

# R Training Notes

## Lesson 4

*Stefanie Molin*

April 17, 2017

## I. Data Visualization

Now, we are going to build on the knowledge from the last session and learn how to use the **ggplot2** package to visualize data in conjunction with the packages we covered in prior lessons. Let's continue to work with the `dataframe_pivot` that we defined in the **dplyr** section as:

```
# load dplyr
library(dplyr)

# get date for 30 days ago
startDate <- Sys.Date() - 30

# query for last 30 days of client stats for [REDACTED]
query <- "
SELECT
  *
FROM [REDACTED]
WHERE
  day >= '%s'
  AND client_id = 4624
"

# query Vertica for data and store in dataframe [REDACTED]
[REDACTED] <- QueryVertica(username, sprintf(query, startDate), password)

# pivot dataframe
[REDACTED]_pivot <- [REDACTED] %>%
  select(day, displays, clicks, revenue, pc_conv = post_click_conversions,
         pc_sales = post_click_sales) %>%
  filter(as.Date(day) >= Sys.Date() - 25) %>%
  group_by(day) %>%
  summarize(total_clicks = sum(clicks, na.rm = TRUE),
            totalimps = sum(displays, na.rm = TRUE),
            spend = sum(revenue, na.rm = TRUE),
            conv = sum(pc_conv, na.rm = TRUE)) %>%
  mutate(ctr = total_clicks/totalimps, cpc = spend/total_clicks) %>%
  arrange(day)
```

```
# see first few rows
head(macys_pivot)
```

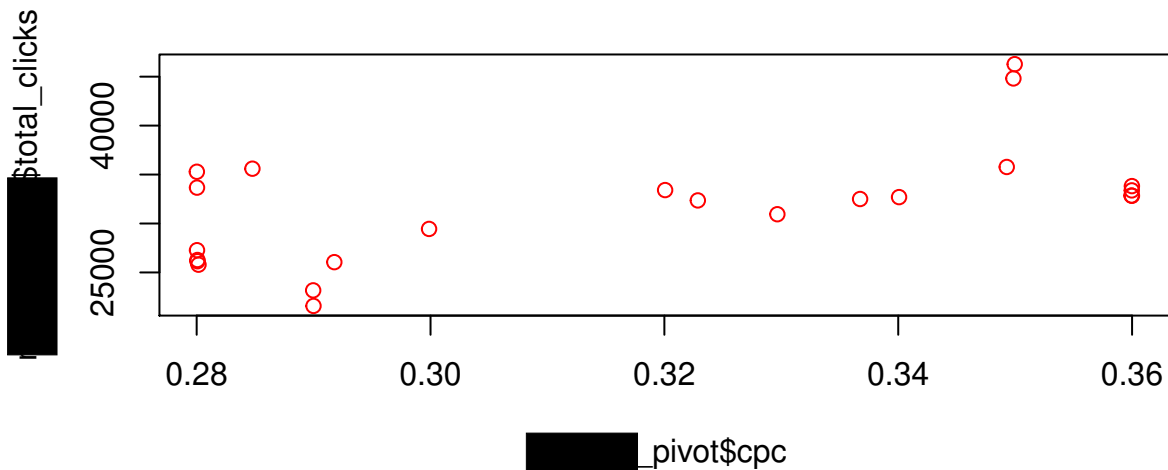
```
## # A tibble: 6 × 7
##       day total_clicks total_imps spend conv ctr cpc
##       <chr>      <dbl>      <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 2017-03-23  25000  100000  1000 0.01 0.01 0.01
## 2 2017-03-24  35000  150000  1500 0.01 0.01 0.01
## 3 2017-03-25  20000  80000  800 0.01 0.01 0.01
## 4 2017-03-26  25000  100000  1000 0.01 0.01 0.01
## 5 2017-03-27  25000  100000  1000 0.01 0.01 0.01
## 6 2017-03-28  45000  180000  1800 0.01 0.01 0.01
```

## A. Plotting with base R

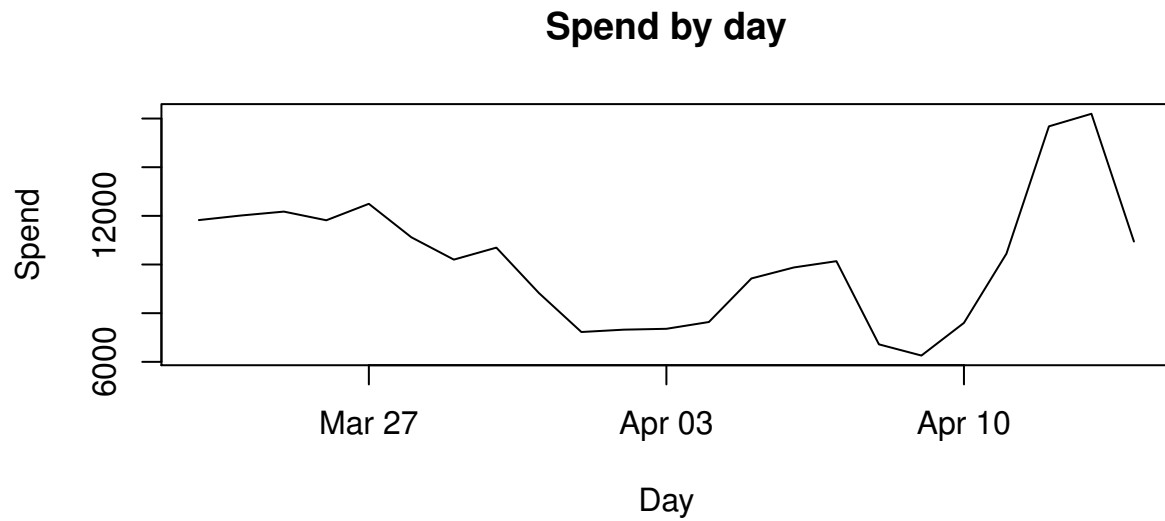
Before we get into using `ggplot2`, let's learn how to make some simple plots with base R.

`plot()`: can be used to make scatterplots, line graphs, and variations of the two.

```
# using the macys data, create a red scatterplot of clicks and cpc
plot(macys_pivot$cpc, macys_pivot$total_clicks, col = "red")
```

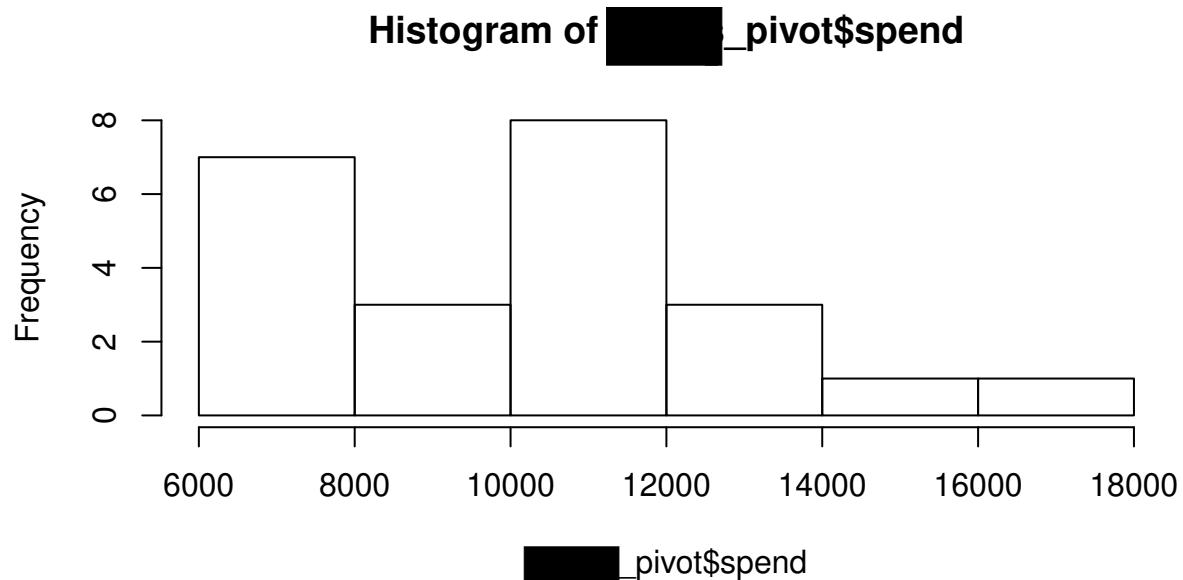


```
# plot spend by day with labels
plot(as.Date(█_pivot$day), █_pivot$spend, type = "l",
     main = "Spend by day", xlab = "Day", ylab = "Spend")
```



hist(): make a histogram

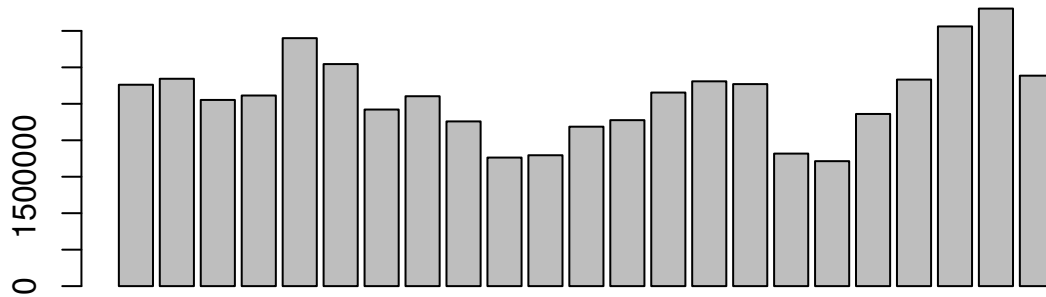
```
# histogram of spend the last 25 days
hist(█_pivot$spend)
```



```

barplot(): draw a bar chart
# bar chart of impressions
barplot(█_pivot$totalimps)

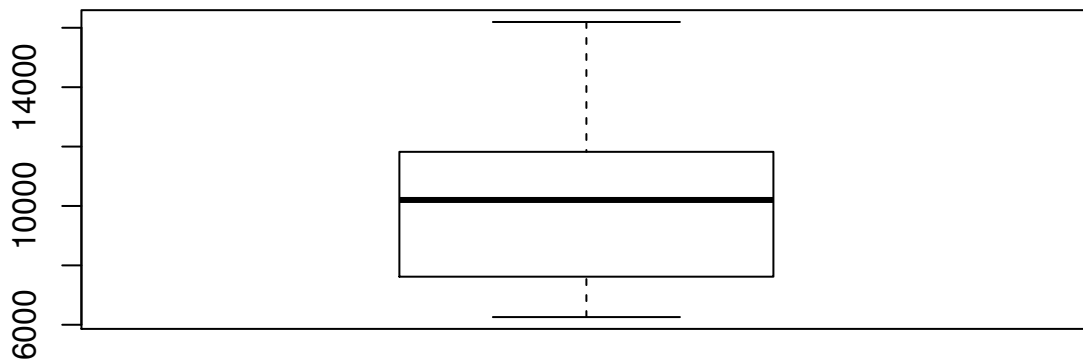
```



```

boxplot(): box-and-whisker plot
# box-and-whisker plot of spend
boxplot(█_pivot$spend)

```



We won't go into how to change point shape, color, line type, etc. You can read up about that in the documentation for each function.

## B. ggplot2

Base R plots are pretty basic and not that pretty, so they are more used to quickly get a sense of any patterns in the data you are dealing with but not for sharing with other parties. To make graphs fit for sharing, we will use the popular `ggplot2` package. This package is based on the grammar of graphics; you tell `ggplot2` how to map your variables and what options to use and `ggplot2` builds the plot. Note that is is just a sampling of what you can do with `ggplot2`, you can find plenty more plot types, customizations, etc. in the package documentation.

First, we load the packages:

```
# for plotting  
library(ggplot2)
```

### 1. Basic syntax

```
# basic ggplot usage with geom and ... for additional aesthetics (col, fill, size, etc.)  
df %>% ggplot(aes(x = col_name_1, y = col_name_2, ...)) +  
  geom_*
```

### 2. Aesthetics layer (`aes()`)

The aesthetics layer tells `ggplot` how to map our variables. We are going to cover the most common here.

x, y: map variables to the x and y axes

col: determine the colors based on unique values of this variable

fill: does the same as color but for bar charts

size: determine the size of points based on the value of this variable

### 3. Common geoms

The geom layer tells ggplot how to display the mapped variables.

`geom_point()` – scatterplot

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
# show days of the week as different colors sized by spend (just to illustrate use)

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks,
                             col = factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                                             levels = c("Sunday", "Monday", "Tuesday",
                                                         "Wednesday", "Thursday", "Friday",
                                                         "Saturday")),
                             size = spend)) +

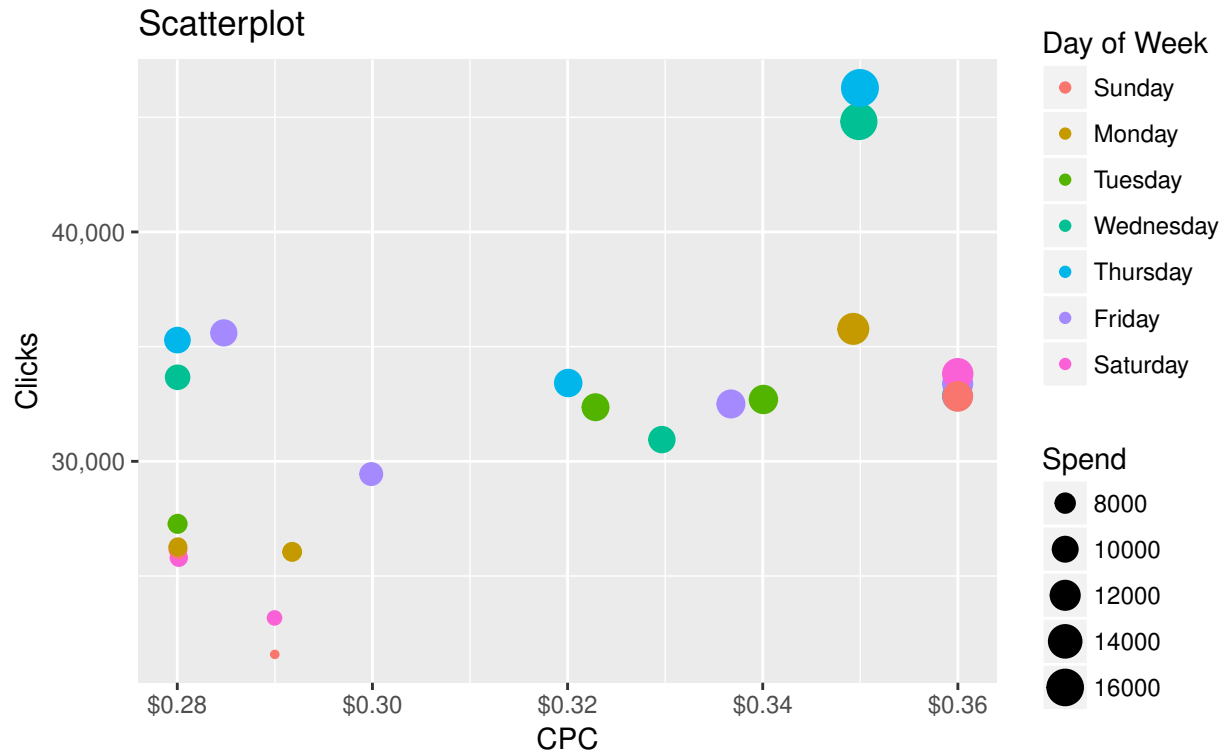
# add points
geom_point() +

# optional stuff to make it cleaner
# format labels
scale_x_continuous(labels = scales::dollar) +
scale_y_continuous(labels = scales::comma) +

# add title
ggtitle("Scatterplot") +

# fix axis labels
xlab("CPC") +
ylab("Clicks") +

# fix legend titles (match to aes labels)
labs(col = "Day of Week", size = "Spend")
```



That may look more complicated, but it is very readable and easy to add onto. Notice how we used **factors** here to show the days of the week in the order we wanted (using **levels**). We also cleaned up the axis labels and legend title since they get labeled exactly what you put into that value as the aesthetics, which can be confusing in the case of the `col` argument [`factor(format(as.Date(████_pivot$day), "%A"), levels = c("Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"))`]. The “`::`” notation in `scale_x_continuous(labels = scales::dollar)` is not required, but it helps to see where the function the code is using came from.

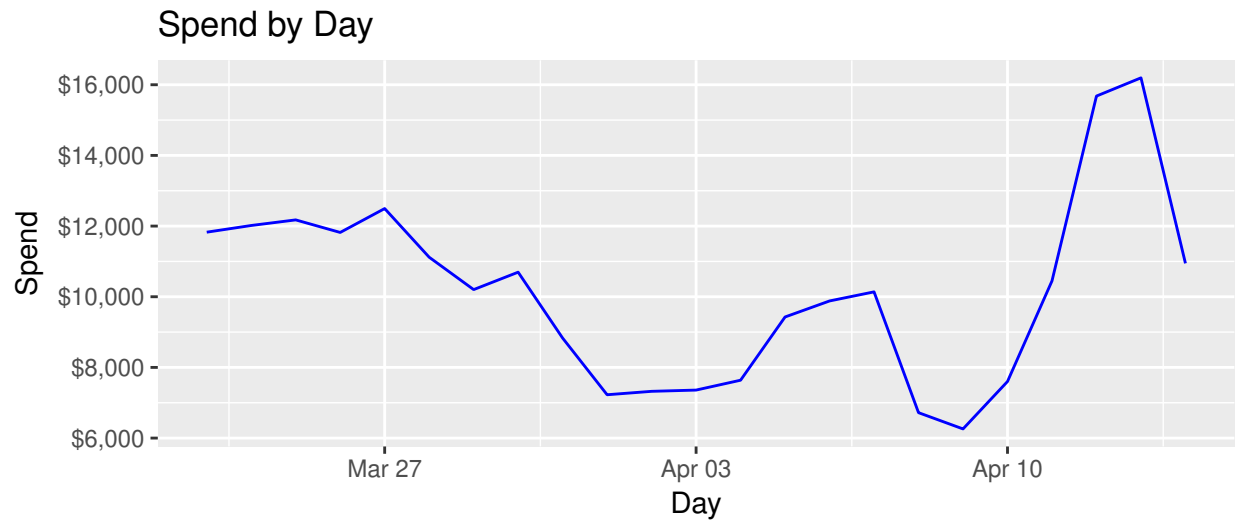
`geom_line()` – line graph

```
# spend by day for █████
# define aesthetics
████_pivot %>% ggplot(aes(x = as.Date(day), y = spend)) +
  # add line
  geom_line(col = "blue") +

  # optional stuff to make it cleaner
  # format labels
  scale_y_continuous(labels = scales::dollar) +

  # add title
  ggtitle("Spend by Day") +

  # fix axis labels
  xlab("Day") +
  ylab("Spend")
```



```

geom_histogram() – histogram
# histogram of [redacted] spend

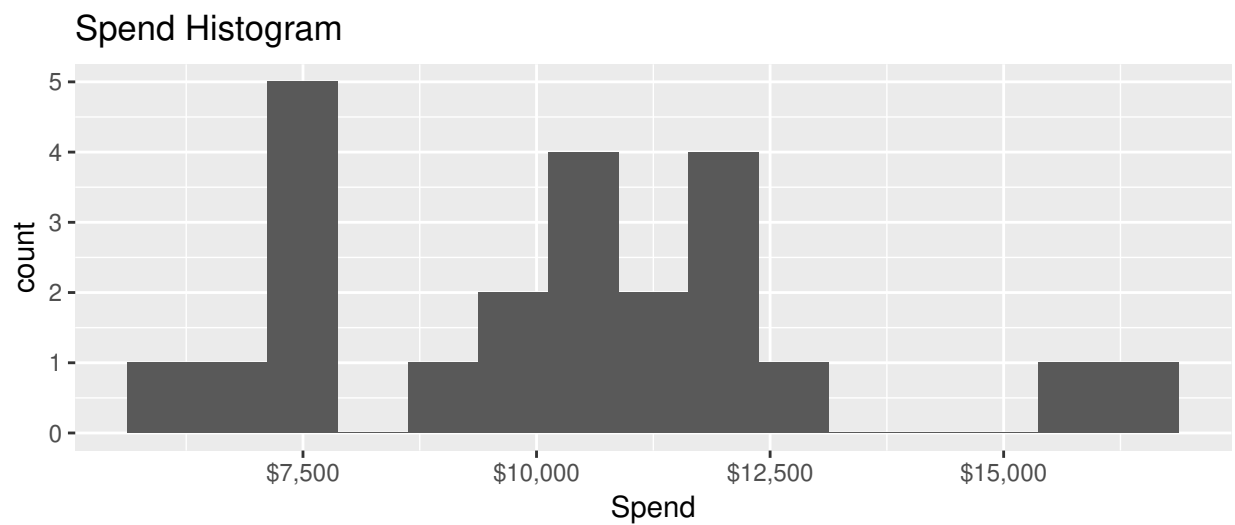
# define aesthetics
[redacted]pivot %>% ggplot(aes(x = spend)) +
  # add histogram
  geom_histogram(binwidth = 750) +

  # optional stuff to make it cleaner
  # format labels
  scale_x_continuous(labels = scales::dollar) +

  # add title
  ggtitle("Spend Histogram") +

  # fix axis labels
  xlab("Spend")

```





```

geom_bar() – bar chart of count geom_col() – bar chart of values
# bar chart counting days of week in this data set

# define aesthetics
pivot %>%
  ggplot(aes(x = as.Date(day), y = totalimps,
             fill = factor(
               ifelse(format(as.Date(pivot$day), "%A") %in%
                 c("Sunday", "Saturday"),
                 "Weekend", "Weekday")))) +

# add bars
geom_col() +

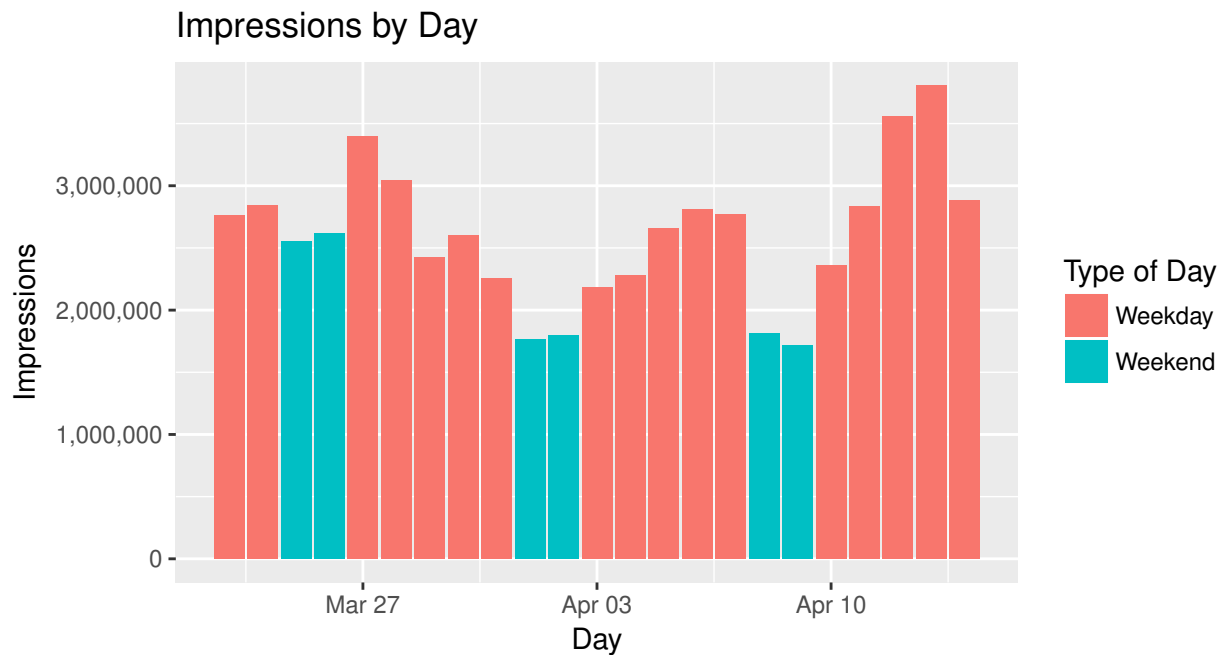
# optional stuff to make it cleaner
# format labels
scale_y_continuous(labels = scales::comma) +

# add title
ggtitle("Impressions by Day") +

# fix axis labels
xlab("Day") +
ylab("Impressions") +

# fix legend title
labs(fill = "Type of Day")

```



```

geom_boxplot() – box and whisker plot
# ██████ spend boxplot

# define aesthetics
██████_pivot %>%
  ggplot(aes(x = factor(1L), y = spend)) +

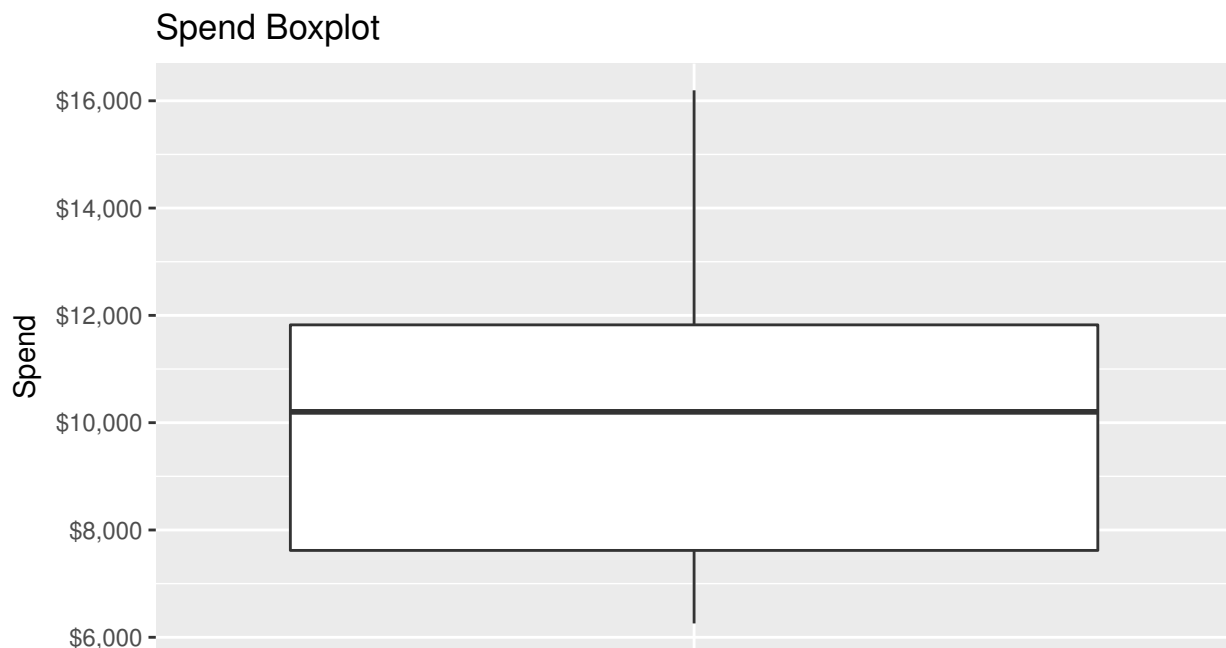
  # add boxplot
  geom_boxplot() +

  # optional stuff to make it cleaner
  # format labels
  scale_y_continuous(labels = scales::dollar) +

  # add title
  ggtitle("Spend Boxplot") +

  # remove x-axis and fix y-axis labels
  theme(axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank()) +
  ylab("Spend")

```



#### 4. Layering on

##### i. `geom_abline()`/`geom_hline()`/`geom_vline()`

Add a line to your graph either by specifying slope and intercept or the location for a vertical/horizontal line. These are the `ggplot2` versions of base R.

geom\_abline() Add a darkgreen abline to the scatterplot.

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
# show days of the week as different colors

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks,
                             col = factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                                           levels = c("Sunday", "Monday", "Tuesday",
                                                       "Wednesday", "Thursday", "Friday",
                                                       "Saturday")))) +

# add points
geom_point() +

# optional stuff to make it cleaner
# format labels
scale_x_continuous(labels = scales::dollar) +
scale_y_continuous(labels = scales::comma) +

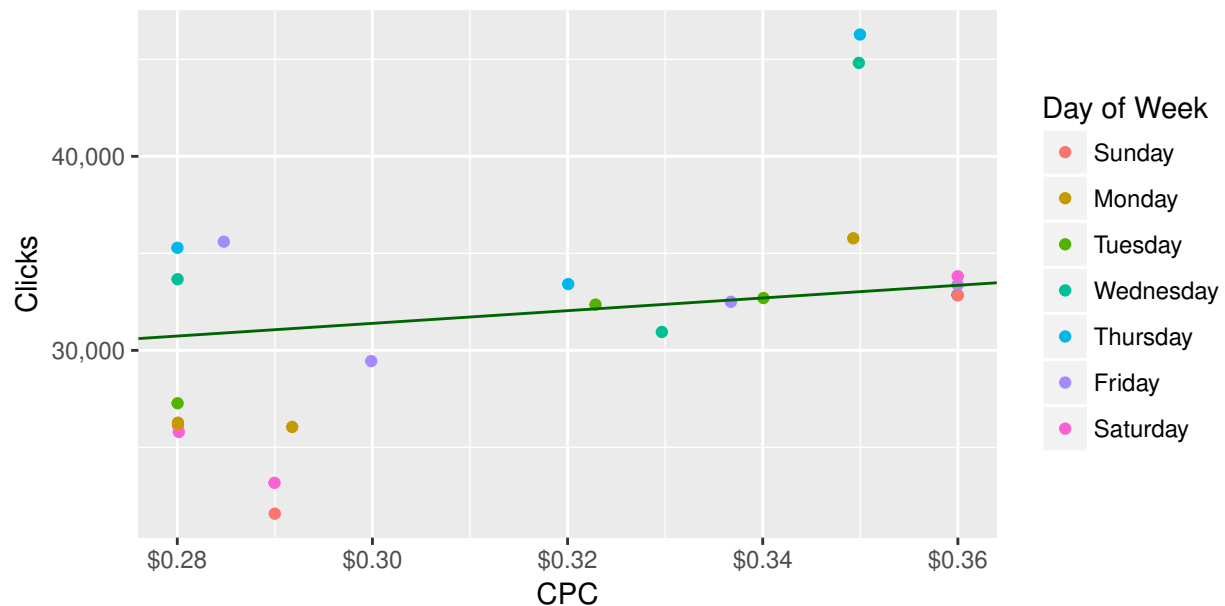
# add title
ggtitle("Scatterplot with abline") +

# fix axis labels
xlab("CPC") +
ylab("Clicks") +

# fix legend title
labs(col = "Day of Week") +

# add line specifying slope and intercept
geom_abline(intercept = min([REDACTED]_pivot$total_clicks),
            slope = median([REDACTED]_pivot$total_clicks), col = "darkgreen")
```

Scatterplot with abline



```

    geom_hline() Add a orange horizontal line at median spend ($10,202.06).
# spend by day for [REDACTED]

# define aesthetics
[REDACTED]_pivot %>%
  ggplot(aes(x = as.Date(day), y = spend)) +

  # add line
  geom_line(col = "blue") +

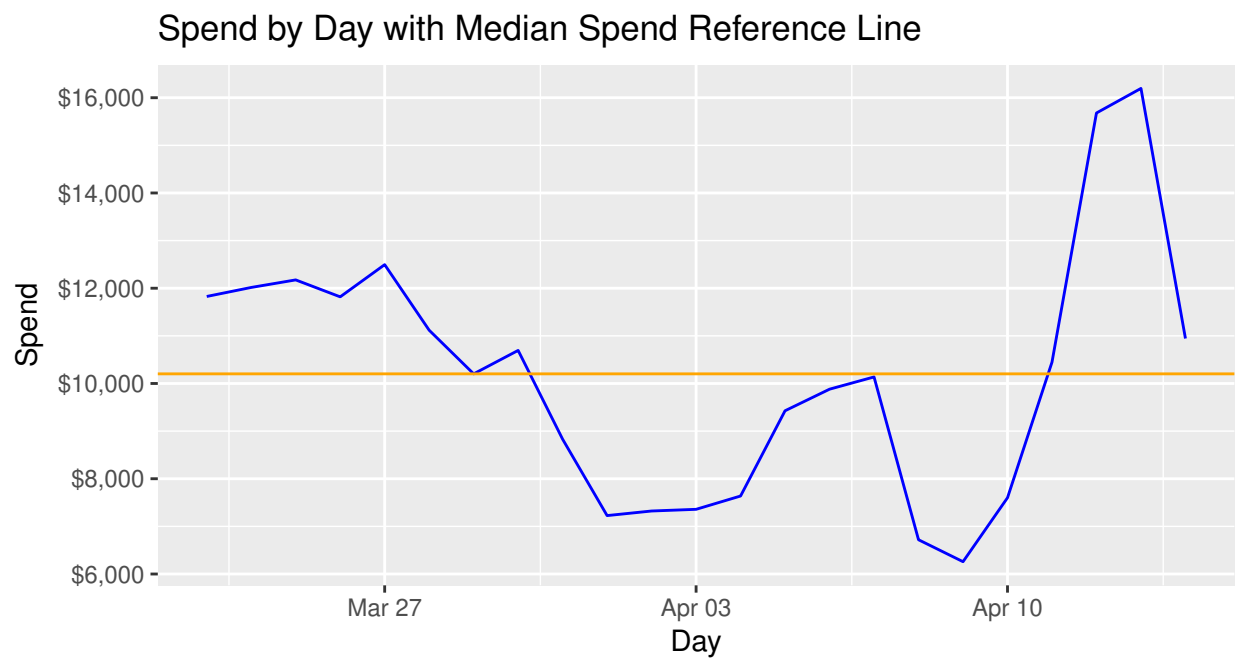
  # optional stuff to make it cleaner
  # format labels
  scale_y_continuous(labels = scales::dollar) +

  # add title
  ggtitle("Spend by Day with Median Spend Reference Line") +

  # fix axis labels
  xlab("Day") +
  ylab("Spend") +

  # add reference line
  geom_hline(yintercept = median([REDACTED]_pivot$spend), col = "orange")

```



`geom_vline()` Add a red vertical line at the median CPC (\$0.32).

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
# show days of the week as different colors

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks,
                              col = factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                                             levels = c("Sunday", "Monday", "Tuesday",
                                                         "Wednesday", "Thursday", "Friday",
                                                         "Saturday")))) +

# add points
geom_point() +

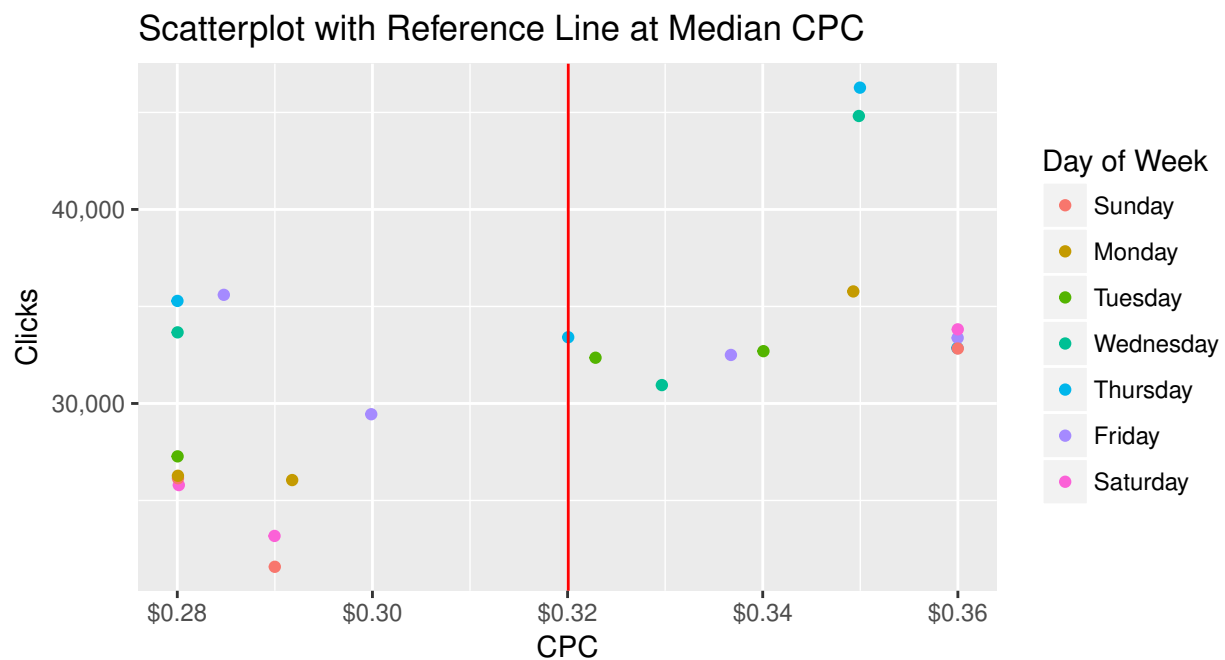
# optional stuff to make it cleaner
# format labels
scale_x_continuous(labels = scales::dollar) +
scale_y_continuous(labels = scales::comma) +

# add title
ggtitle("Scatterplot with Reference Line at Median CPC") +

# fix axis labels
xlab("CPC") +
ylab("Clicks") +

# fix legend title
labs(col = "Day of Week") +

# add reference line
geom_vline(xintercept = median([REDACTED]_pivot$cpc), col = "red")
```



Note that these can also be combined (you can keep adding layers). Let's see what the median CPC and

median clicks look like on the scatterplot.

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
# show days of the week as different colors

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks,
                             col = factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                                           levels = c("Sunday", "Monday", "Tuesday",
                                                       "Wednesday", "Thursday", "Friday",
                                                       "Saturday")))) +

# add points
geom_point() +

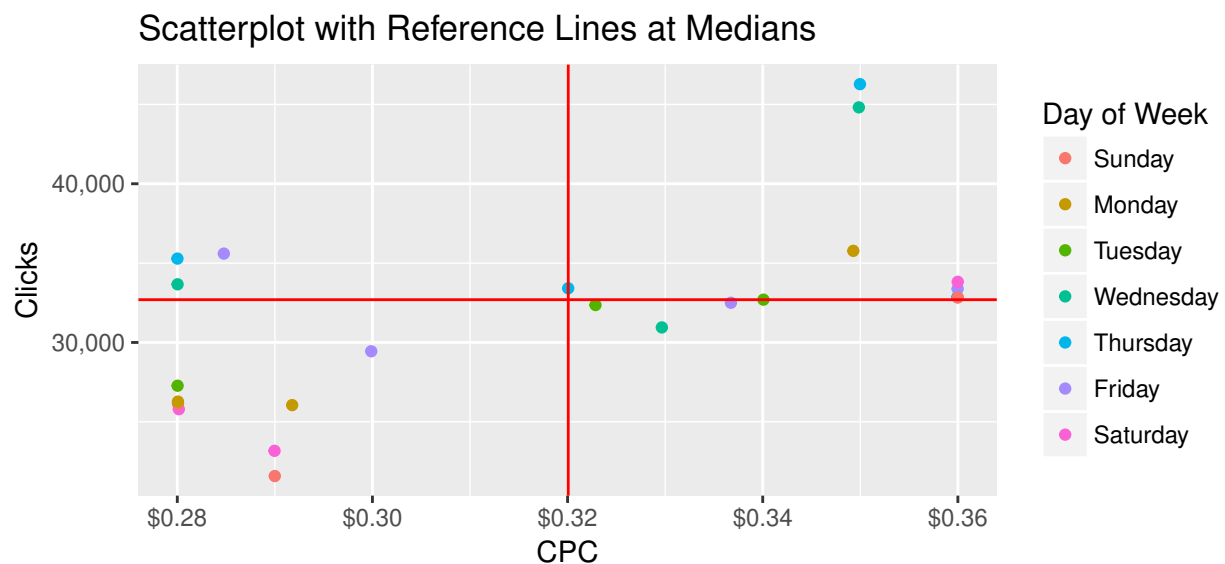
# optional stuff to make it cleaner
# format labels
scale_x_continuous(labels = scales::dollar) +
scale_y_continuous(labels = scales::comma) +

# add title
ggtitle("Scatterplot with Reference Lines at Medians") +

# fix axis labels
xlab("CPC") +
ylab("Clicks") +

# fix legend title
labs(col = "Day of Week") +

# add crosshairs for medians
geom_vline(xintercept = median([REDACTED]_pivot$cpc), col = "red") +
geom_hline(yintercept = median([REDACTED]_pivot$total_clicks), col = "red")
```



ii. `stat_smooth(method = "lm", se = FALSE, alpha = 0.6)`

Add a regression line with transparency 0.6 (`alpha`). Note that when plotting multiple items on a graph, it may be useful to adjust the transparency of the series so that they don't cover each other up. This can be done by specifying the `alpha` parameter with a value less than 1 (1 = opaque, 0 = transparent).

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
```

```
# define aesthetics
```

```
[REDACTED]_pivot %>%
```

```
ggplot(aes(x = cpc, y = total_clicks)) +
```

```
# add points
```

```
geom_point() +
```

```
# optional stuff to make it cleaner
```

```
# format labels
```

```
scale_x_continuous(labels = scales::dollar) +
```

```
scale_y_continuous(labels = scales::comma) +
```

```
# add title
```

```
ggtitle("Scatterplot with Regression Line") +
```

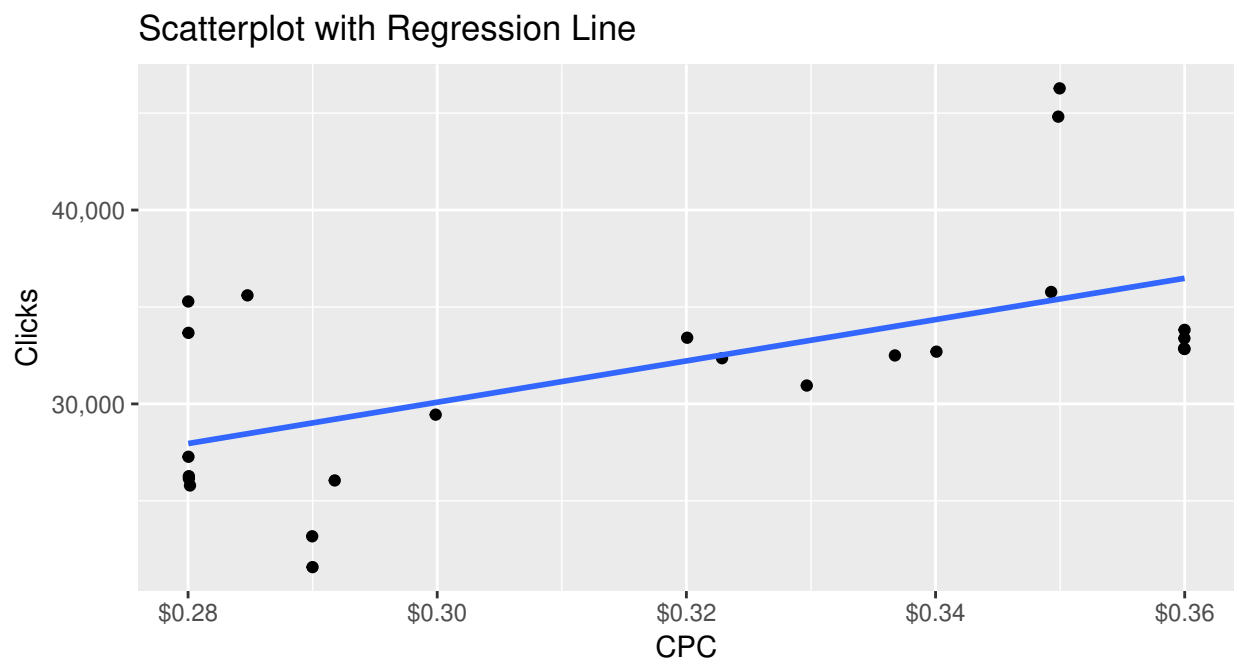
```
# fix axis labels
```

```
xlab("CPC") +
```

```
ylab("Clicks") +
```

```
# add regression line
```

```
stat_smooth(method = "lm", se = FALSE, alpha = 0.6)
```



## 5. Faceting

What if we want to see metrics over categories, like events by site type or tag name? We could put these on the color aesthetic, but things get messy very quickly, and it gets hard to distinguish colors after 5 or 6 of them get added to the graph. Faceting helps us by breaking them out into separate graphs either wrapped or in a grid with all possible combinations, so we can see everything in one plot.

Look at the linear regression of the scatterplot with the day of week added to see the motivation behind this.

```
# scatterplot showing relation of CPC and clicks for [REDACTED]
# show days of the week as different colors

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks,
                             col = factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                             levels = c("Sunday", "Monday", "Tuesday",
                                         "Wednesday", "Thursday", "Friday",
                                         "Saturday")))) +

# add points
geom_point() +

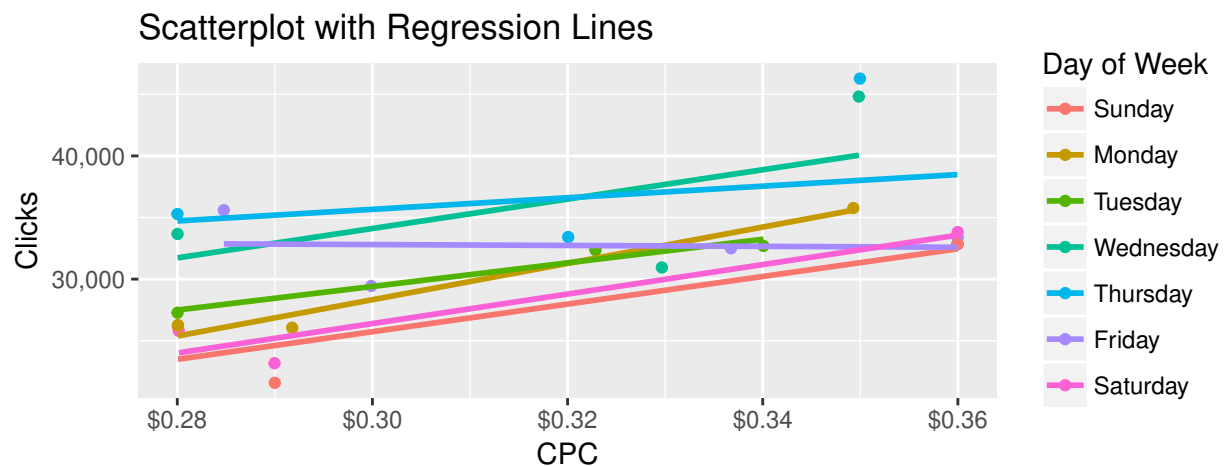
# optional stuff to make it cleaner
# format labels
scale_x_continuous(labels = scales::dollar) +
scale_y_continuous(labels = scales::comma) +

# add title
ggtitle("Scatterplot with Regression Lines") +

# fix axis labels
xlab("CPC") +
ylab("Clicks") +

# fix legend title
labs(col = "Day of Week") +

# add regression lines
stat_smooth(method = "lm", se = FALSE, alpha = 0.6)
```



Wow, that's hard to read! Let's clean this up by faceting.



#### i. facet\_grid()

Facet grids contain all possible combinations of the variables you want to facet on. Note that the “.” here means you aren’t faceting on anything to be printed along the y-axis. The “~” separates the rows facet from the columns facet. Here we only facet by columns with the day of week.

```
# scatterplot showing relation of CPC and clicks for [REDACTED]

# define aesthetics
[REDACTED]_pivot %>%
  ggplot(aes(x = cpc, y = total_clicks)) +

  # add points
  geom_point() +

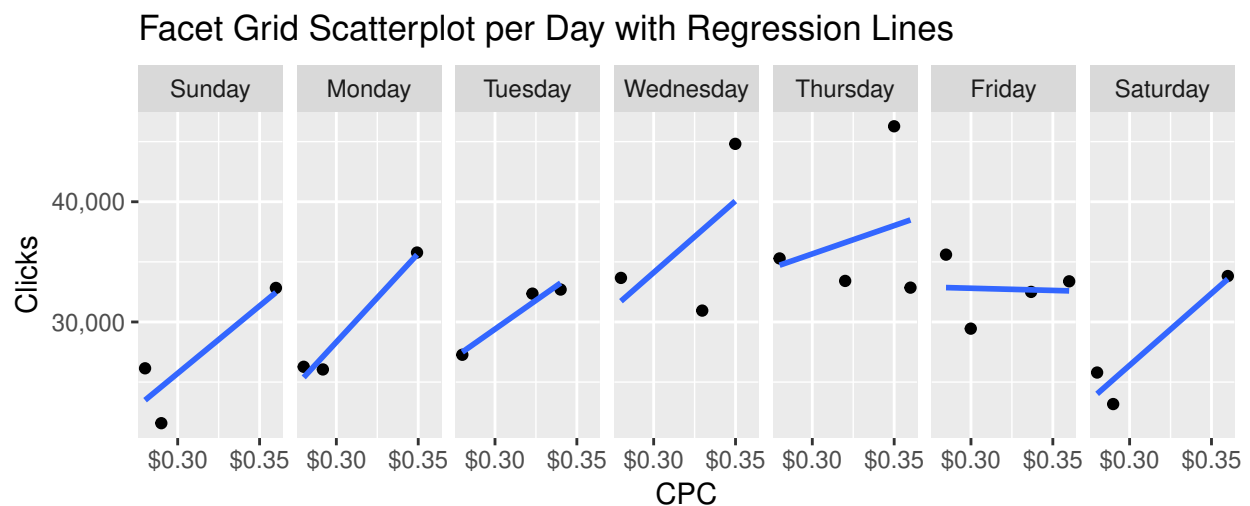
  # format labels
  scale_x_continuous(labels = scales::dollar, breaks = scales::pretty_breaks(n = 2)) +
  scale_y_continuous(labels = scales::comma) +

  # add title
  ggtitle("Facet Grid Scatterplot per Day with Regression Lines") +

  # fix legend title and axis labels
  labs(col = "Day of Week", x = "CPC", y = "Clicks") +

  # add regression lines
  stat_smooth(method = "lm", se = FALSE, alpha = 0.6) +

  # add facet grid
  facet_grid(. ~ factor(format(as.Date([REDACTED]_pivot$day), "%A"),
    levels = c("Sunday", "Monday", "Tuesday", "Wednesday",
      "Thursday", "Friday", "Saturday")))
```



## ii. facet\_wrap()

Facet wrap only contains all possible combinations for which there is data to fill the intersection of a given row and column whereas facet grids will have a blank section if there is no data. This is why facet grids will not have whitespace and facet wraps will. Facet wraps, just as the name implies, create a series of plots from left to right and wrap to the next rows until they have plotted all combinations with data.

Note that here we don't need to add "." if we aren't using that faceting element as we did with `facet_grid()`

```
# scatterplot showing relation of CPC and clicks for [REDACTED]

# define aesthetics
[REDACTED]_pivot %>% ggplot(aes(x = cpc, y = total_clicks)) +
  # add points
  geom_point() +

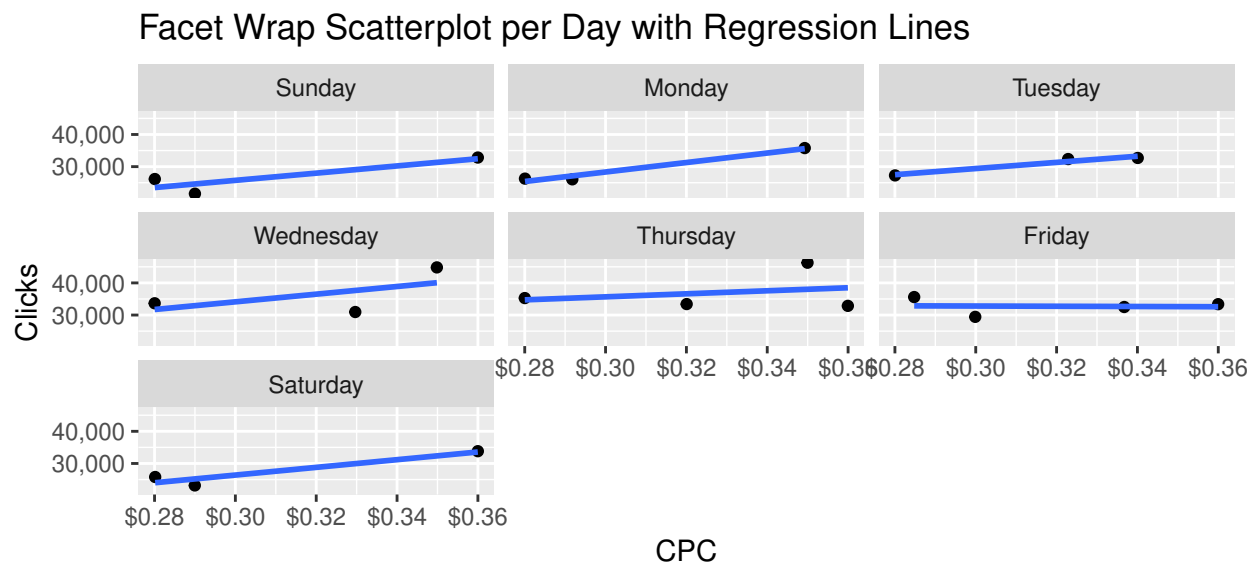
  # format labels
  scale_x_continuous(labels = scales::dollar) +
  scale_y_continuous(labels = scales::comma) +

  # add title
  ggtitle("Facet Wrap Scatterplot per Day with Regression Lines") +

  # fix legend title and axis labels
  labs(col = "Day of Week", x = "CPC", y = "Clicks") +

  # add regression lines
  stat_smooth(method = "lm", se = FALSE, alpha = 0.6) +

  # add facet wrap
  facet_wrap(~ factor(format(as.Date([REDACTED]_pivot$day), "%A"),
                        levels = c("Sunday", "Monday", "Tuesday",
                                   "Wednesday", "Thursday", "Friday",
                                   "Saturday")))
```



Notice how we have empty space the size of 2 plots next to Saturday in this graph, but we didn't have that with `facet_grid()`. Which you use depends on your data and what you are looking to show.