        CTR), y = 0, yend = CTR), color = "grey50", size = 1) +
    geom_point(size = 7, color = "#FD9301", fill = alpha("#FD9301",
        0.5), shape = 21, stroke = 2) + theme_cowplot(12) + labs(x = "",
    y = "", title = "Campaigns by Clickthrough Rate (CTR)") +
    theme(plot.title = element_text(size = 14, face = "bold",
        hjust = 0.5), axis.text.y = element_text(face = "bold")) +
    coord_flip() + scale_y_continuous(labels = percent_format())

cpi.plot = marketsale %>%
    group_by(xyz_campaign_id) %>%
    summarise(imp.sum = sum(Impressions), conv.sum = sum(Approved_Conversion)) %>%
    mutate(CPI = conv.sum/imp.sum) %>%
    ggplot(aes(x = reorder(xyz_campaign_id, CPI), y = CPI)) +
    geom_segment(aes(x = reorder(xyz_campaign_id, CPI), xend = reorder(xyz_campaign_id,
        CPI), y = 0, yend = CPI), color = "grey50", size = 1) +
    geom_point(size = 7, color = "#FD9301", fill = alpha("#FD9301",
        0.5), shape = 21, stroke = 2) + theme_cowplot(12) + labs(x = "",
    y = "", title = "Campaigns by Conversion per Impression (CPI)") +
    theme(plot.title = element_text(size = 14, face = "bold",
```

```
        hjust = 0.5), axis.text.y = element_text(face = "bold")) +
    coord_flip() + scale_y_continuous(labels = percent_format())

ctr.plot
```




```
cpi.plot
```




```
library(ggpubr)
```

```
##
## Attaching package: 'ggpubr'
```

```
## The following object is masked from 'package:cowplot':
##
##     get_legend
```

```
## arrange plots into static dashboard
conv.plots = ggarrange(total.con, approve, ncol = 1, nrow = 2,
    heights = c(1, 1))
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
top = ggarrange(ad.breakdown, conv.plots, nrow = 1, ncol = 2,
    widths = c(1, 0.6))
bottom = ggarrange(ctr.plot, cpi.plot, nrow = 1, ncol = 2)
web.dash = ggarrange(top, bottom, ncol = 1, nrow = 2, heights = c(1,
    0.6))
web.final = annotate_figure(web.dash, top = text_grob("Marketing Dashboard - Web Analytics",
    color = "#2e2a39", face = "bold", size = 28, just = "center",
    lineheight = 2), bottom = text_grob("Data source: kaggle.com",
    color = "#2e2a39", just = "right", x = 1, face = "italic",
    size = 10))
save_plot("webdash.pdf", web.final, base_height = 10, base_width = 10)
```

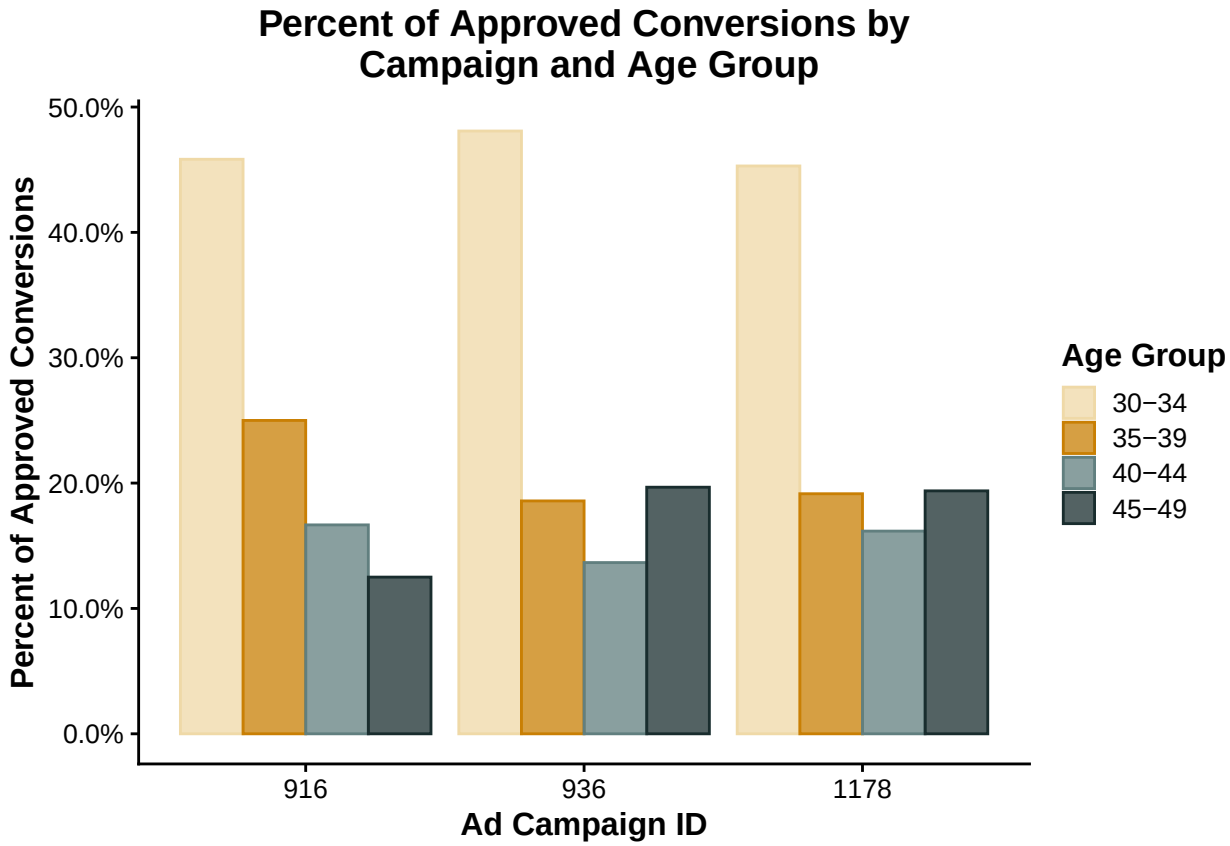
You can view the final web analytics dashboard as a pdf on the "PortfolioProjects" repository. It's saved as "webdash.pdf".

Next, I want to create a dashboard that looks at customer insights and determines what groups should potentially be targeted in the future.

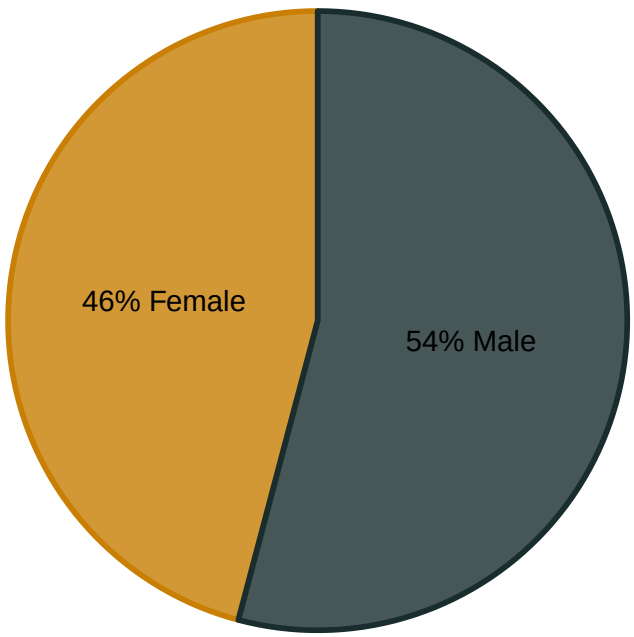
```
## create bar chart showing the percentage of approved
## conversions for each ad based on age group
customer1 = marketsale %>%
    group_by(xyz_campaign_id, age) %>%
    summarise(approve.sum = sum(Approved_Conversion)) %>%
    mutate(percent.approve = approve.sum/sum(approve.sum)) %>%
    ggplot(aes(x = as.factor(xyz_campaign_id), y = percent.approve,
        fill = age, color = age)) + geom_bar(stat = "identity",
    position = position_dodge()) + scale_fill_manual(values = alpha(c("#efd9a7",
    "#c97f04", "#617f7f", "#192d2e"), 0.75), name = "Age Group") +
    scale_color_manual(values = c("#efd9a7", "#c97f04", "#617f7f",
        "#192d2e"), name = "Age Group") + theme_cowplot(12) +
    scale_y_continuous(labels = scales::percent) + labs(x = "Ad Campaign ID",
    y = "Percent of Approved Conversions", title = "Percent of Approved Conversions by\n Campaign and Ag
    theme(plot.title = element_text(size = 14, face = "bold",
        hjust = 0.5), legend.title = element_text(size = 12,
        face = "bold"), legend.text = element_text(size = 10),
        axis.title = element_text(face = "bold"))
```

```
## `summarise()` has grouped output by 'xyz_campaign_id'. You can override using the `.groups` argument
```

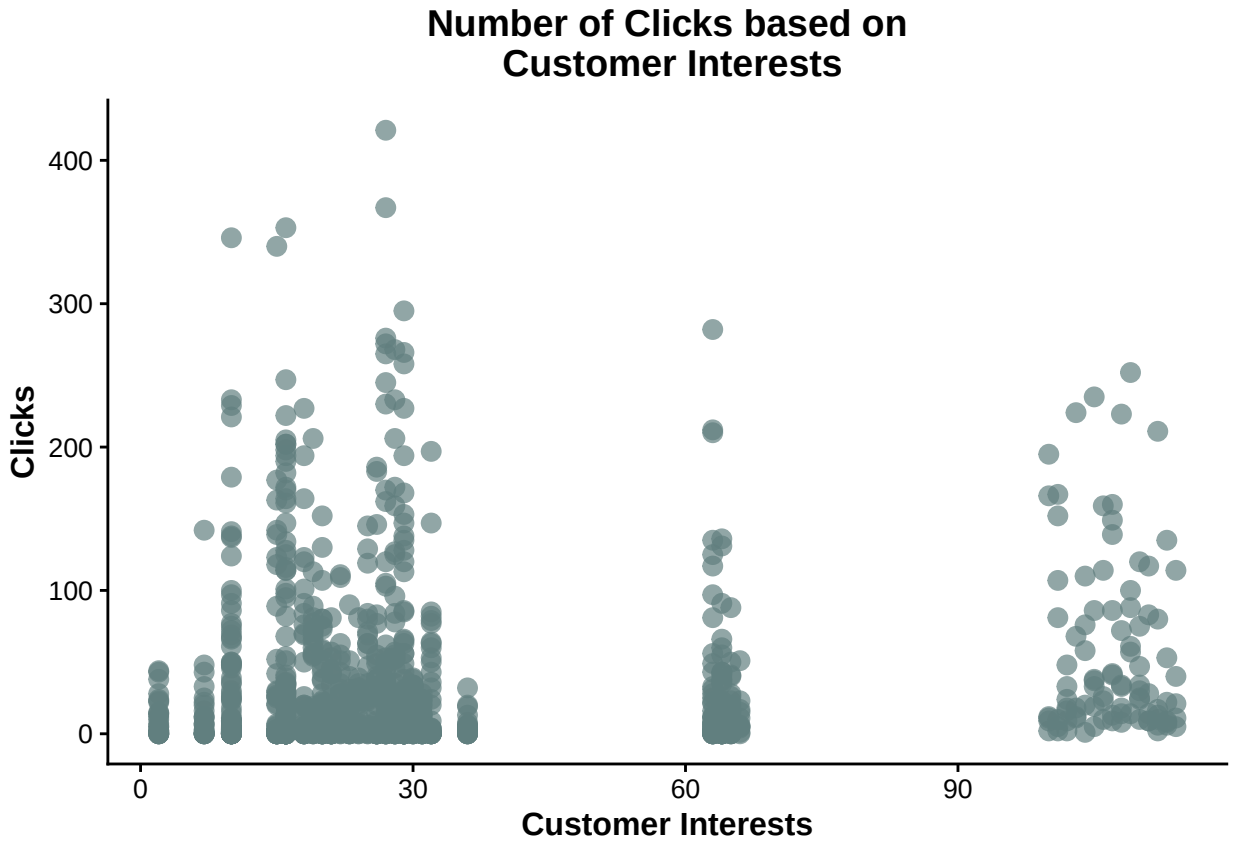
```
customer1
```



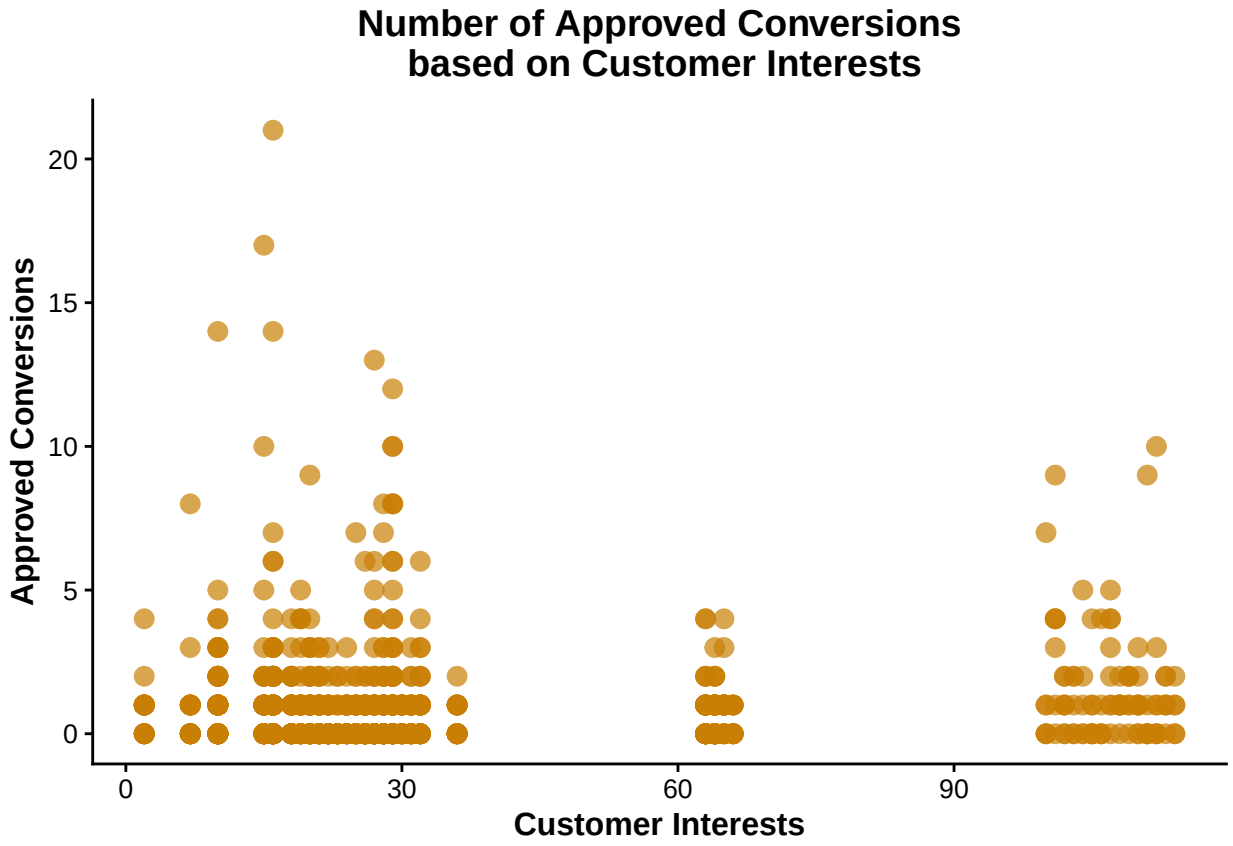

```
## change M and F in gender column to Male and Female
marketsale$gender[marketsale$gender == "F"] <- "Female"
marketsale$gender[marketsale$gender == "M"] <- "Male"

## create graph breaking down percent of approved
## conversions by gender
customer2 = marketsale %>%
    group_by(gender) %>%
    summarise(approve.sum = sum(Approved_Conversion)) %>%
    mutate(g.convert = approve.sum/sum(approve.sum) * 100) %>%
    ggplot(aes(x = "", y = g.convert, fill = gender, color = gender)) +
    geom_bar(stat = "identity", width = 1, size = 1) + coord_polar("y",
    start = 0) + theme_void() + theme(legend.position = "none") +
    scale_fill_manual(values = alpha(c("#c97f04", "#192d2e"),
        0.8)) + scale_color_manual(values = c("#c97f04", "#192d2e")) +
    geom_text(aes(label = paste0(round(g.convert), "% ", gender)),
        color = "black", size = 4, show.legend = FALSE, position = position_stack(vjust = 0.5)) +
    labs(title = "Approved Conversions by Gender\n for All Ad Campaigns") +
    theme(plot.title = element_text(size = 14, face = "bold",
        hjust = 0.5))

customer2
```




```
## create graph showing total number of clicks based on
## customer interests (on Facebook)
clicks.int = ggplot(marketsale, aes(x = interest, y = Clicks)) +
    geom_point(color = "#617f7f", alpha = 0.7, size = 3) + theme_cowplot(12) +
    labs(x = "Customer Interests", title = "Number of Clicks based on\n Customer Interests") +
    theme(axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(hjust = 0.5))

## create graph showing munber of approved interactions
## based on customer interests (on Facebook)
approve.int = ggplot(marketsale, aes(x = interest, y = Approved_Conversion)) +
    geom_point(color = "#c97f04", alpha = 0.7, size = 3) + theme_cowplot(12) +
    labs(x = "Customer Interests", y = "Approved Conversions",
        title = "Number of Approved Conversions\n based on Customer Interests") +
    theme(axis.title = element_text(size = 12, face = "bold"),
        plot.title = element_text(hjust = 0.5))

clicks.int
```


Number of Clicks based on
Customer Interests
Clicks
400
300
200
100
0
0
30
60
90
Customer Interests

```
approve.int
```




```
## arrange plots into static dashboard
customer.top = ggarrange(customer1,customer2,nrow = 1,ncol = 2,widths = c(1,0.5))
customer.bottom = ggarrange(clicks.int,approve.int,nrow = 1,ncol = 2)
customer.dash = ggarrange(customer.top, customer.bottom,ncol = 1,nrow = 2, heights = c(1,0.7))
customer.final = annotate_figure(customer.dash, top = text_grob("Marketing Dashboard - Customer Insights
save_plot("customerdash.pdf",customer.final,base_height = 10,base_width = 10)
```

You can view the final customer insights dashboard as a pdf on the "PortfolioProjects" repository. It's saved as "customerdash.pdf".