101_WK4_Data_Vis_Basic

Seung Hyun Sung

11/8/2021

ANATOMY OF A GGPLOT2 OBJECT —-

1A Framing ggplot

Think of ggplots like building layers of a cake. Each layer is added on top.

[1] Create a canvas defined by mapping to columns in your data

[2] Add 1 or more geometrics (geoms)

[3] Add formatting features. {Scales, Themes, Facets, etc}

Geom/Geometries: These define how your data looks on your plot

- Stands for geometrics
- Geometrics are the funcdermental way to represent data in your plot
- Determines Plot Type:
- Histograms
- Scatter Plots
- Box Plots
- Bar/Column Plots
- More and more

Formatting: These add customization to your plot to control wide-range of appearence options. Scales, Faceting, Position Adjustments, Labels, Legends, Themes are commonly customerized.

> *Enables Customization on Steroids ggplot2 is super flexible giving tons of options. The downside in this flexibility is that it takes a while to learn. Matt Dancho —-*

> *For business reports the it is important to get the themes right and reported with same formated theme. ___ Matt Dancho*

> *The key to a good ggplot is knowing how to format the data for a ggplot.*

1B How ggplot works —-

Step 1: Format data —-

unlike base graphics, ggplot works with data.frames and not individual vectors.

```
revenue_by_year_tbl <- bike_orderlines_tbl %>%
    select(order_date, total_price) %>%
    mutate(year = year(order_date)) %>%

    group_by(year) %>%
    summarize(revenue = sum(total_price)) %>%
    ungroup()


revenue_by_year_tbl
```

```
## # A tibble: 5 x 2
##    year  revenue
##   <dbl>    <dbl>
## 1  2011 11292885
## 2  2012 12163075
## 3  2013 16480775
## 4  2014 13924085
## 5  2015 17171510
```

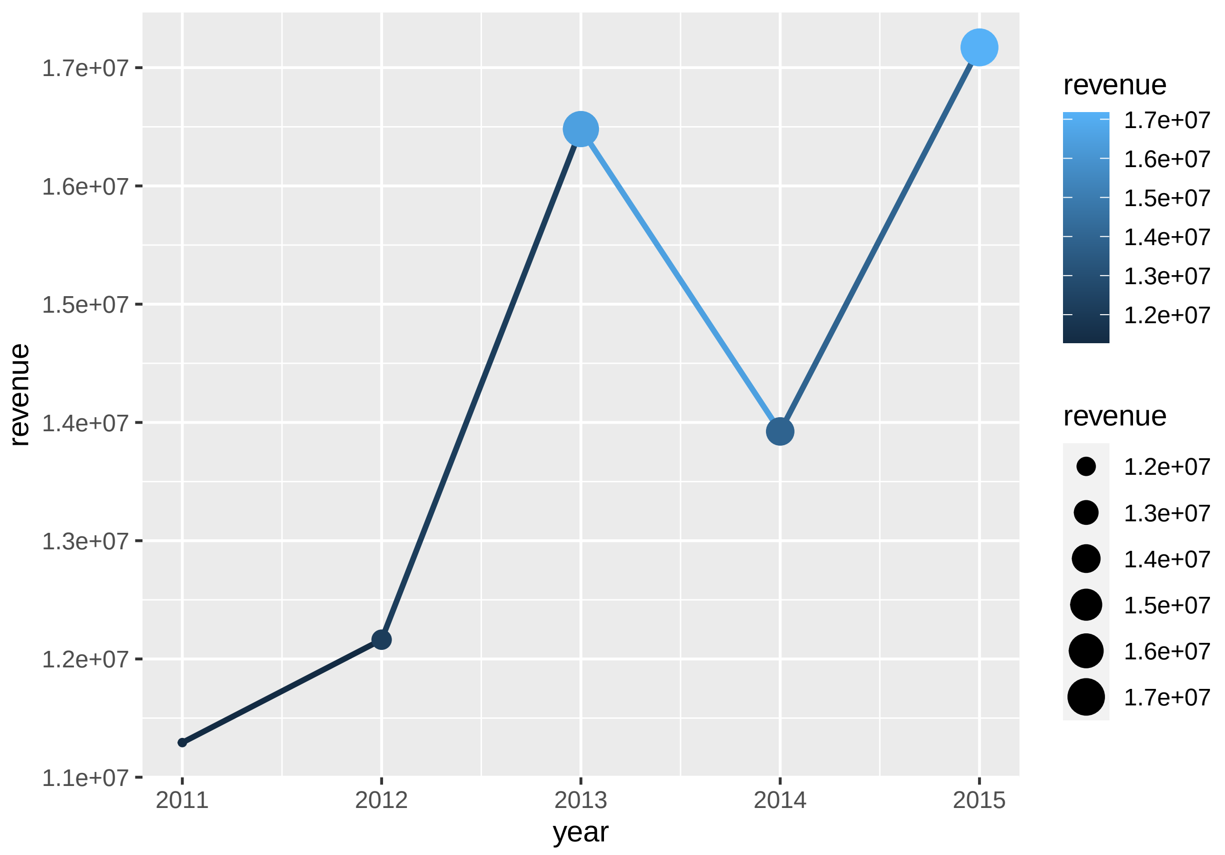
Step 2: Plot —-

Mapping: Connects data columns to ggplot aesthetics

GGplot structure

- Step 1: Build Canvas: Involves mapping **columns** in data to ggplot() **aesthetics** (x, y, color, fill, size etc) via aes() function
- Step 2: Geometries: 2nd Layer that generates a visual depiction of the data using a geometry type (e.g. Line Plot)

```
revenue_by_year_tbl %>%
    # Canvas
    ggplot(aes(x = year, y = revenue, color = revenue)) +

    # Geometries
    geom_line(size = 1) +
    # aesthetics specifically targeting certain geometries
    geom_point(aes(size = revenue))
```



- Scale Color: Enables customizing the color aesthetic (mapped to revenue in this case)
- Scale X & Y: Enables customizing the x-axis and y-axis (mapped to year and revenue in this case)
- Labels: Changes the **text** for title, subtitle, x, y, legends & captions
- Themes: Usually we start with a base theme e.g. theme_bw() and then modify with the theme() function

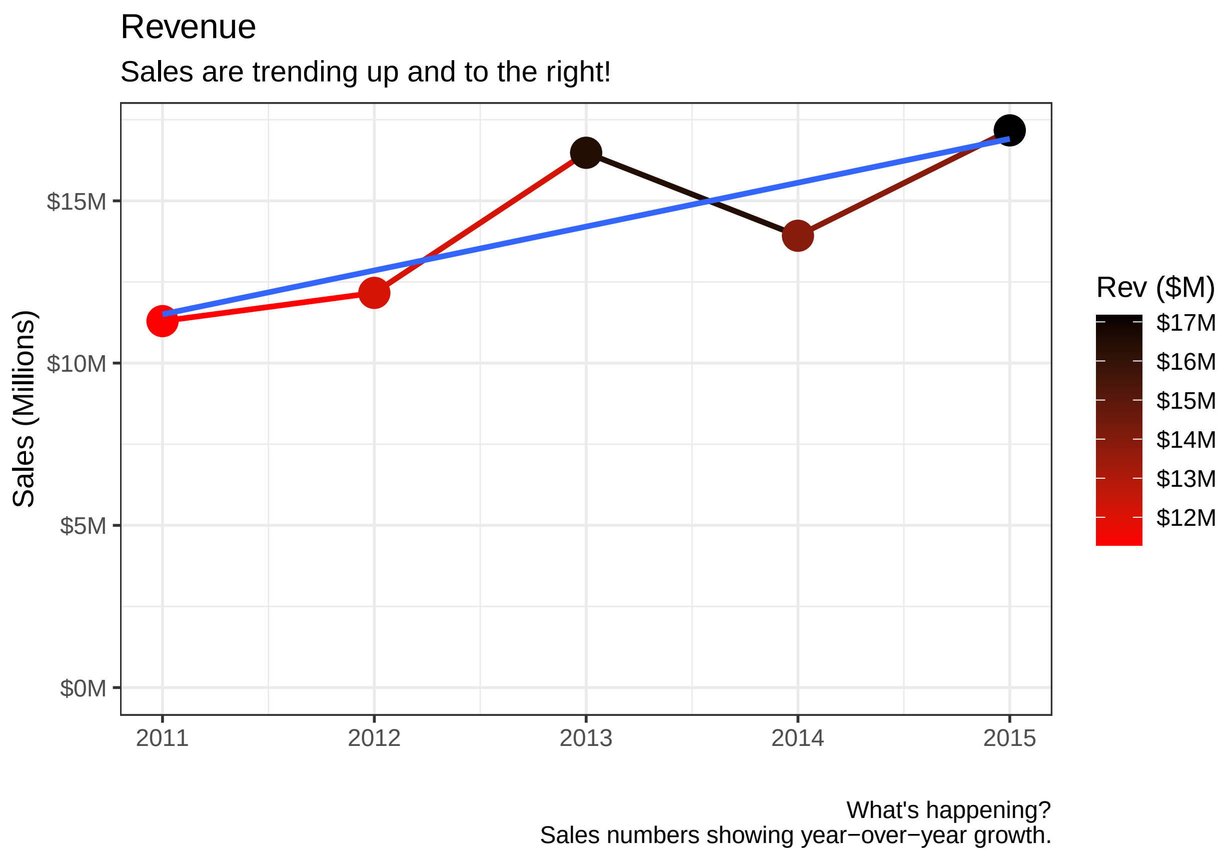
```
g <- revenue_by_year_tbl %>%
    # Canvas
    ggplot(aes(x = year, y = revenue, color = revenue)) +

    # Geometries
    geom_line(size = 1) +
    geom_point(size = 5) +
    geom_smooth(method = "lm", se = FALSE) +

    # Formatting
    expand_limits(y = 0) +
    scale_color_continuous(low = "red", high = "black",
                           labels = scales::dollar_format(scale = 1/1e6, suffix = "M")) +
    scale_y_continuous(labels = scales::dollar_format(scale = 1/1e6, suffix = "M")) +
    labs(
        title = "Revenue",
        subtitle = "Sales are trending up and to the right!",
        x = "",
        y = "Sales (Millions)",
        color = "Rev ($M)",
        caption = "What's happening?\nSales numbers showing year-over-year growth."
    ) +
    theme_bw() +
    theme(legend.position = "right", legend.direction = "vertical")
```

1C What is a ggplot? —-

Key Concept: The ggplot object is just a list that captures layers, scales, mappings, theme, coordinates, and labels that you customize

```
g
```

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



Types of Graphs: ggplot2 Geometries

```
bike_orderlines_tbl <- read_rds("~/Desktop/University_business_science/DS4B_101/00_data//bike_sales/dat

glimpse(bike_orderlines_tbl)
```

```
## Rows: 15,644
## Columns: 13
## $ order_date     <dttm> 2011-01-07, 2011-01-07, 2011-01-10, 2011-01-10, 2011-0~
## $ order_id       <dbl> 1, 1, 2, 2, 3, 3, 3, 3, 3, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7~
## $ order_line     <dbl> 1, 2, 1, 2, 1, 2, 3, 4, 5, 1, 1, 2, 3, 4, 1, 2, 3, 4, 1~
## $ quantity       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1~
## $ price          <dbl> 6070, 5970, 2770, 5970, 10660, 3200, 12790, 5330, 1570,~
## $ total_price    <dbl> 6070, 5970, 2770, 5970, 10660, 3200, 12790, 5330, 1570,~
## $ model          <chr> "Jekyll Carbon 2", "Trigger Carbon 2", "Beast of the Ea~
## $ category_1     <chr> "Mountain", "Mountain", "Mountain", "Mountain", "Road",~
## $ category_2     <chr> "Over Mountain", "Over Mountain", "Trail", "Over Mounta~
## $ frame_material <chr> "Carbon", "Carbon", "Aluminum", "Carbon", "Carbon", "Ca~
## $ bikeshop_name  <chr> "Ithaca Mountain Climbers", "Ithaca Mountain Climbers",~
## $ city           <chr> "Ithaca", "Ithaca", "Kansas City", "Kansas City", "Loui~
## $ state          <chr> "NY", "NY", "KS", "KS", "KY", "KY", "KY", "KY", "KY", "~
```

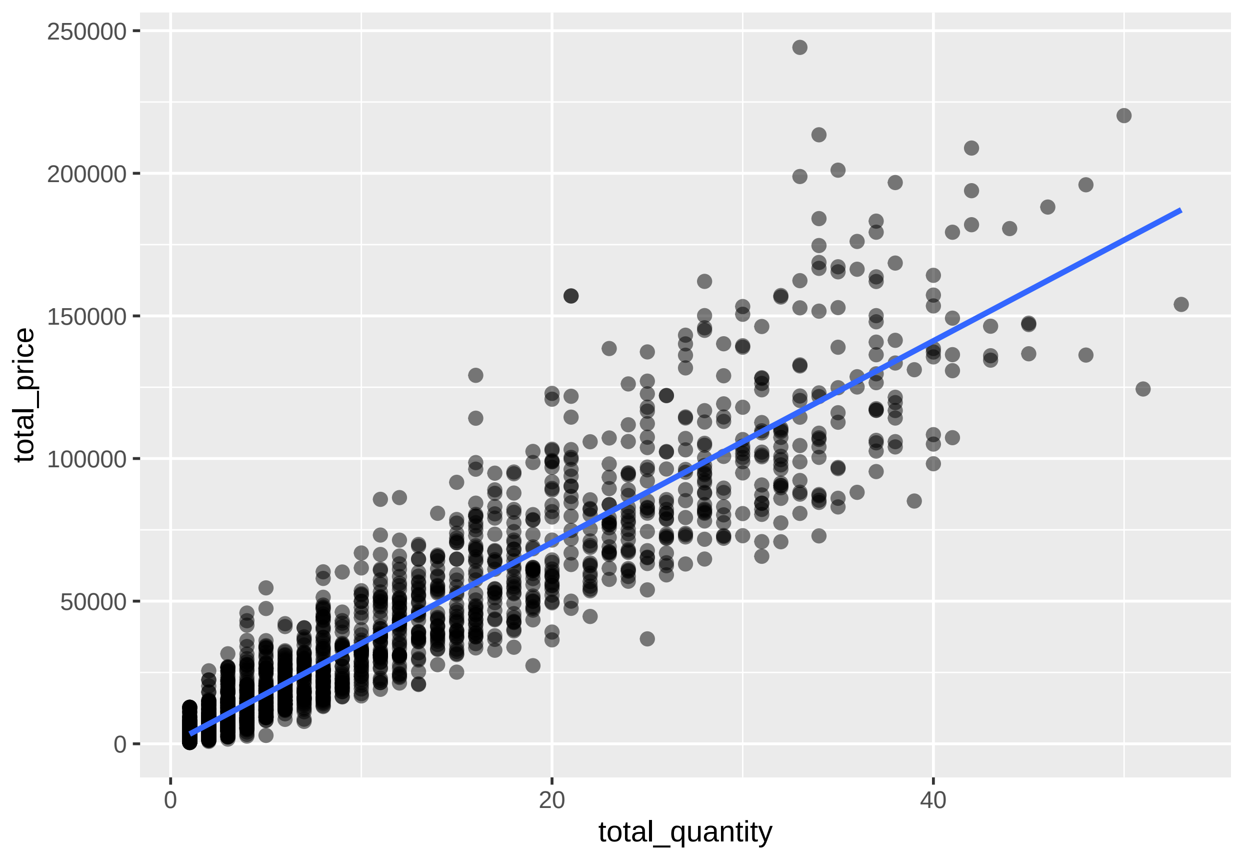
2A Point / Scatter Plots (geom_point)

- Great for Continuous vs Continuous
- Also good for Lollipop Charts (more on this in advanced plots)

Goal: Explain relationship between order value and quantity of bikes sold

```r
# Data Manipulation
order_value_tbl <- bike_orderlines_tbl %>%
    select(order_id, order_line, total_price, quantity) %>%
    group_by(order_id) %>%
    summarise(
        total_quantity = sum(quantity),
        total_price    = sum(total_price)
    ) %>%
    ungroup()


# Scatter Plot
order_value_tbl %>%
    ggplot(aes(x = total_quantity, y = total_price)) +
    # geometries
    geom_point(alpha = 0.5, size = 2) +
    # uses spine (default) y ~ s(x, bs = "cs")
    # change method to 'lm'
    geom_smooth(method = 'lm', se = FALSE)
```

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

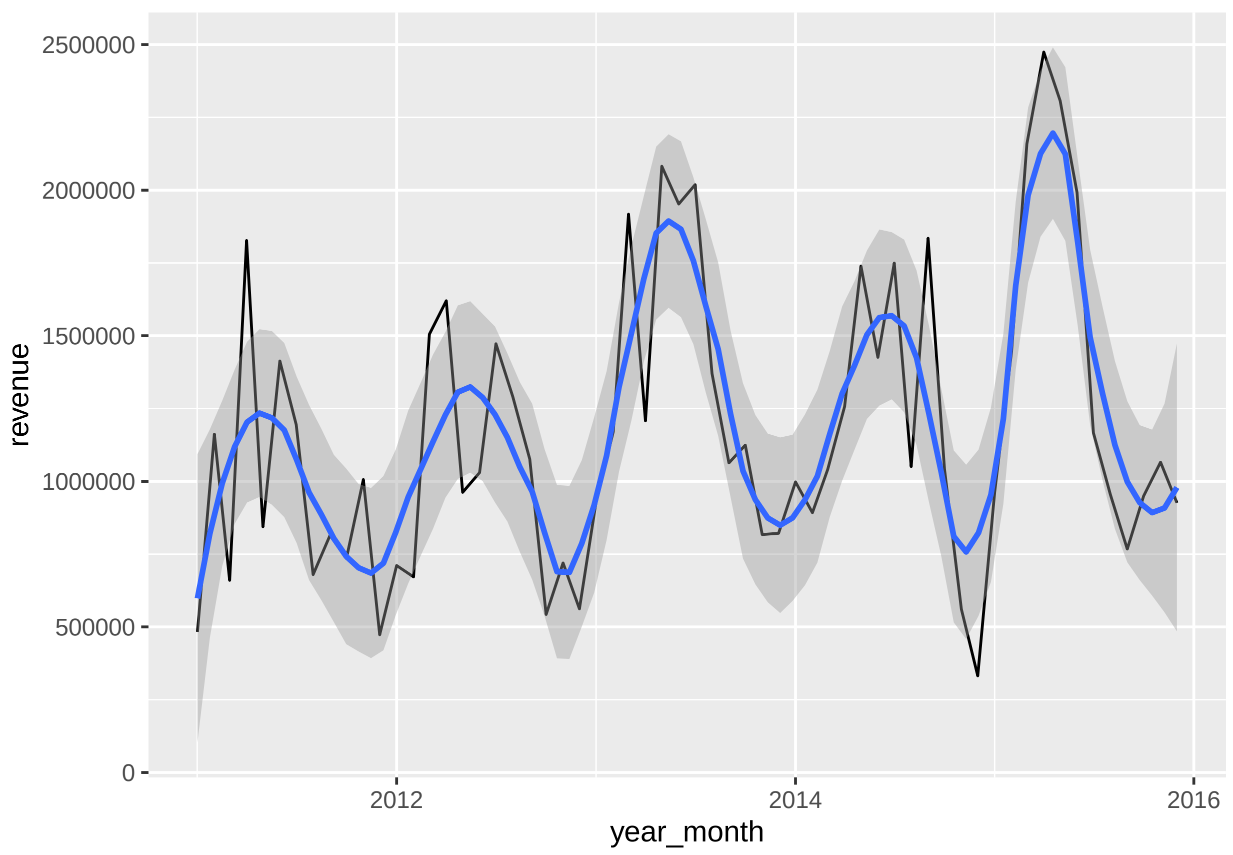


2B Line Plots —-

- Great for time series
- Goal: Describe revenue by Month, expose cyclic nature

```
# Data Manipulation
revenue_by_month_tbl <- bike_orderlines_tbl %>%
    select(order_date, total_price) %>%
    mutate(year_month = floor_date(order_date, "months") %>% ymd()) %>%
    group_by(year_month) %>%
    summarise(revenue = sum(total_price)) %>%
    ungroup()



# Line Plot
revenue_by_month_tbl %>%
    ggplot(aes(x = year_month, y = revenue)) +
    geom_line(size = 0.5, linetype = 1) +
    geom_smooth(method = 'loess', span = 0.2)
```

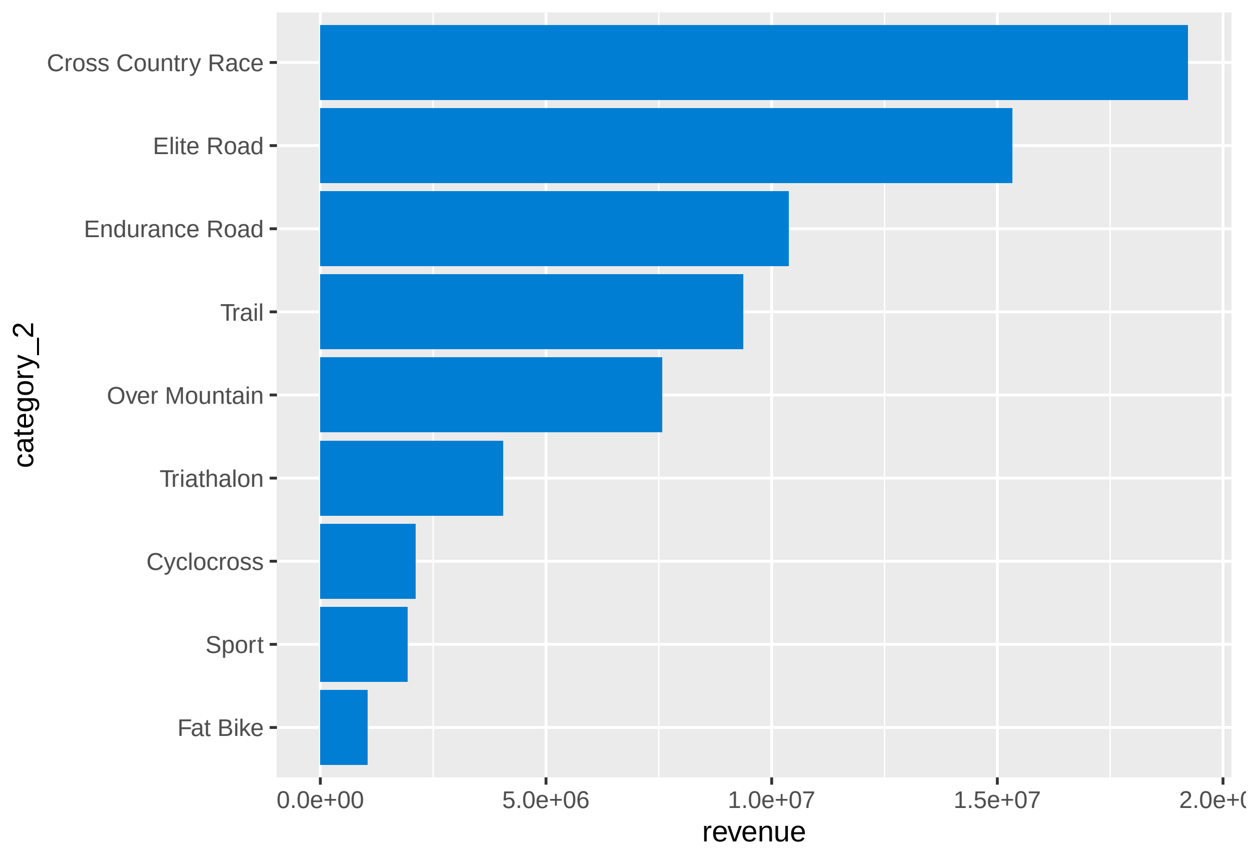
```
## `geom_smooth()` using formula 'y ~ x'
```



2C Bar / Column Plots —-

- – Great for categories
- Goal: Sales by Descriptive Category

```
# Data Manipulation
revenue_by_category_2 <- bike_orderlines_tbl %>%
    select(category_2, total_price) %>%
    group_by(category_2) %>%
    summarise(revenue = sum(total_price)) %>%
    ungroup()


# Bar Plot
revenue_by_category_2 %>%
    mutate(category_2 = category_2 %>% fct_reorder(revenue)) %>%
    ggplot(aes(x = category_2, y = revenue)) +
    geom_col(fill = palette_dark()[6]) +
    coord_flip()
```



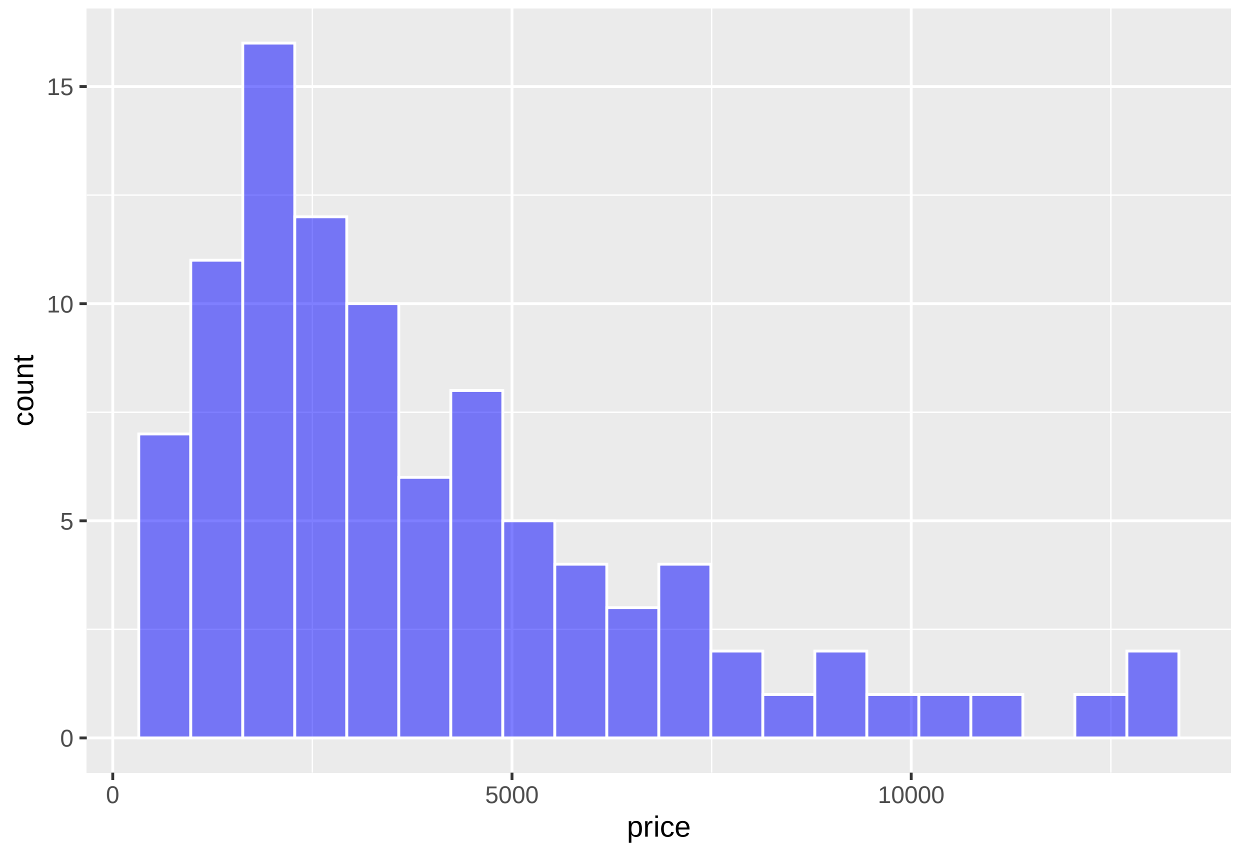
R - difference between geom_bar() and geom_col() in ggplot2

- There are two types of bar charts: geom_bar() and geom_col().
- geom_bar() makes the height of the bar proportional to the number of cases in each group (or if the weight aesthetic is supplied, the sum of the weights). If you want the heights of the bars to represent values in the data, use geom_col() instead geom_bar() uses stat_count() by default: it counts the number of cases at each x position
- summary:
  - geom_bar : represents stats (number of cases in each group)
  - geom_bar : represents value

2D Histogram / Density Plots —-

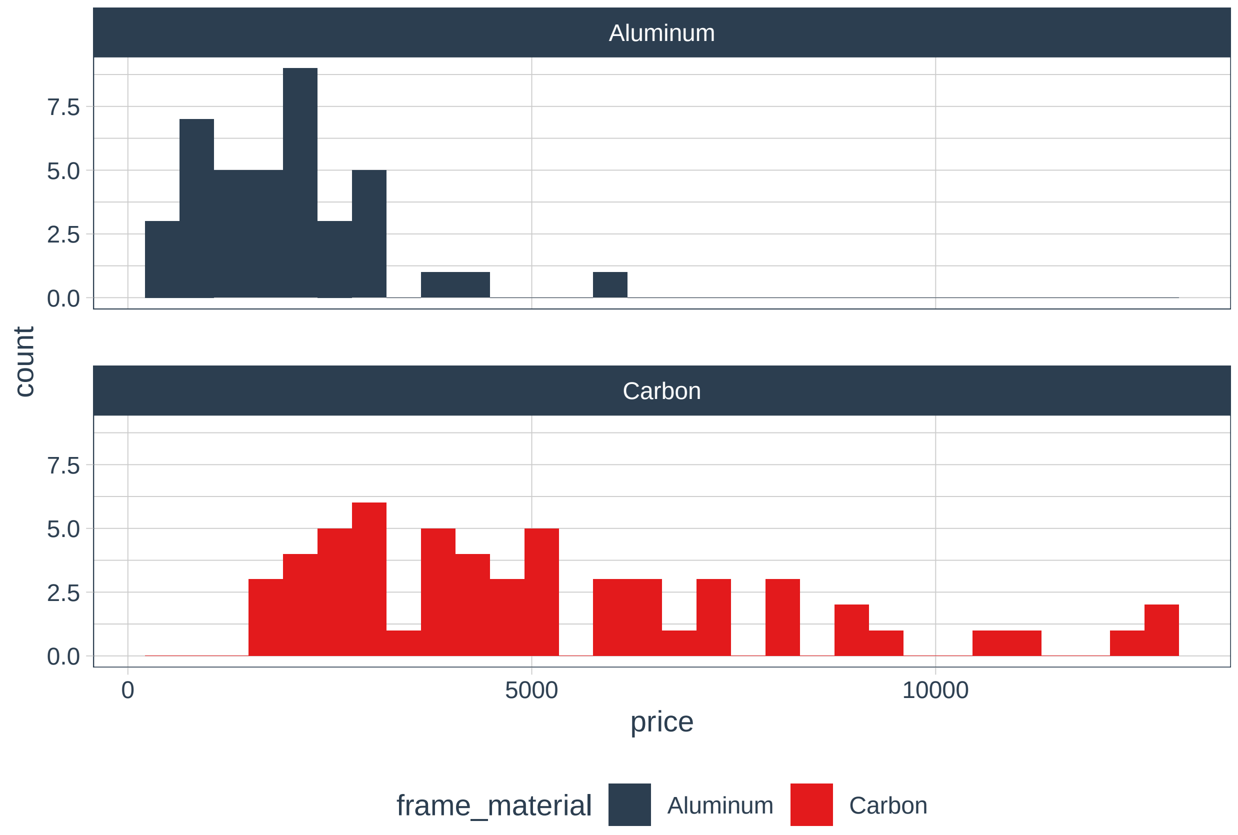
- Great for inspecting the distribution of a variable
- Goal: Unit price of bicycles

```
# Histogram
bike_orderlines_tbl %>%
    distinct(model, price) %>%
    ggplot(aes(price)) +
    geom_histogram(bins = 20, color = "white", fill = "blue", alpha = 0.5)
```

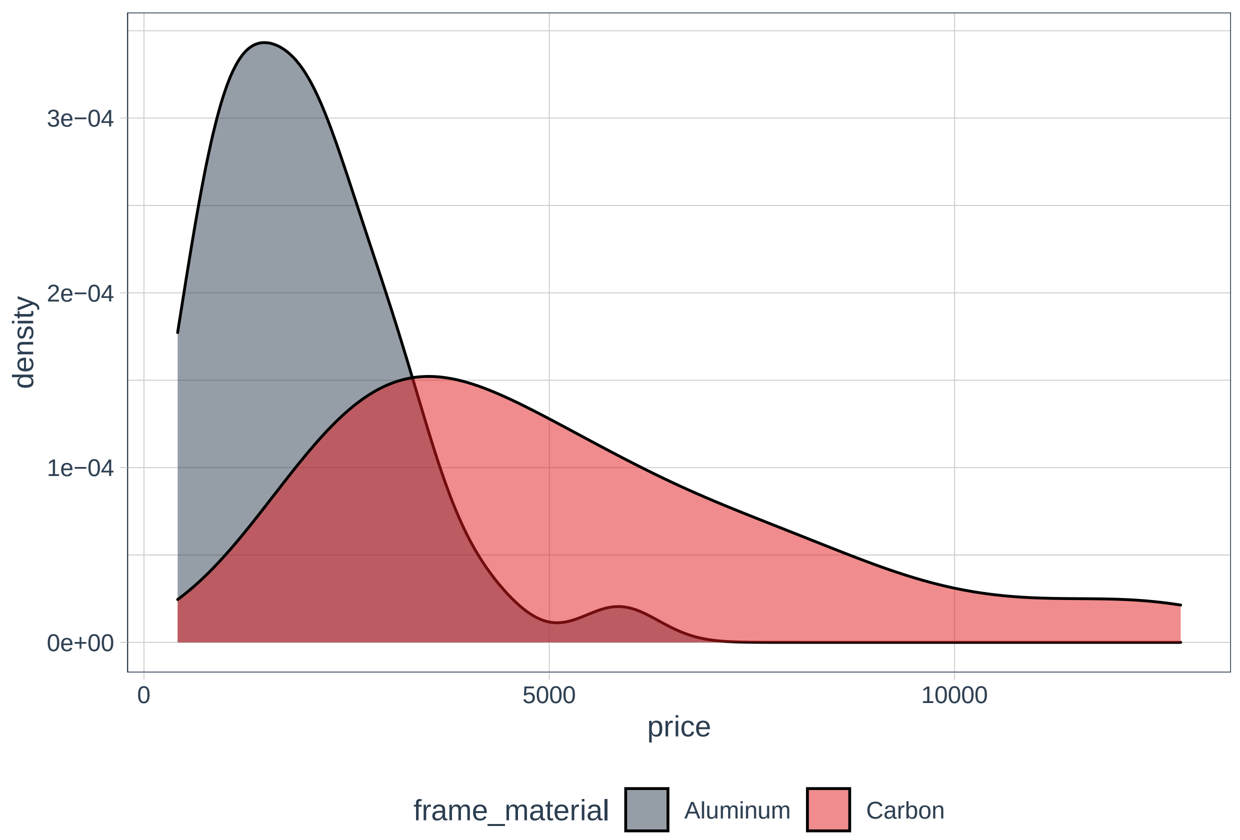


```
# Goal: Unit price of bicylce, segmenting by frame material

# Histogram
bike_orderlines_tbl %>%
    distinct(model, price, frame_material) %>%
    ggplot(aes(price, fill = frame_material)) +
    geom_histogram() +
    facet_wrap(~frame_material, ncol = 1) +
    scale_fill_tq() +
    theme_tq()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



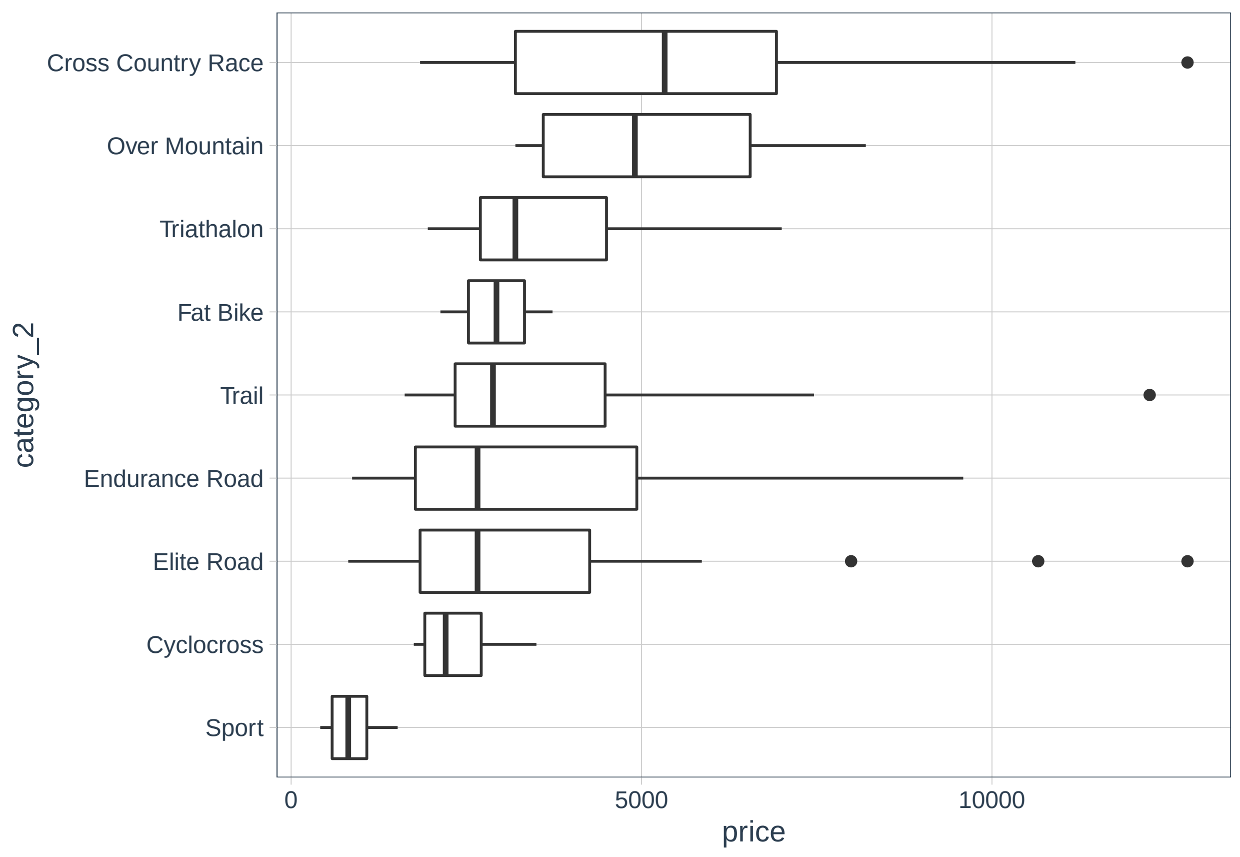
```
# Density
bike_orderlines_tbl %>%
    distinct(model, price, frame_material) %>%
    ggplot(aes(price, fill = frame_material)) +
    geom_density(alpha = 0.5) +
    scale_fill_tq() +
    theme_tq()
```



2E Box Plot / Violin Plot —-

- Great for comparing distributions
- Goal: Unit price of models, segmenting by category 2

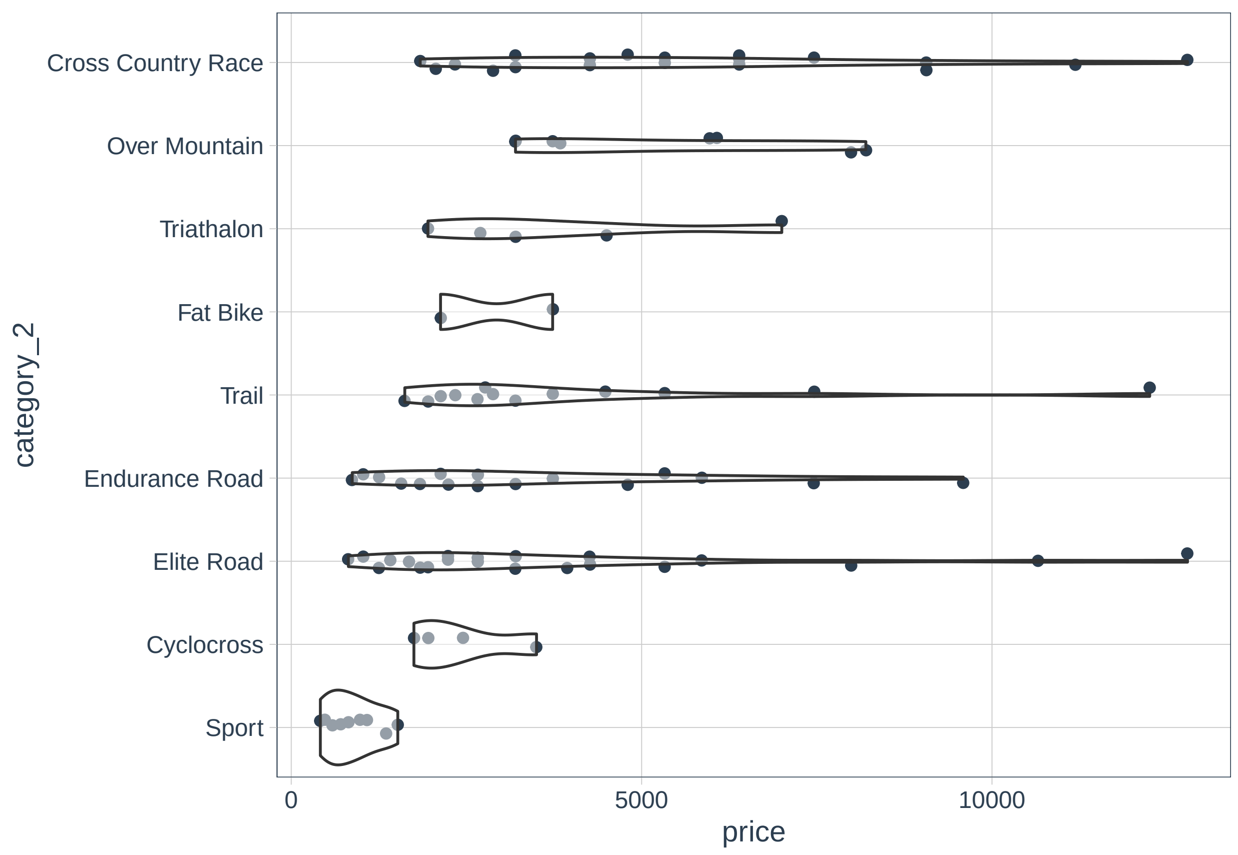
```
# Data Manipulation
unit_price_by_cat_2_tbl <- bike_orderlines_tbl %>%
    select(category_2, model, price) %>%
    distinct() %>%
    mutate(category_2 = as_factor(category_2) %>% fct_reorder(price))

# Box Plot
unit_price_by_cat_2_tbl %>%
    ggplot(aes(category_2, price)) +
    geom_boxplot() +
    coord_flip() +
    theme_tq()
```



2F Violin Plot & Jitter Plot

- It adds a small amount of random variation to the location of each point, and is a useful way of handling overplotting caused by discreteness in smaller datasets.

```
unit_price_by_cat_2_tbl %>%
    ggplot(aes(category_2, price)) +
    geom_jitter(width = 0.1, color = "#2c3e50") +
    geom_violin(alpha = 0.5) +
    coord_flip() +
    theme_tq()
```

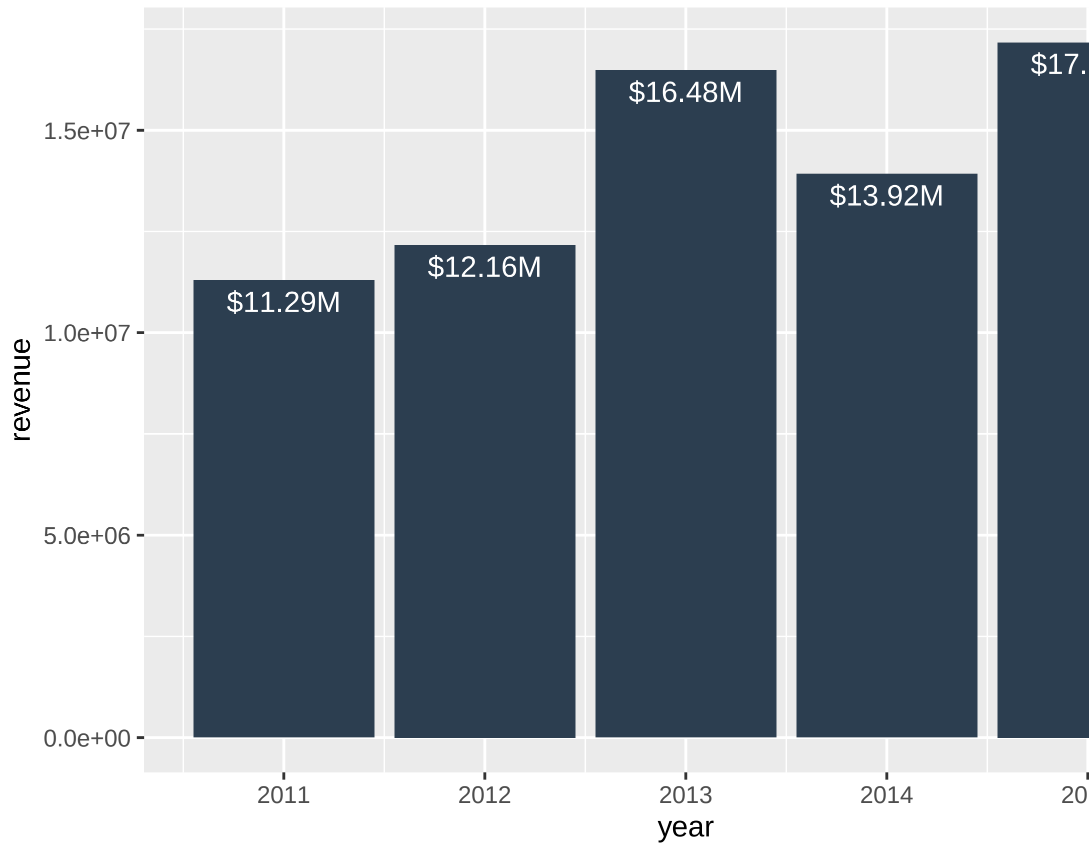


2G Adding Text & Labels —-

- Goal: Exposing sales over time, highlighting outlier

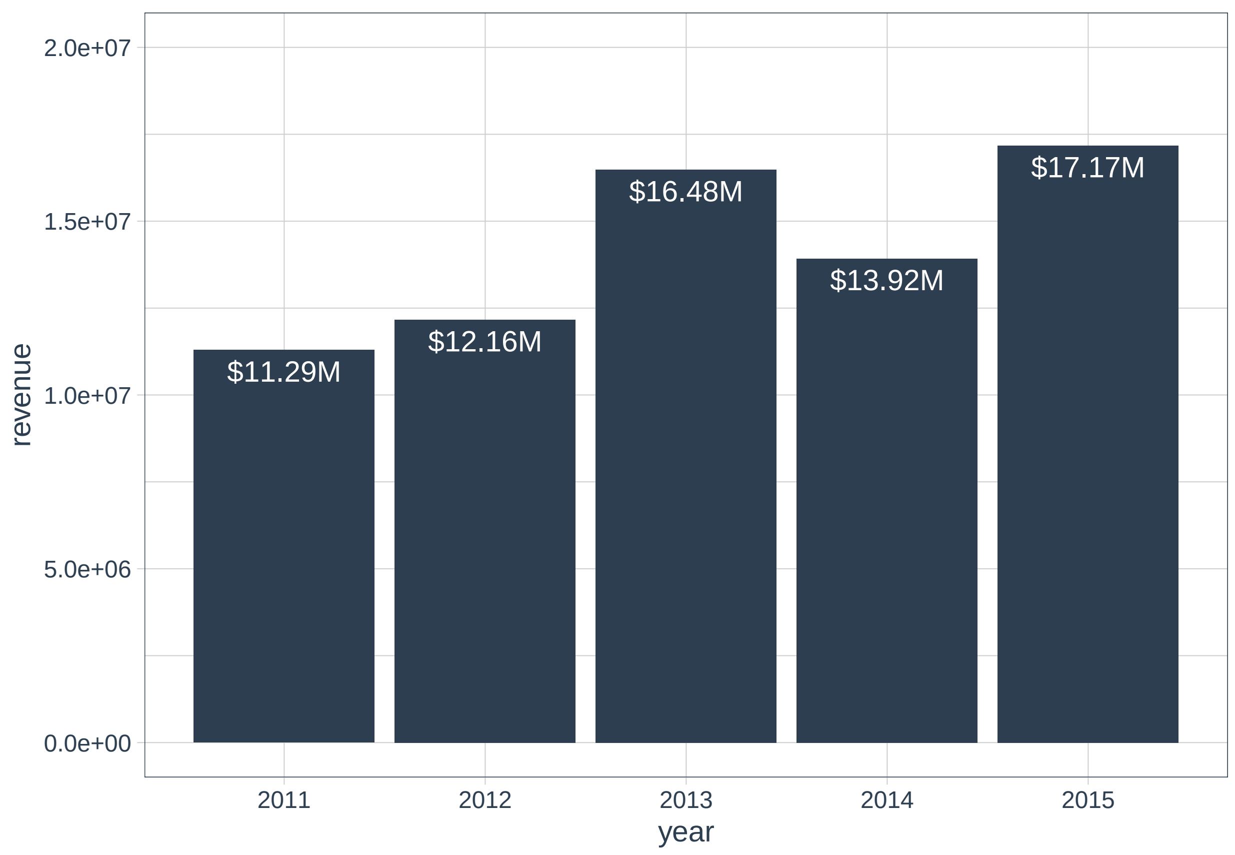
```
# Data Manipulation
revenue_by_year_tbl <- bike_orderlines_tbl %>%
    select(order_date, total_price) %>%
    mutate(year = year(order_date)) %>%
    group_by(year) %>%
    summarise(revenue = sum(total_price)) %>%
    ungroup()
```

```
revenue_by_year_tbl %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill="#2c3e50") +
    geom_text(aes(label = scales::dollar(revenue, scale = 1e-6, suffix = "M")), vjust = 1.5, color = "wh
```

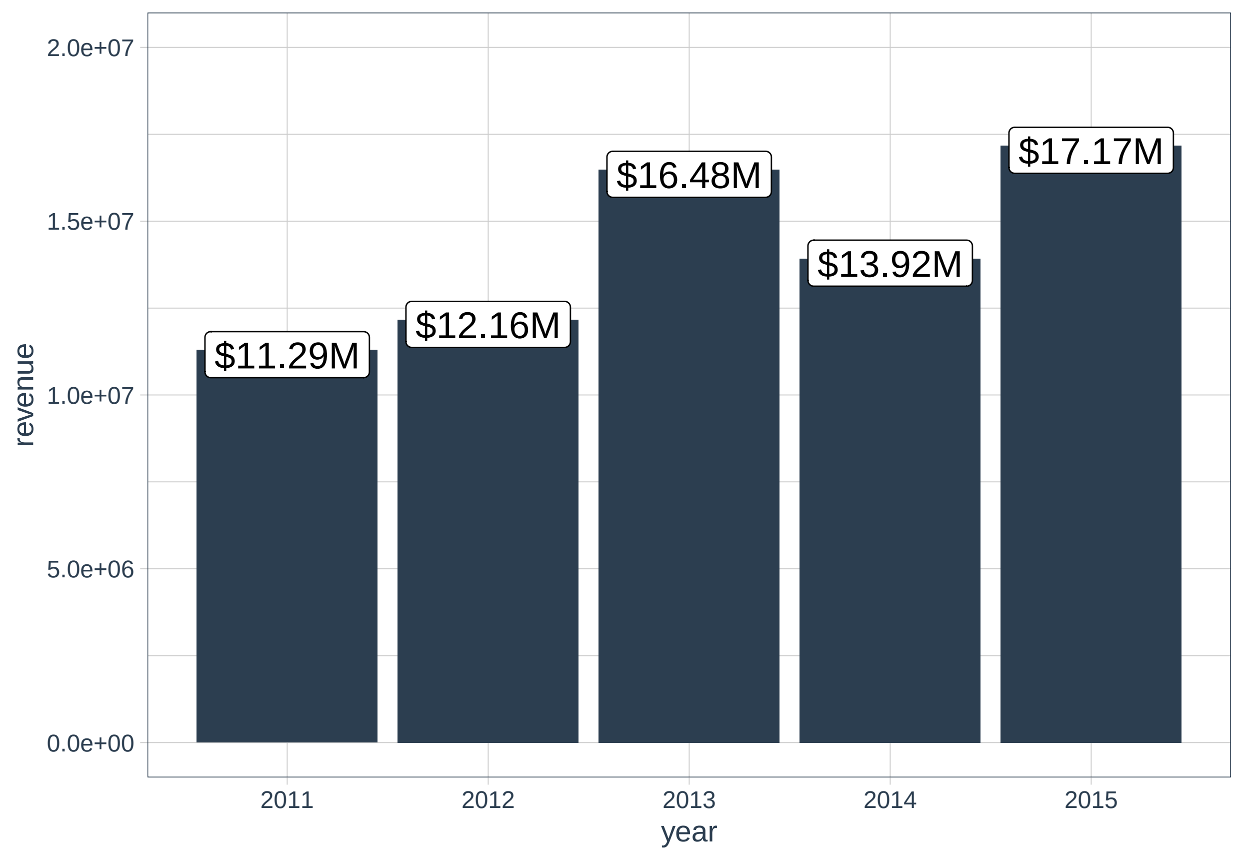


Adding text to bar chart

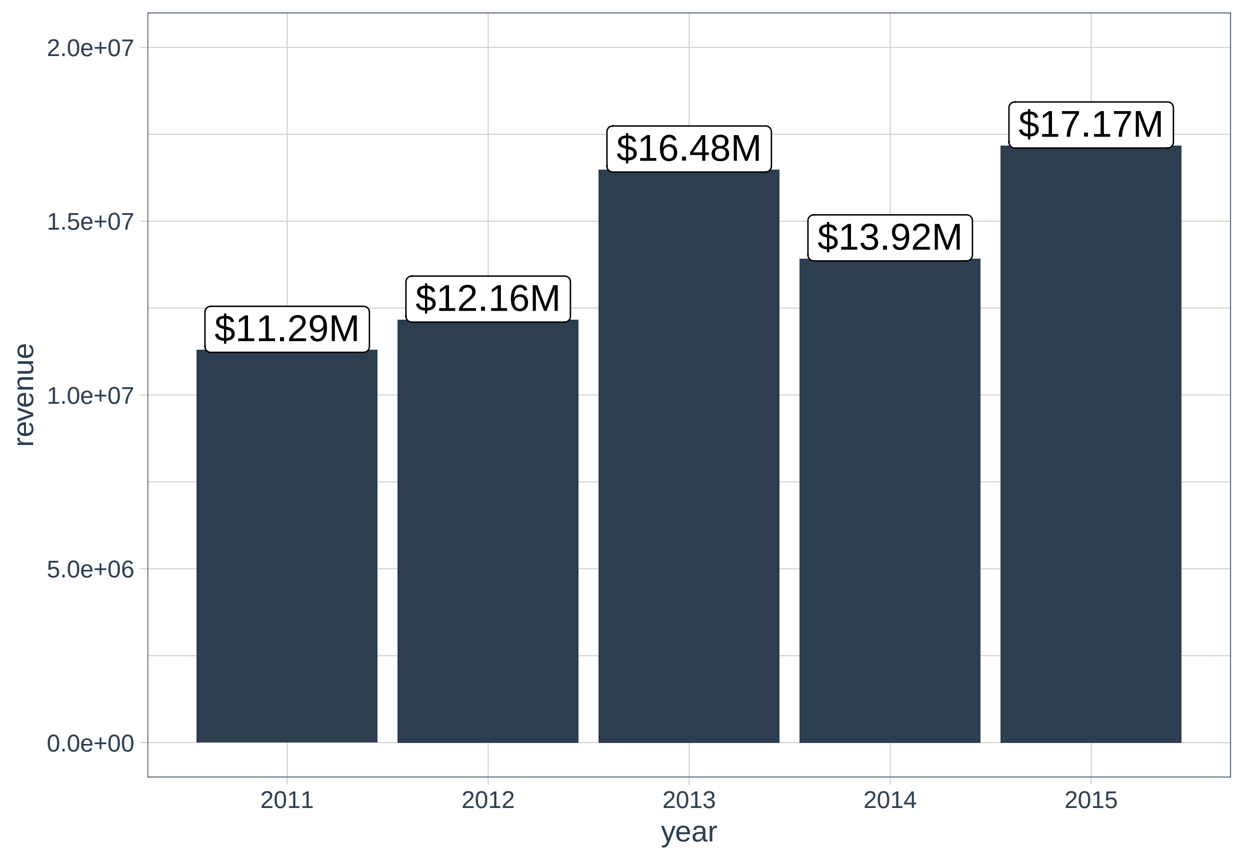
```
revenue_by_year_tbl %>%
    mutate(revenue_text = scales::dollar(revenue, scale = 1e-6, suffix = "M")) %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill="#2c3e50") +
    geom_text(aes(label = revenue_text), vjust = 1.5, color = "white") +
    expand_limits(y = 2e7) +
    theme_tq()
```



```
revenue_by_year_tbl %>%
    mutate(revenue_text = scales::dollar(revenue, scale = 1e-6, suffix = "M")) %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill="#2c3e50") +
    geom_label(aes(label = revenue_text), vjust = 0.6, size = 5) +
    expand_limits(y = 2e7) +
    theme_tq()
```



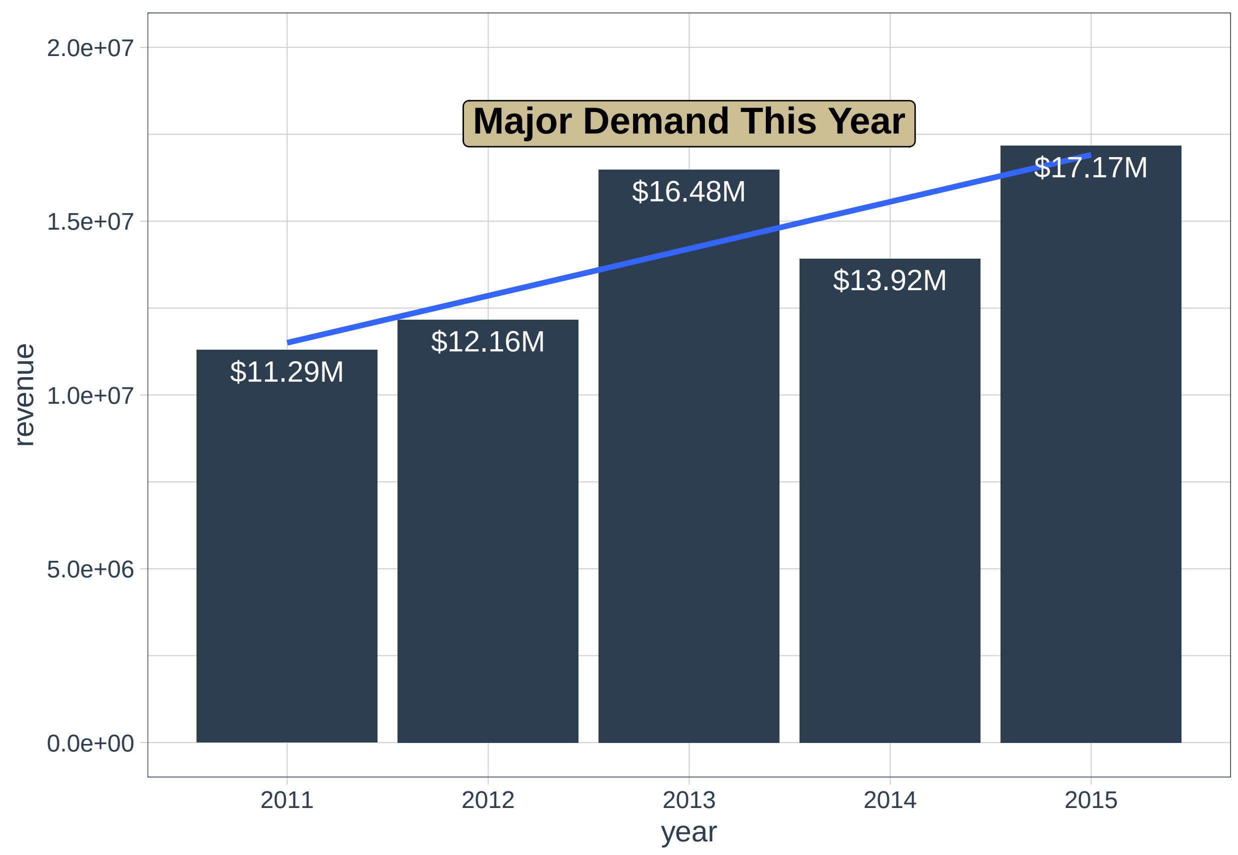
```
revenue_by_year_tbl %>%
    mutate(revenue_text = scales::dollar(revenue, scale = 1e-6, suffix = "M")) %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill="#2c3e50") +
    geom_label(aes(label = revenue_text), vjust = 0.05, size = 5) +
    expand_limits(y = 2e7) +
    theme_tq()
```



```r
revenue_by_year_tbl %>%
    mutate(revenue_text = scales::dollar(revenue, scale = 1e-6, suffix = "M")) %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill="#2c3e50") +
    geom_smooth(method = "lm", se = FALSE) +
    geom_text(aes(label = revenue_text), vjust = 1.5, color = "white") +
    geom_label(label = "Major Demand This Year",
               vjust = -0.5,
               size = 5,
               fontface = "bold",
               fill = palette_light()[4],
               data = revenue_by_year_tbl %>%
                   filter(year %in% c(2013))) +
    expand_limits(y = 2e7) +
    theme_tq()
```

Filtering labels to highlight a point

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



3.0 ggplot2 Formatting

3A. Data Manipulation

fct_reorder : reorder by one axis (e.g. revenue)

fct_reorder2 : reorder by two axis (e.g. year, revenue)

```
sales_by_year_category_2_tbl <- bike_orderlines_tbl %>%
    select(order_date, category_2, total_price) %>%

    mutate(order_date = ymd(order_date)) %>%
    mutate(year = year(order_date)) %>%

    group_by(category_2, year) %>%
    summarize(revenue = sum(total_price)) %>%
    ungroup() %>%

    mutate(category_2 = fct_reorder2(category_2, year, revenue))

sales_by_year_category_2_tbl
```

```
## # A tibble: 45 x 3
##    category_2          year revenue
##    <fct>              <dbl>   <dbl>
##  1 Cross Country Race  2011 2917250
##  2 Cross Country Race  2012 3360800
##  3 Cross Country Race  2013 4315430
##  4 Cross Country Race  2014 3691780
##  5 Cross Country Race  2015 4939370
##  6 Cyclocross          2011  378980
##  7 Cyclocross          2012  342090
##  8 Cyclocross          2013  503580
##  9 Cyclocross          2014  390250
## 10 Cyclocross          2015  493220
## # ... with 35 more rows
```

```
sales_by_year_category_2_tbl %>%
    mutate(category_2_num = as.numeric(category_2)) %>%
    arrange(category_2_num)
```

```
## # A tibble: 45 x 4
##    category_2          year revenue category_2_num
##    <fct>              <dbl>   <dbl>          <dbl>
##  1 Cross Country Race  2011 2917250              1
##  2 Cross Country Race  2012 3360800              1
##  3 Cross Country Race  2013 4315430              1
##  4 Cross Country Race  2014 3691780              1
##  5 Cross Country Race  2015 4939370              1
##  6 Elite Road          2011 2493315              2
##  7 Elite Road          2012 2637935              2
##  8 Elite Road          2013 3394210              2
##  9 Elite Road          2014 3170125              2
## 10 Elite Road          2015 3639080              2
## # ... with 35 more rows
```

3B working with colors

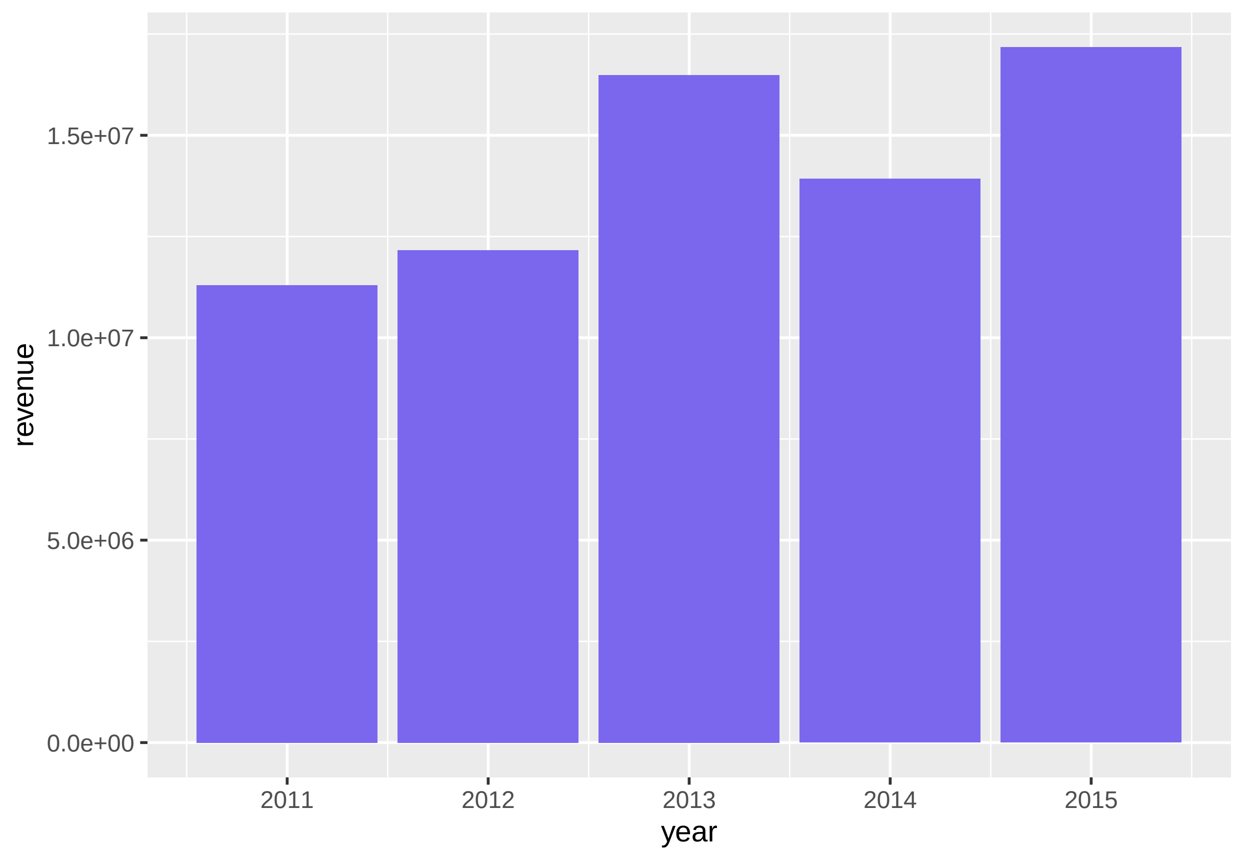
It is important to be comfrontable working with colors

3B.1 Color Conversion

```
colours()
```

```
##   [1] "white"                "aliceblue"            "antiquewhite"
##   [4] "antiquewhite1"        "antiquewhite2"        "antiquewhite3"
##   [7] "antiquewhite4"        "aquamarine"           "aquamarine1"
##  [10] "aquamarine2"          "aquamarine3"          "aquamarine4"
##  [13] "azure"                "azure1"               "azure2"
##  [16] "azure3"               "azure4"               "beige"
##  [19] "bisque"               "bisque1"              "bisque2"
##  [22] "bisque3"              "bisque4"              "black"
##  [25] "blanchedalmond"       "blue"                 "blue1"
##  [28] "blue2"                "blue3"                "blue4"
##  [31] "blueviolet"           "brown"                "brown1"
##  [34] "brown2"               "brown3"               "brown4"
##  [37] "burlywood"            "burlywood1"           "burlywood2"
##  [40] "burlywood3"           "burlywood4"           "cadetblue"
##  [43] "cadetblue1"           "cadetblue2"           "cadetblue3"
##  [46] "cadetblue4"           "chartreuse"           "chartreuse1"
##  [49] "chartreuse2"          "chartreuse3"          "chartreuse4"
##  [52] "chocolate"            "chocolate1"           "chocolate2"
##  [55] "chocolate3"           "chocolate4"           "coral"
##  [58] "coral1"               "coral2"               "coral3"
##  [61] "coral4"               "cornflowerblue"       "cornsilk"
##  [64] "cornsilk1"            "cornsilk2"            "cornsilk3"
##  [67] "cornsilk4"            "cyan"                 "cyan1"
##  [70] "cyan2"                "cyan3"                "cyan4"
##  [73] "darkblue"             "darkcyan"             "darkgoldenrod"
##  [76] "darkgoldenrod1"       "darkgoldenrod2"       "darkgoldenrod3"
##  [79] "darkgoldenrod4"       "darkgray"             "darkgreen"
##  [82] "darkgrey"             "darkkhaki"            "darkmagenta"
##  [85] "darkolivegreen"       "darkolivegreen1"      "darkolivegreen2"
##  [88] "darkolivegreen3"      "darkolivegreen4"      "darkorange"
##  [91] "darkorange1"          "darkorange2"          "darkorange3"
##  [94] "darkorange4"          "darkorchid"           "darkorchid1"
##  [97] "darkorchid2"          "darkorchid3"          "darkorchid4"
## [100] "darkred"              "darksalmon"           "darkseagreen"
## [103] "darkseagreen1"        "darkseagreen2"        "darkseagreen3"
## [106] "darkseagreen4"        "darkslateblue"        "darkslategray"
## [109] "darkslategray1"       "darkslategray2"       "darkslategray3"
## [112] "darkslategray4"       "darkslategrey"        "darkturquoise"
## [115] "darkviolet"           "deeppink"             "deeppink1"
## [118] "deeppink2"            "deeppink3"            "deeppink4"
## [121] "deepskyblue"          "deepskyblue1"         "deepskyblue2"
## [124] "deepskyblue3"         "deepskyblue4"         "dimgray"
## [127] "dimgrey"              "dodgerblue"           "dodgerblue1"
## [130] "dodgerblue2"          "dodgerblue3"          "dodgerblue4"
## [133] "firebrick"            "firebrick1"           "firebrick2"
## [136] "firebrick3"           "firebrick4"           "floralwhite"
## [139] "forestgreen"          "gainsboro"            "ghostwhite"
## [142] "gold"                 "gold1"                "gold2"
## [145] "gold3"                "gold4"                "goldenrod"
## [148] "goldenrod1"           "goldenrod2"           "goldenrod3"
## [151] "goldenrod4"           "gray"                 "gray0"
## [154] "gray1"                "gray2"                "gray3"
## [157] "gray4"                "gray5"                "gray6"
## [160] "gray7"                "gray8"                "gray9"
## [163] "gray10"               "gray11"               "gray12"
## [166] "gray13"               "gray14"               "gray15"
## [169] "gray16"               "gray17"               "gray18"
## [172] "gray19"               "gray20"               "gray21"
## [175] "gray22"               "gray23"               "gray24"
## [178] "gray25"               "gray26"               "gray27"
## [181] "gray28"               "gray29"               "gray30"
## [184] "gray31"               "gray32"               "gray33"
## [187] "gray34"               "gray35"               "gray36"
## [190] "gray37"               "gray38"               "gray39"
## [193] "gray40"               "gray41"               "gray42"
## [196] "gray43"               "gray44"               "gray45"
## [199] "gray46"               "gray47"               "gray48"
## [202] "gray49"               "gray50"               "gray51"
## [205] "gray52"               "gray53"               "gray54"
## [208] "gray55"               "gray56"               "gray57"
## [211] "gray58"               "gray59"               "gray60"
## [214] "gray61"               "gray62"               "gray63"
## [217] "gray64"               "gray65"               "gray66"
## [220] "gray67"               "gray68"               "gray69"
## [223] "gray70"               "gray71"               "gray72"
## [226] "gray73"               "gray74"               "gray75"
## [229] "gray76"               "gray77"               "gray78"
## [232] "gray79"               "gray80"               "gray81"
## [235] "gray82"               "gray83"               "gray84"
## [238] "gray85"               "gray86"               "gray87"
## [241] "gray88"               "gray89"               "gray90"
## [244] "gray91"               "gray92"               "gray93"
## [247] "gray94"               "gray95"               "gray96"
## [250] "gray97"               "gray98"               "gray99"
## [253] "gray100"              "green"                "green1"
## [256] "green2"               "green3"               "green4"
## [259] "greenyellow"          "grey"                 "grey0"
## [262] "grey1"                "grey2"                "grey3"
## [265] "grey4"                "grey5"                "grey6"
## [268] "grey7"                "grey8"                "grey9"
## [271] "grey10"               "grey11"               "grey12"
## [274] "grey13"               "grey14"               "grey15"
## [277] "grey16"               "grey17"               "grey18"
## [280] "grey19"               "grey20"               "grey21"
## [283] "grey22"               "grey23"               "grey24"
## [286] "grey25"               "grey26"               "grey27"
## [289] "grey28"               "grey29"               "grey30"
## [292] "grey31"               "grey32"               "grey33"
## [295] "grey34"               "grey35"               "grey36"
## [298] "grey37"               "grey38"               "grey39"
## [301] "grey40"               "grey41"               "grey42"
## [304] "grey43"               "grey44"               "grey45"
## [307] "grey46"               "grey47"               "grey48"
## [310] "grey49"               "grey50"               "grey51"
## [313] "grey52"               "grey53"               "grey54"
## [316] "grey55"               "grey56"               "grey57"
## [319] "grey58"               "grey59"               "grey60"
## [322] "grey61"               "grey62"               "grey63"
## [325] "grey64"               "grey65"               "grey66"
## [328] "grey67"               "grey68"               "grey69"
## [331] "grey70"               "grey71"               "grey72"
## [334] "grey73"               "grey74"               "grey75"
## [337] "grey76"               "grey77"               "grey78"
## [340] "grey79"               "grey80"               "grey81"
## [343] "grey82"               "grey83"               "grey84"
## [346] "grey85"               "grey86"               "grey87"
## [349] "grey88"               "grey89"               "grey90"
## [352] "grey91"               "grey92"               "grey93"
## [355] "grey94"               "grey95"               "grey96"
## [358] "grey97"               "grey98"               "grey99"
## [361] "grey100"              "honeydew"             "honeydew1"
## [364] "honeydew2"            "honeydew3"            "honeydew4"
## [367] "hotpink"              "hotpink1"             "hotpink2"
## [370] "hotpink3"             "hotpink4"             "indianred"
## [373] "indianred1"           "indianred2"           "indianred3"
## [376] "indianred4"           "ivory"                "ivory1"
## [379] "ivory2"               "ivory3"               "ivory4"
## [382] "khaki"                "khaki1"               "khaki2"
## [385] "khaki3"               "khaki4"               "lavender"
## [388] "lavenderblush"        "lavenderblush1"       "lavenderblush2"
## [391] "lavenderblush3"       "lavenderblush4"       "lawngreen"
## [394] "lemonchiffon"         "lemonchiffon1"        "lemonchiffon2"
## [397] "lemonchiffon3"        "lemonchiffon4"        "lightblue"
## [400] "lightblue1"           "lightblue2"           "lightblue3"
## [403] "lightblue4"           "lightcoral"           "lightcyan"
## [406] "lightcyan1"           "lightcyan2"           "lightcyan3"
## [409] "lightcyan4"           "lightgoldenrod"       "lightgoldenrod1"
## [412] "lightgoldenrod2"      "lightgoldenrod3"      "lightgoldenrod4"
## [415] "lightgoldenrodyellow" "lightgray"            "lightgreen"
## [418] "lightgrey"            "lightpink"            "lightpink1"
## [421] "lightpink2"           "lightpink3"           "lightpink4"
## [424] "lightsalmon"          "lightsalmon1"         "lightsalmon2"
## [427] "lightsalmon3"         "lightsalmon4"         "lightseagreen"
## [430] "lightskyblue"         "lightskyblue1"        "lightskyblue2"
## [433] "lightskyblue3"        "lightskyblue4"        "lightslateblue"
## [436] "lightslategray"       "lightslategrey"       "lightsteelblue"
## [439] "lightsteelblue1"      "lightsteelblue2"      "lightsteelblue3"
## [442] "lightsteelblue4"      "lightyellow"          "lightyellow1"
## [445] "lightyellow2"         "lightyellow3"         "lightyellow4"
## [448] "limegreen"            "linen"                "magenta"
## [451] "magenta1"             "magenta2"             "magenta3"
## [454] "magenta4"             "maroon"               "maroon1"
## [457] "maroon2"              "maroon3"              "maroon4"
## [460] "mediumaquamarine"     "mediumblue"           "mediumorchid"
## [463] "mediumorchid1"        "mediumorchid2"        "mediumorchid3"
## [466] "mediumorchid4"        "mediumpurple"         "mediumpurple1"
## [469] "mediumpurple2"        "mediumpurple3"        "mediumpurple4"
## [472] "mediumseagreen"       "mediumslateblue"      "mediumspringgreen"
## [475] "mediumturquoise"      "mediumvioletred"      "midnightblue"
## [478] "mintcream"            "mistyrose"            "mistyrose1"
## [481] "mistyrose2"           "mistyrose3"           "mistyrose4"
## [484] "moccasin"             "navajowhite"          "navajowhite1"
## [487] "navajowhite2"         "navajowhite3"         "navajowhite4"
## [490] "navy"                 "navyblue"             "oldlace"
## [493] "olivedrab"            "olivedrab1"           "olivedrab2"
## [496] "olivedrab3"           "olivedrab4"           "orange"
## [499] "orange1"              "orange2"              "orange3"
## [502] "orange4"              "orangered"            "orangered1"
## [505] "orangered2"           "orangered3"           "orangered4"
## [508] "orchid"               "orchid1"              "orchid2"
## [511] "orchid3"              "orchid4"              "palegoldenrod"
## [514] "palegreen"           "palegreen1"           "palegreen2"
## [517] "palegreen3"          "palegreen4"           "paleturquoise"
## [520] "paleturquoise1"      "paleturquoise2"       "paleturquoise3"
## [523] "paleturquoise4"      "palevioletred"        "palevioletred1"
## [526] "palevioletred2"      "palevioletred3"       "palevioletred4"
## [529] "papayawhip"          "peachpuff"            "peachpuff1"
## [532] "peachpuff2"          "peachpuff3"           "peachpuff4"
## [535] "peru"                "pink"                 "pink1"
## [538] "pink2"               "pink3"                "pink4"
## [541] "plum"                "plum1"                "plum2"
## [544] "plum3"               "plum4"                "powderblue"
## [547] "purple"              "purple1"              "purple2"
## [550] "purple3"             "purple4"              "red"
## [553] "red1"                "red2"                 "red3"
## [556] "red4"                "rosybrown"            "rosybrown1"
## [559] "rosybrown2"          "rosybrown3"           "rosybrown4"
## [562] "royalblue"           "royalblue1"           "royalblue2"
## [565] "royalblue3"          "royalblue4"           "saddlebrown"
## [568] "salmon"              "salmon1"              "salmon2"
## [571] "salmon3"             "salmon4"              "sandybrown"
## [574] "seagreen"            "seagreen1"            "seagreen2"
## [577] "seagreen3"           "seagreen4"            "seashell"
## [580] "seashell1"           "seashell2"            "seashell3"
## [583] "seashell4"           "sienna"               "sienna1"
## [586] "sienna2"             "sienna3"              "sienna4"
## [589] "skyblue"             "skyblue1"             "skyblue2"
## [592] "skyblue3"            "skyblue4"             "slateblue"
## [595] "slateblue1"          "slateblue2"           "slateblue3"
## [598] "slateblue4"          "slategray"            "slategray1"
## [601] "slategray2"          "slategray3"           "slategray4"
## [604] "slategrey"           "snow"                 "snow1"
## [607] "snow2"               "snow3"                "snow4"
## [610] "springgreen"         "springgreen1"         "springgreen2"
## [613] "springgreen3"        "springgreen4"         "steelblue"
## [616] "steelblue1"          "steelblue2"           "steelblue3"
## [619] "steelblue4"          "tan"                  "tan1"
## [622] "tan2"                "tan3"                 "tan4"
## [625] "thistle"             "thistle1"             "thistle2"
## [628] "thistle3"            "thistle4"             "tomato"
## [631] "tomato1"             "tomato2"              "tomato3"
## [634] "tomato4"             "turquoise"            "turquoise1"
## [637] "turquoise2"          "turquoise3"           "turquoise4"
## [640] "violet"              "violetred"            "violetred1"
## [643] "violetred2"          "violetred3"           "violetred4"
## [646] "wheat"               "wheat1"               "wheat2"
## [649] "wheat3"              "wheat4"               "whitesmoke"
## [652] "yellow"              "yellow1"              "yellow2"
## [655] "yellow3"             "yellow4"              "yellowgreen"
```

```
sales_by_year_category_2_tbl %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill = "slateblue2")
```



To RGB : Specifying color values as combinations of Red - Green - Blue (e.g. White = 255 - 255 - 255)

```r
col2rgb("slateblue2")
```

```
##       [,1]
## red    122
## green  103
## blue   238
```

```r
col2rgb("#2C3E50")
```

```
##       [,1]
## red     44
## green   62
## blue    80
```

```r


rgb(44, 62, 80, maxColorValue = 225)
```

To Hex : Specifying a color by hexidecimal

```
## [1] "#32465B"
```

3B.2 Color Paletters

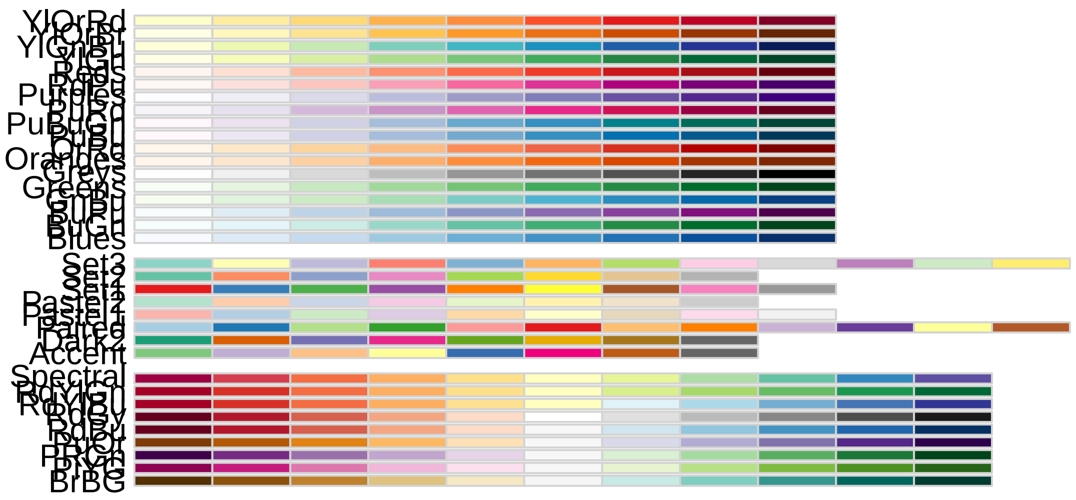
tidyquant

```
tidyquant::palette_light()[1] %>% col2rgb()
```

```
##       blue
## red     44
## green   62
## blue    80
```

Brewer : for discrete data

```
RColorBrewer::display.brewer.all()
```



```
RColorBrewer::brewer.pal.info %>% arrange(desc(maxcolors))
```

```
##         maxcolors category colorblind
## Paired         12     qual       TRUE
## Set3           12     qual      FALSE
## BrBG           11      div       TRUE
## PiYG           11      div       TRUE
## PRGn           11      div       TRUE
## PuOr           11      div       TRUE
## RdBu              11      div       TRUE
## RdGy              11      div      FALSE
## RdYlBu            11      div       TRUE
## RdYlGn            11      div      FALSE
## Spectral          11      div      FALSE
## Pastel1            9     qual      FALSE
## Set1               9     qual      FALSE
## Blues              9      seq       TRUE
## BuGn               9      seq       TRUE
## BuPu               9      seq       TRUE
## GnBu               9      seq       TRUE
## Greens             9      seq       TRUE
## Greys              9      seq       TRUE
## Oranges            9      seq       TRUE
## OrRd               9      seq       TRUE
## PuBu               9      seq       TRUE
## PuBuGn             9      seq       TRUE
## PuRd               9      seq       TRUE
## Purples            9      seq       TRUE
## RdPu               9      seq       TRUE
## Reds               9      seq       TRUE
## YlGn               9      seq       TRUE
## YlGnBu             9      seq       TRUE
## YlOrBr             9      seq       TRUE
## YlOrRd             9      seq       TRUE
## Accent             8     qual      FALSE
## Dark2              8     qual       TRUE
## Pastel2            8     qual      FALSE
## Set2               8     qual       TRUE
```

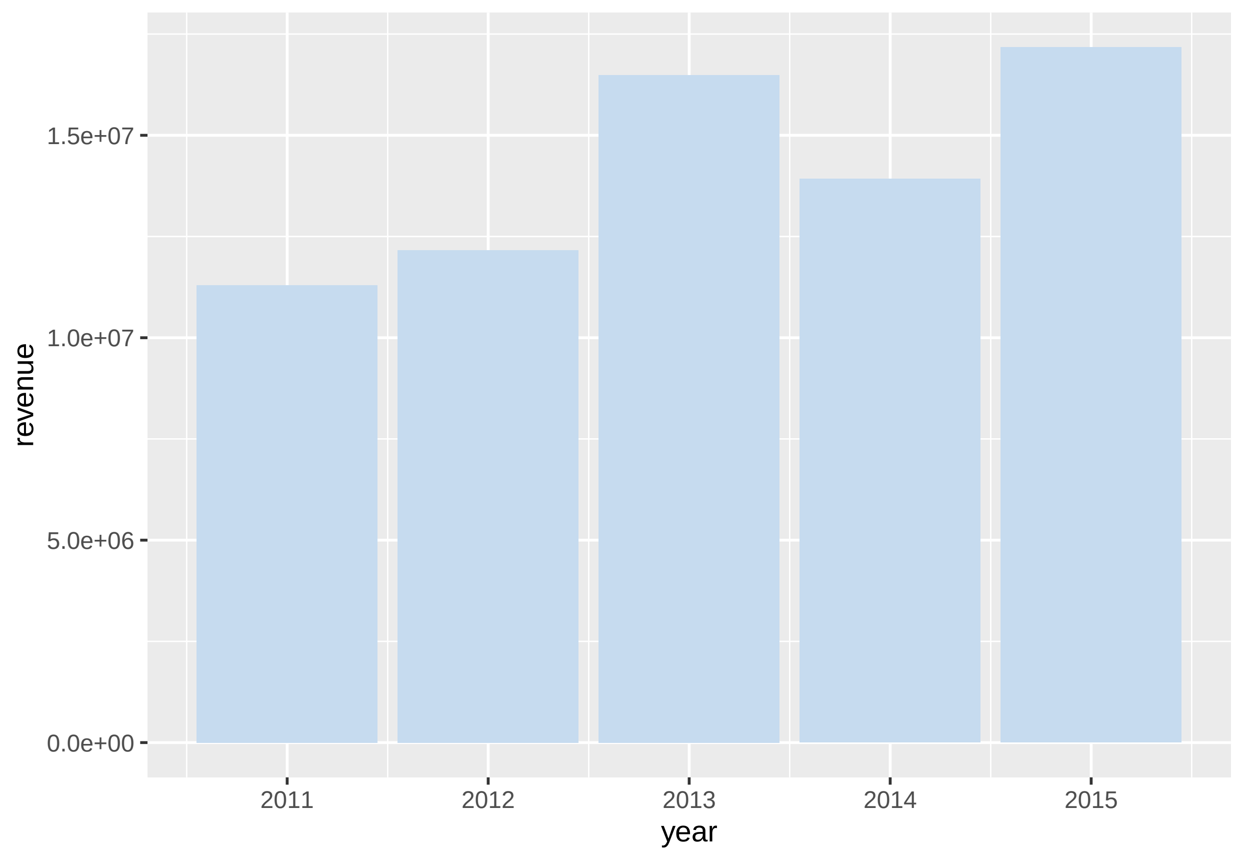
```
RColorBrewer::brewer.pal(n = 100, name = "Blues")
```

```
## Warning in RColorBrewer::brewer.pal(n = 100, name = "Blues"): n too large, allowed maximum for palet
## Returning the palette you asked for with that many colors
```

```
## [1] "#F7FBFF" "#DEEBF7" "#C6DBEF" "#9ECAE1" "#6BAED6" "#4292C6" "#2171B5"
## [8] "#08519C" "#08306B"
```

```
sales_by_year_category_2_tbl %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill = RColorBrewer::brewer.pal(n = 100, name = "Blues")[3])
```

```
## Warning in RColorBrewer::brewer.pal(n = 100, name = "Blues"): n too large, allowed maximum for palet
## Returning the palette you asked for with that many colors
```

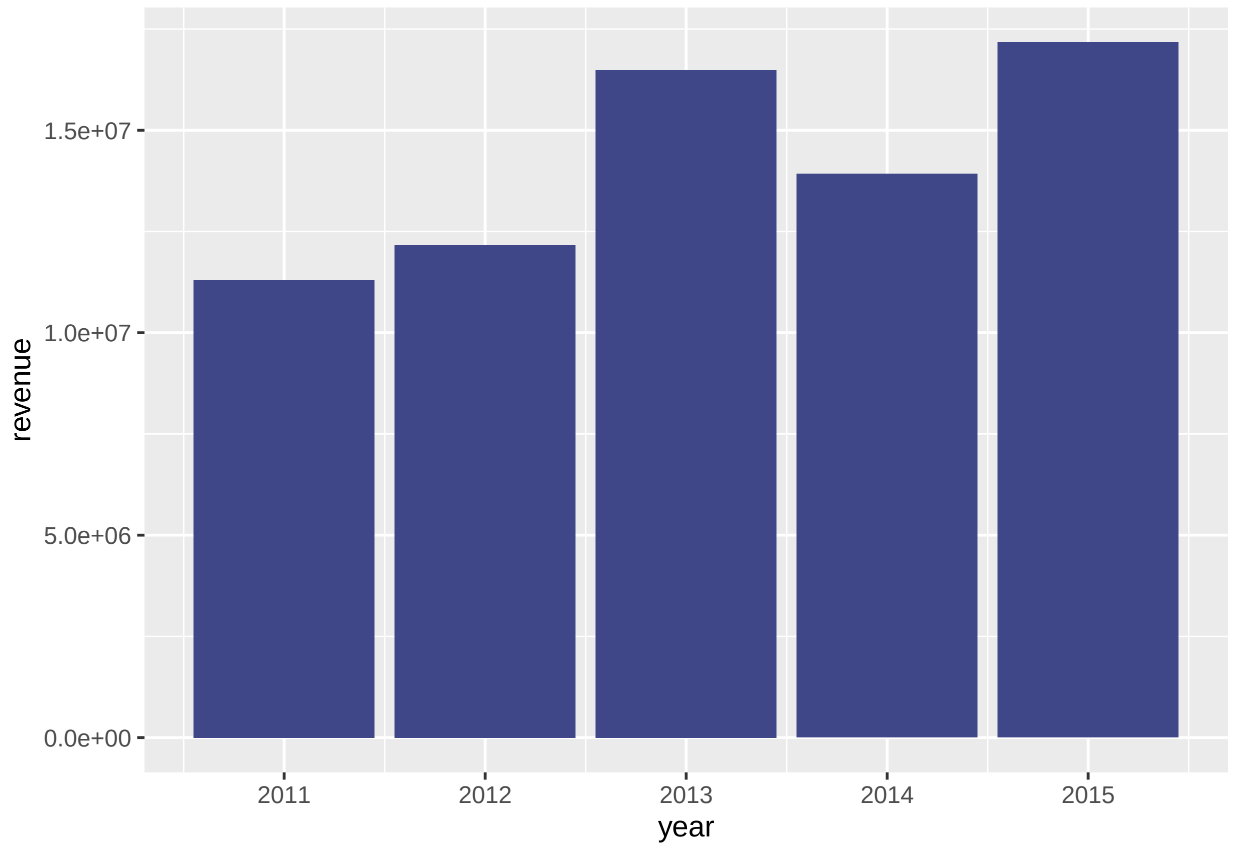


Viridis :

```
viridisLite::viridis(n = 20)
```

```
##  [1] "#440154FF" "#481568FF" "#482677FF" "#453781FF" "#3F4788FF" "#39558CFF"
##  [7] "#32648EFF" "#2D718EFF" "#287D8EFF" "#238A8DFF" "#1F968BFF" "#20A386FF"
## [13] "#29AF7FFF" "#3CBC75FF" "#56C667FF" "#74D055FF" "#94D840FF" "#B8DE29FF"
## [19] "#DCE318FF" "#FDE725FF"
```

```
sales_by_year_category_2_tbl %>%
    ggplot(aes(x = year, y = revenue)) +
    geom_col(fill = viridisLite::viridis(n = 20)[5])
```



4.0 Aesthetic Mappings

4B color

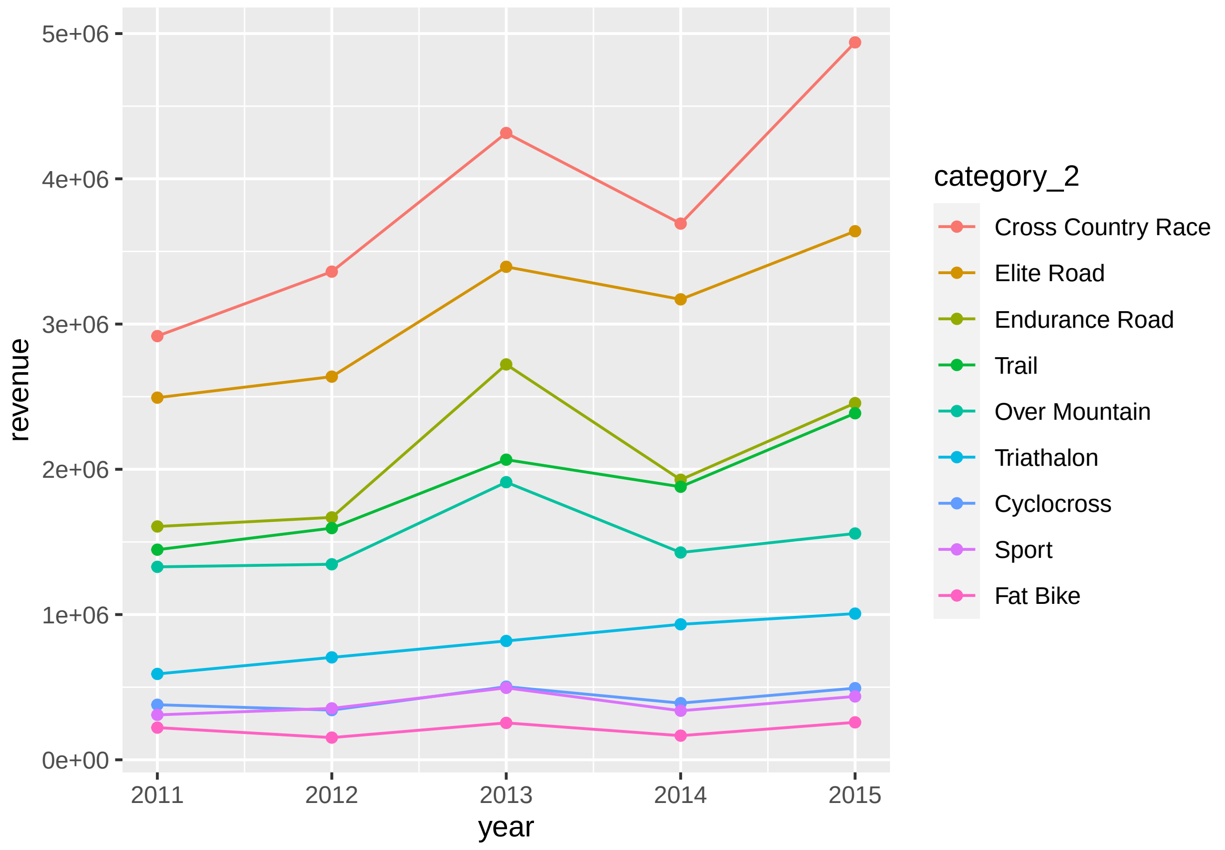
Used with line nd points, Outlines of rectangular objects

ggplot2 data format & Modeling data format are the same!

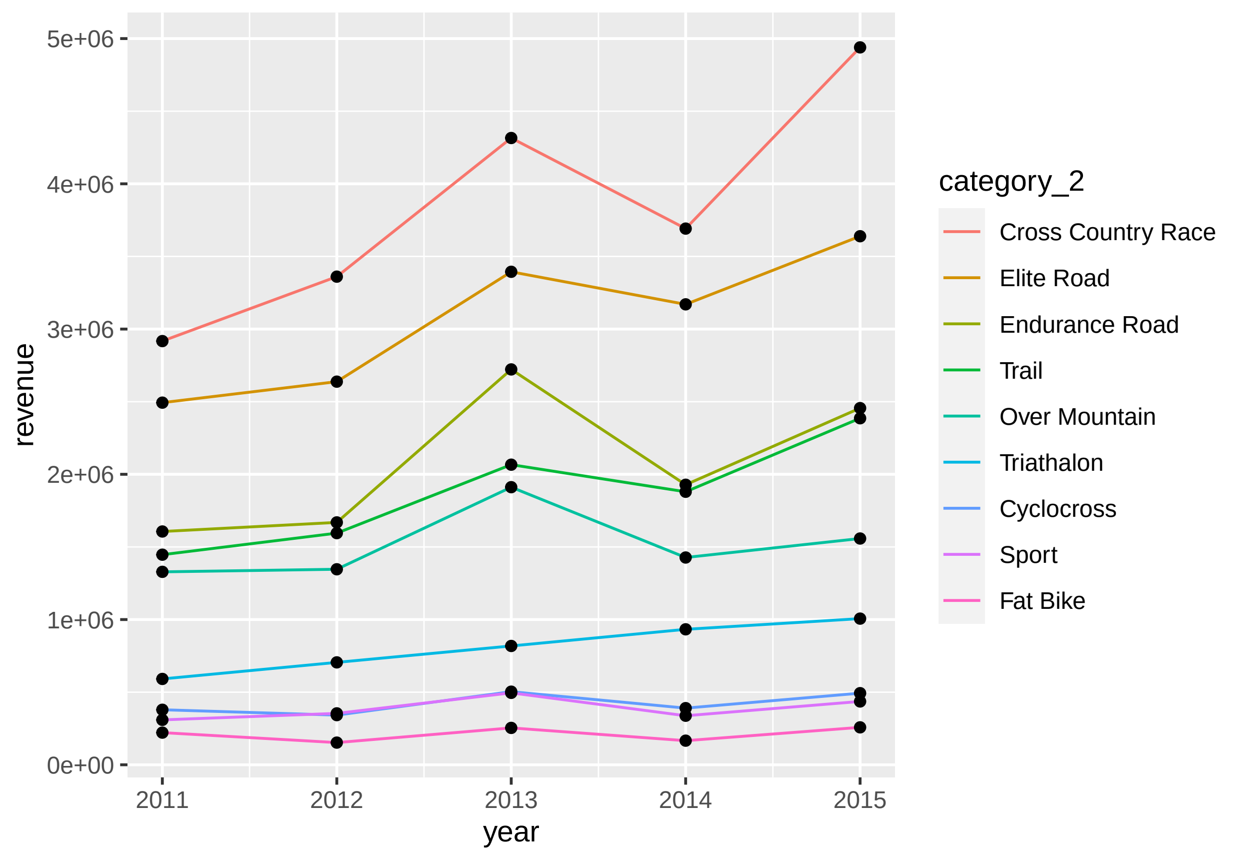
“Tidy Data”: One column of interest known as the target (e.g. target = revenue)

Other columns describe the target

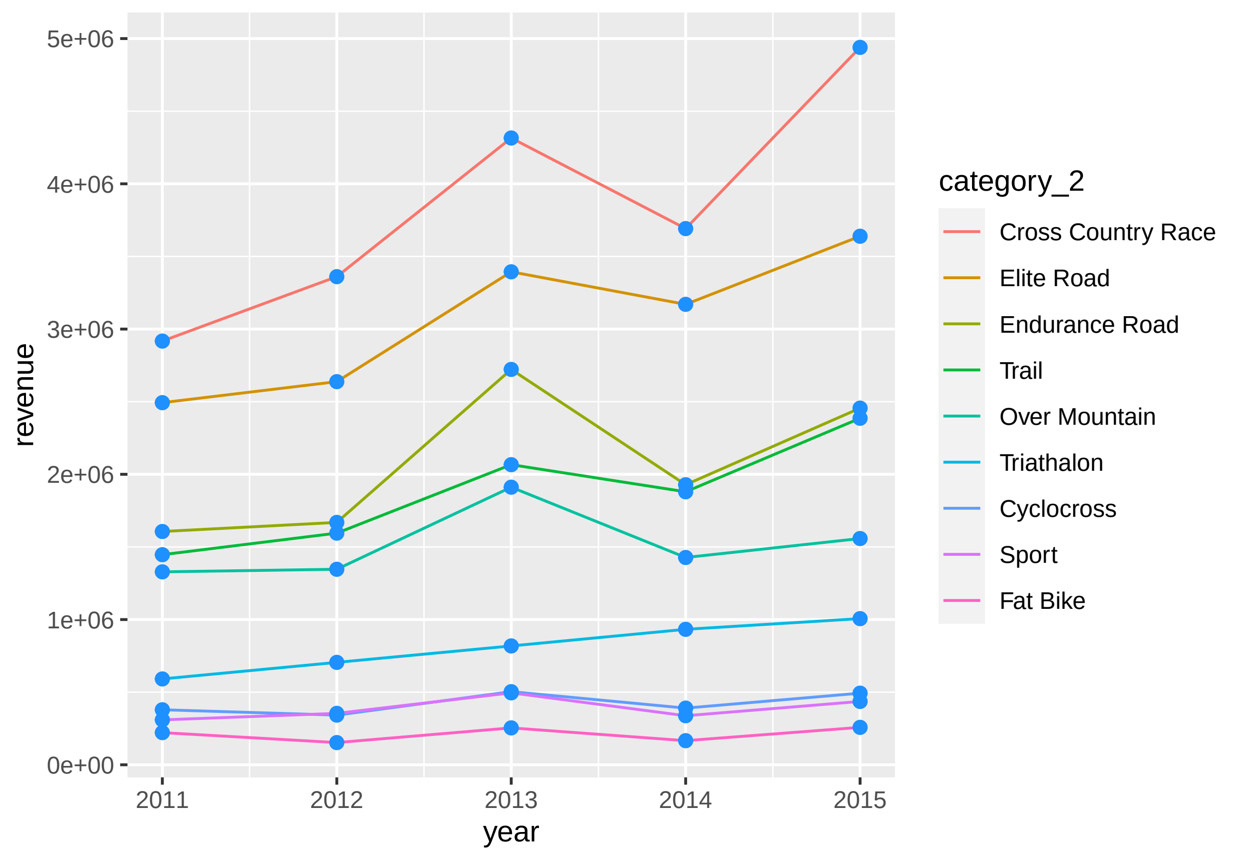
```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, color = category_2)) + # define Globally
    geom_line() +
    geom_point()
```



```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue)) +
    geom_line(aes(color = category_2)) +  # Define locally
    geom_point()
```



```r
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue)) +
    geom_line(aes(color = category_2)) +
    geom_point(color = "dodgerblue", size = 2)
```

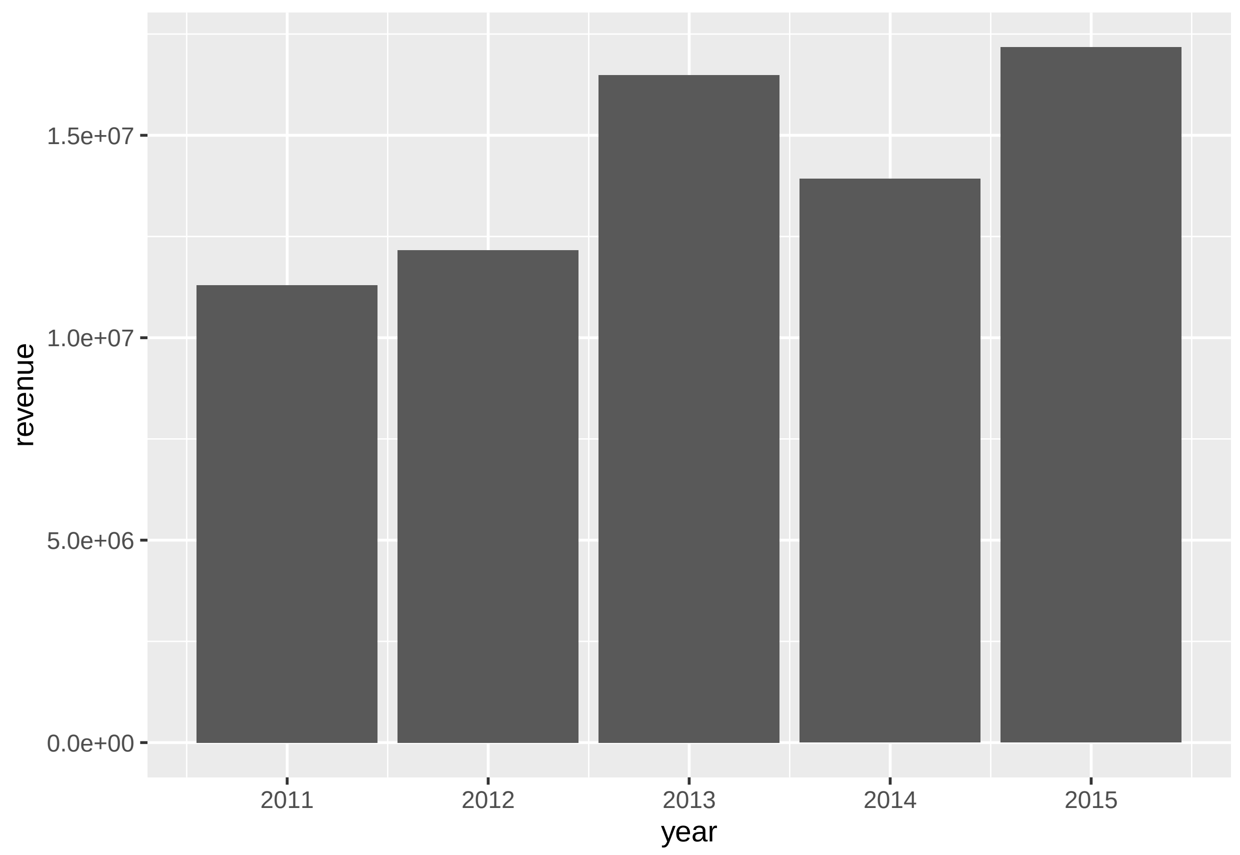


4B Fill

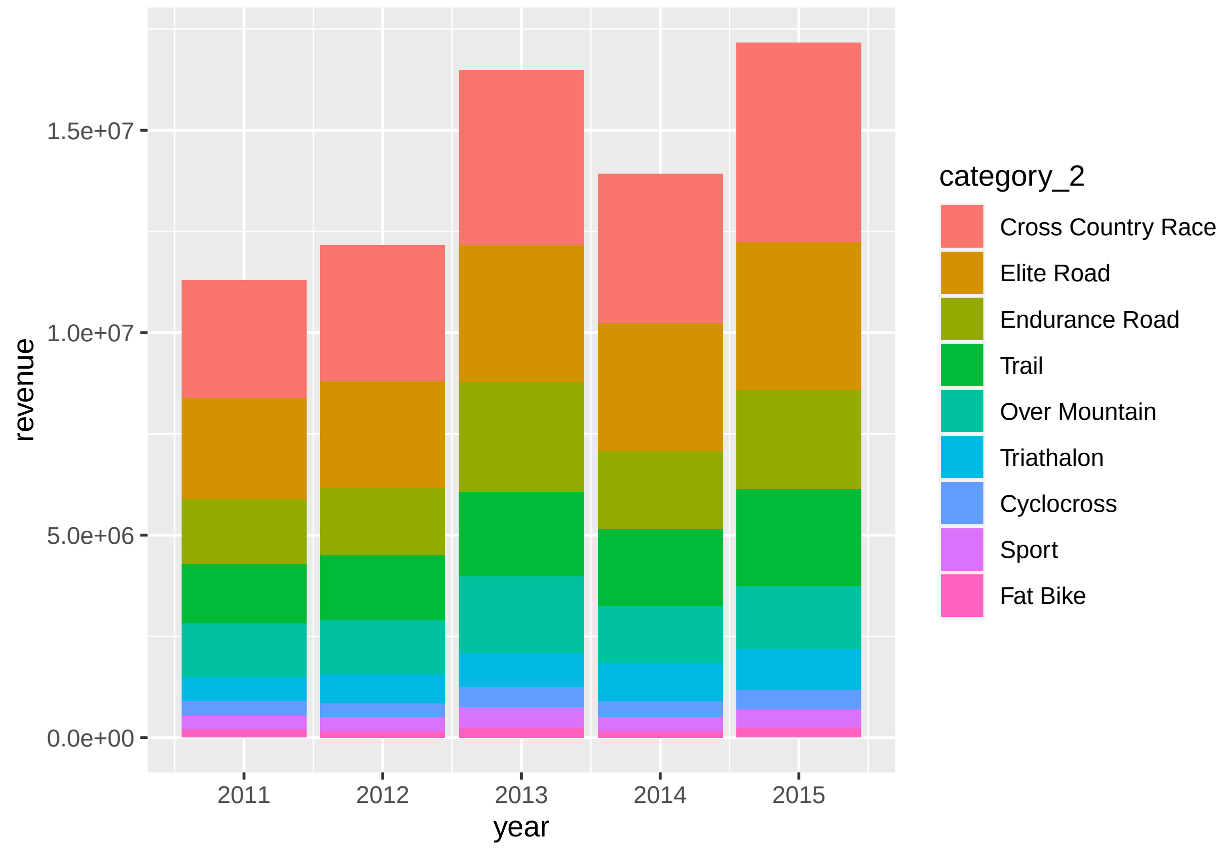
Info: * Used with fill of rectangular objects

- Do not confuse with colour and fill argument

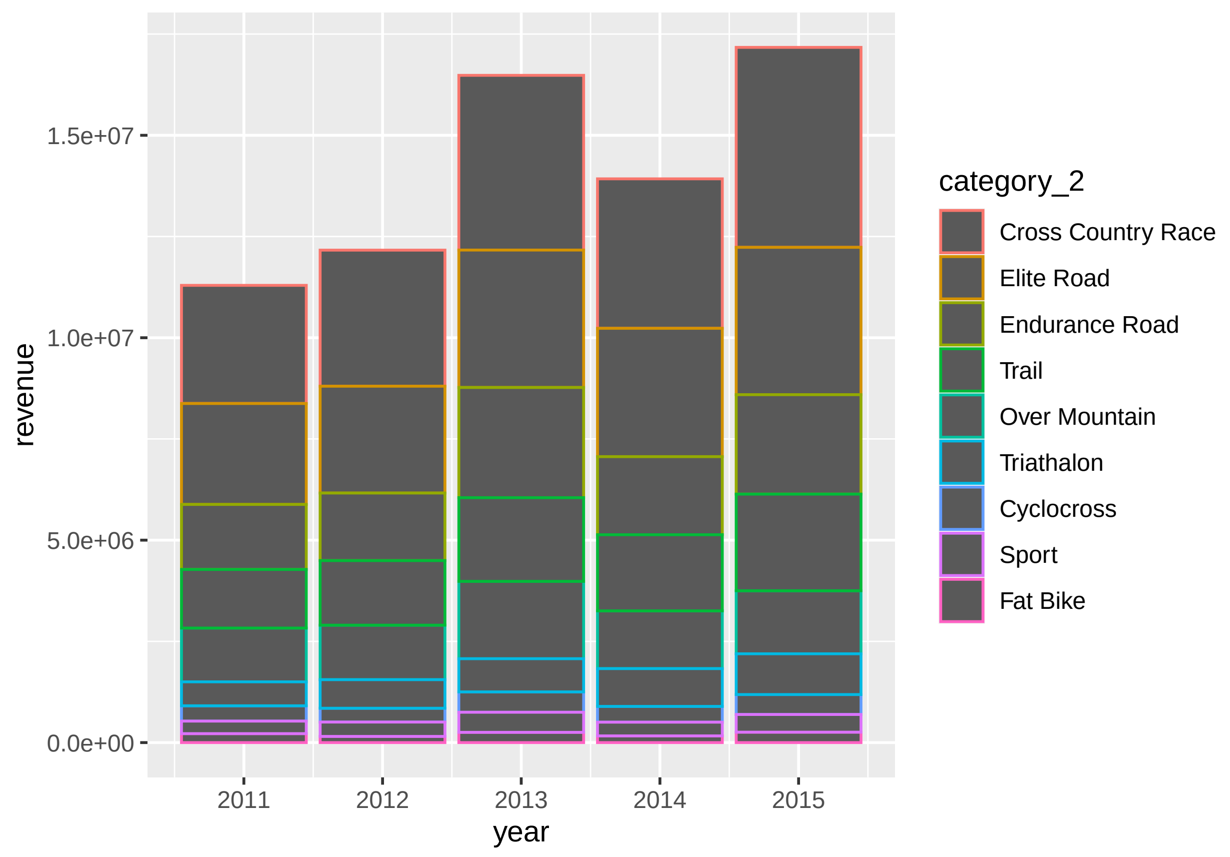
```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue)) + # define Globally
    geom_col()
```



```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, fill = category_2)) + # define Globally
    geom_col()
```



```
# Do not confuse fill with colour!!!
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, color = category_2)) + # define Globally
    geom_col()
```

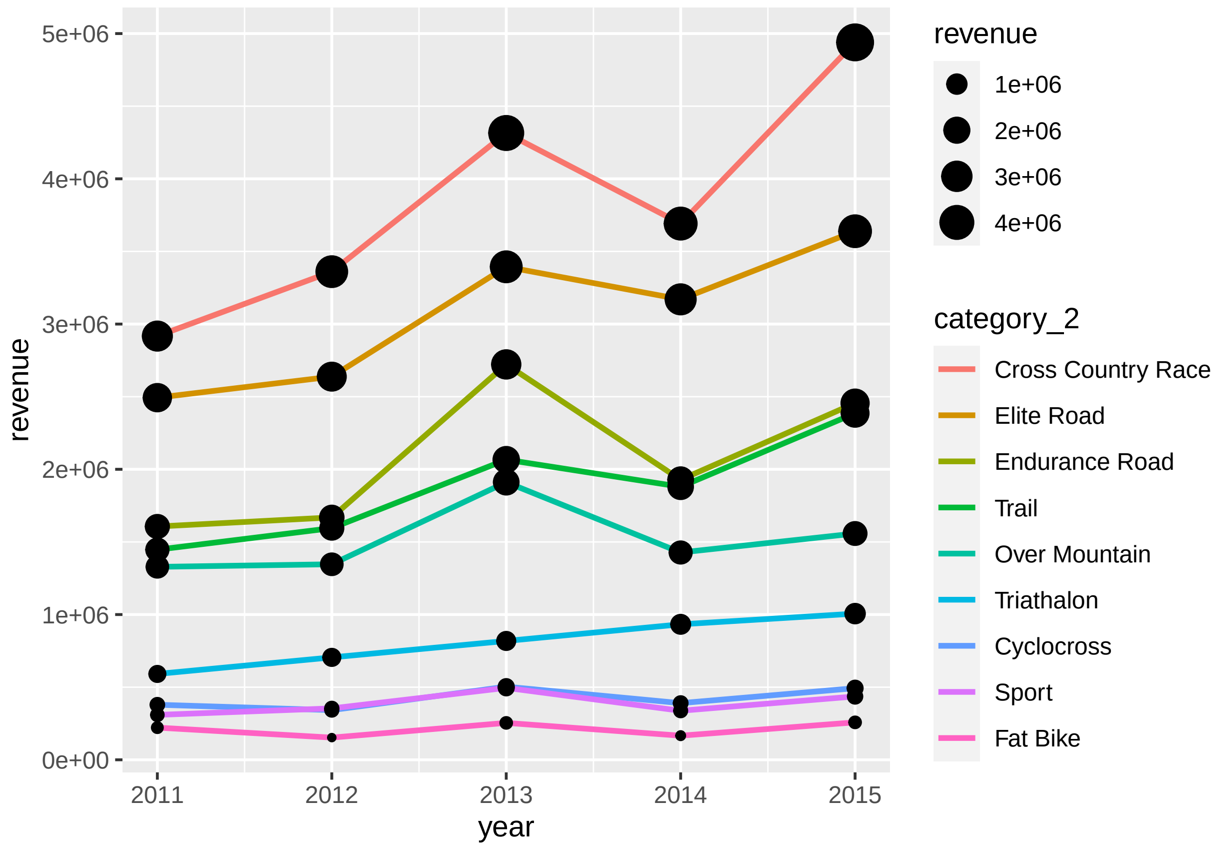


4B Size

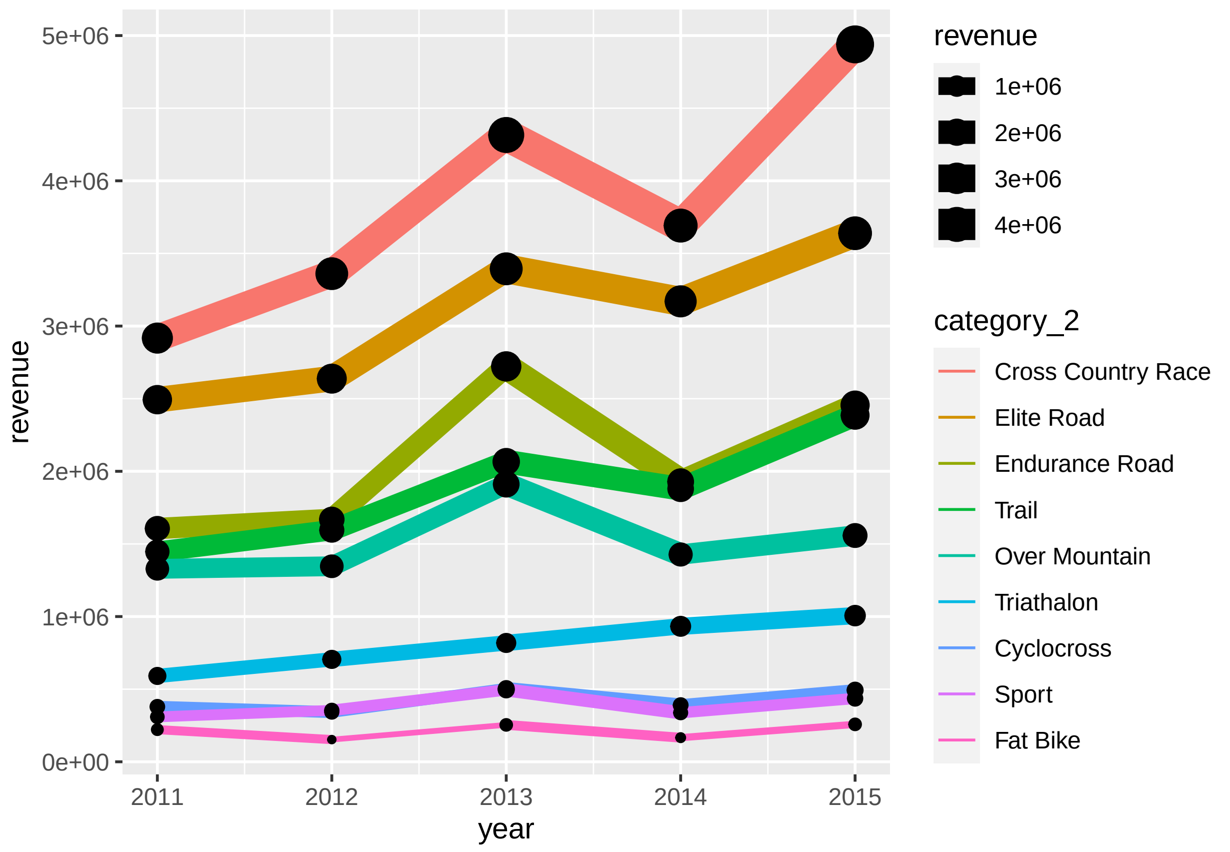
Info :

- Used with points

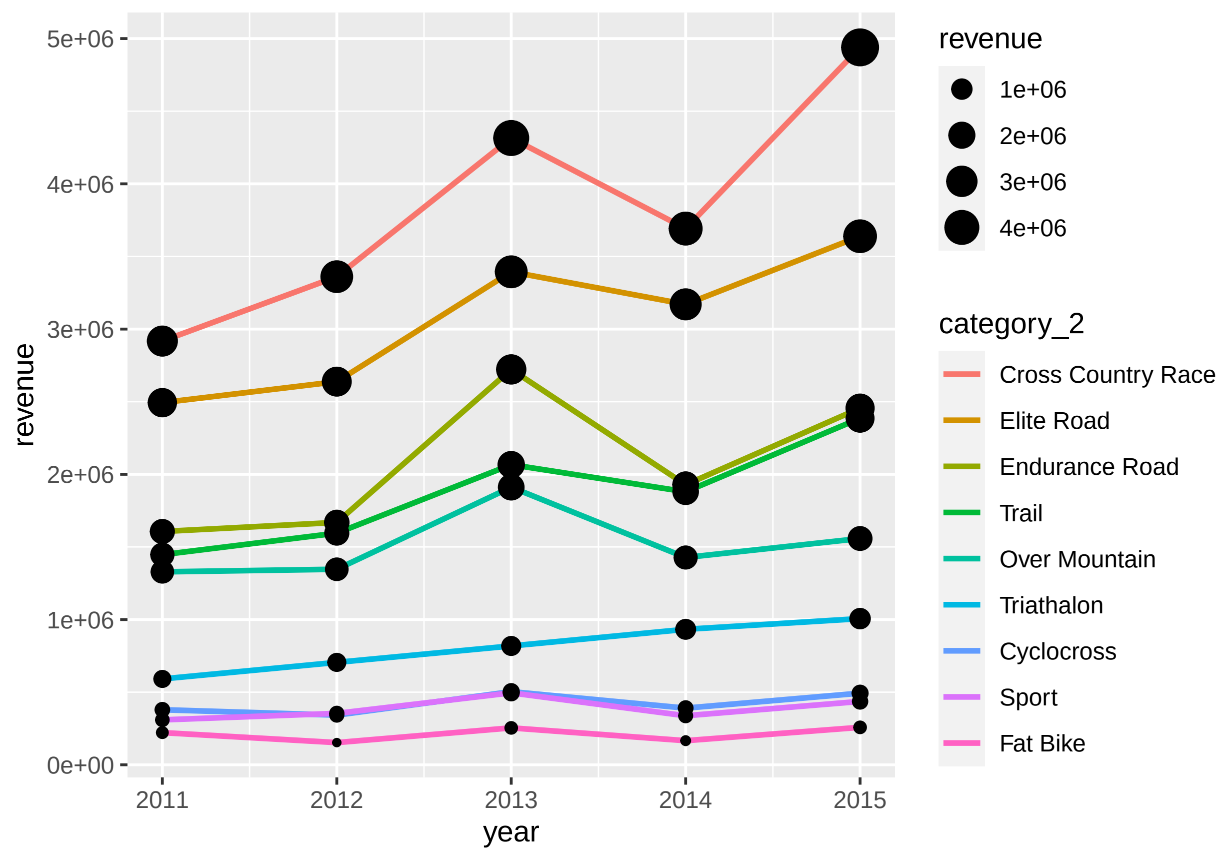
```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue)) +
    geom_line(aes(colour = category_2), size = 1) +
    geom_point(aes(size = revenue))
```



```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, size = revenue)) +
    geom_line(aes(colour = category_2)) +
    geom_point()
```



```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, size = revenue)) +
    geom_line(aes(colour = category_2), size = 1) +
    geom_point()
```



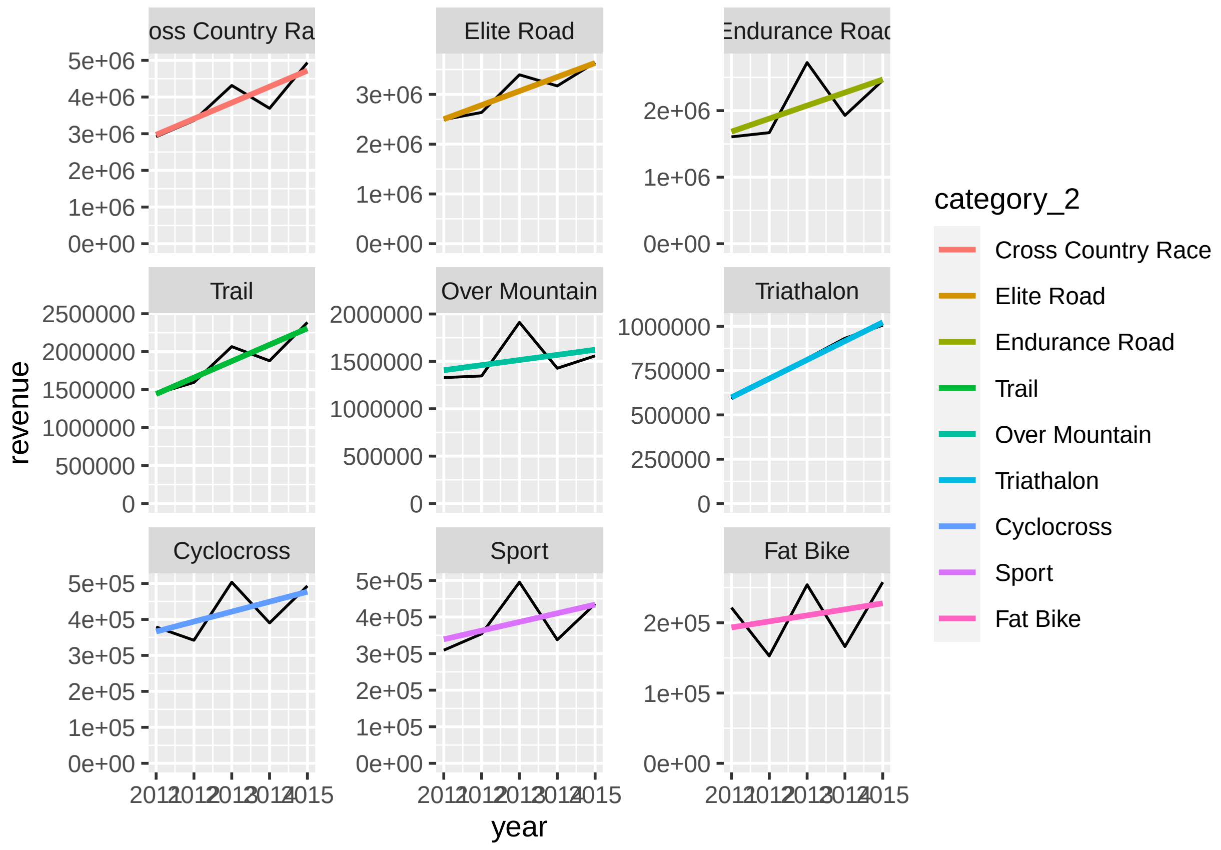
5.0 Faceting —

Great way to tease out variation by category

Goal: Sales annual sales by category 2

```
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, colour = category_2)) +
    geom_line(colour = "black") +
    geom_smooth(method = "lm", se = FALSE) +
    facet_wrap(~ category_2, ncol = 3, scales = "free_y") +
    expand_limits(y = 0)
```

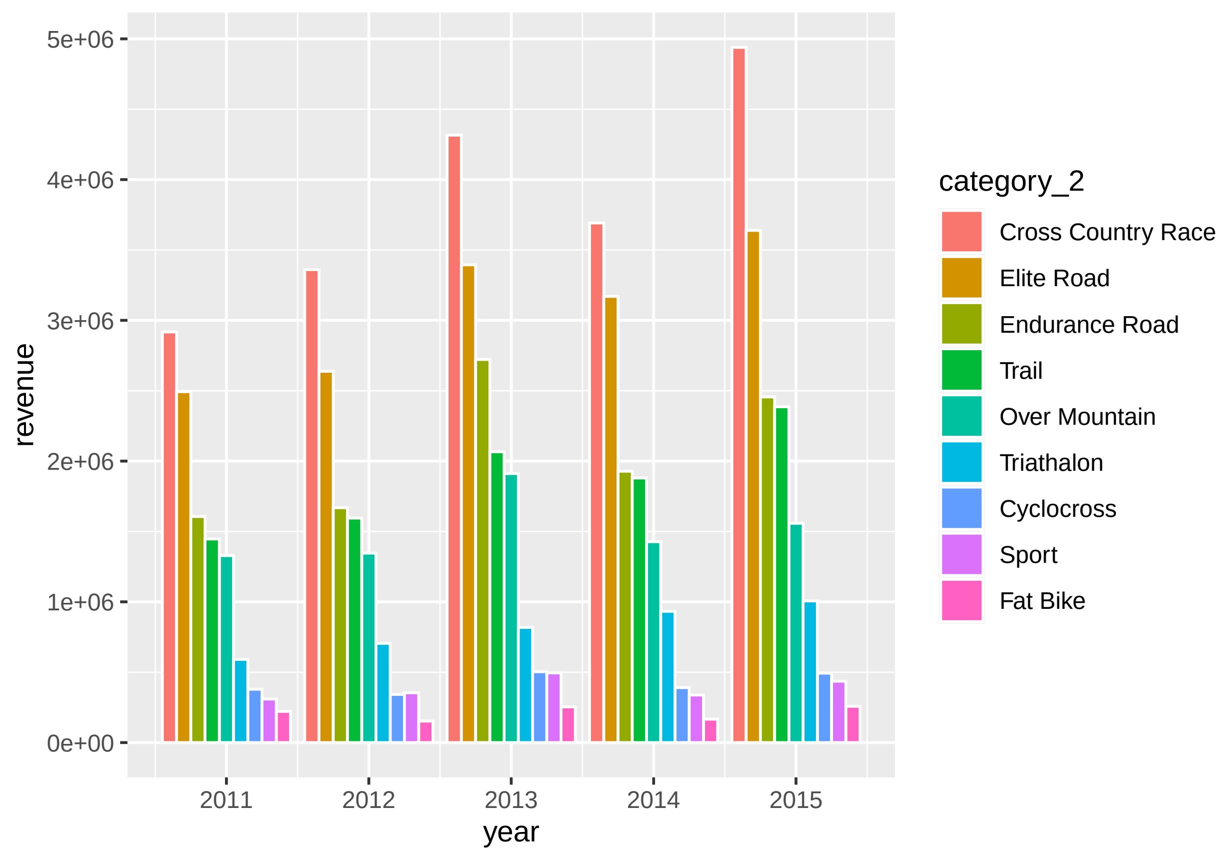
```
## `geom_smooth()` using formula 'y ~ x'
```



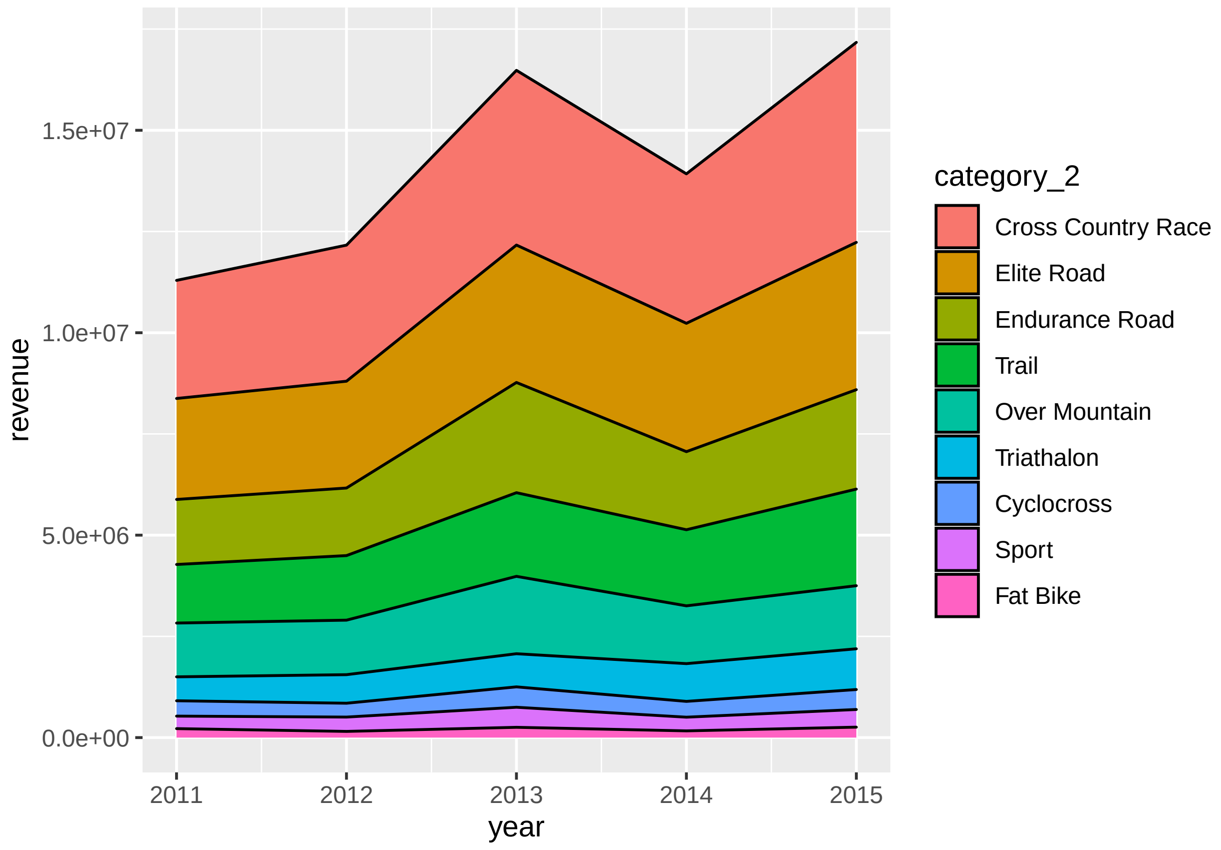
6.0 Position Adjustments (Stack & Dodge)

- Stacked Bars & Side-By-Side Bars
  - position = "Stack": Stack elements on top of each other
  - position_dodge(): Enables further customization of width and height attributes

```
# Stacked Bars & Side-By-Side Bars
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, fill = category_2)) +
    # geom_col(position = "stack") +
    # geom_col(position = "dodge") +
    geom_col(position = position_dodge(width = 0.9), color = "white")
```



```
# Stacked Area
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, fill = category_2)) +
    geom_area(colour = "black")
```

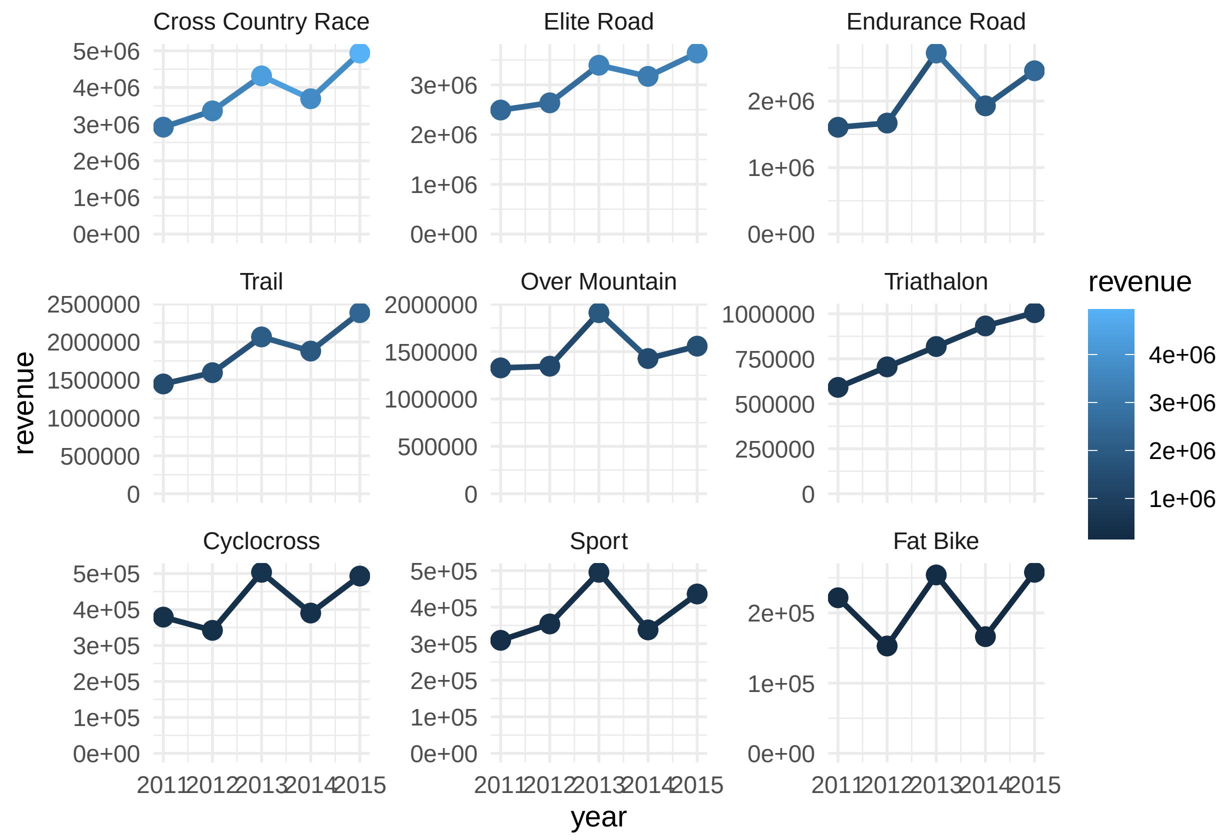


7.0 Scales (colours, Fills. Axis)

- continuous (e.g. Revenue): Changes color via gradient palette
- Categorical (e.g.): Changes colour via discrete palette

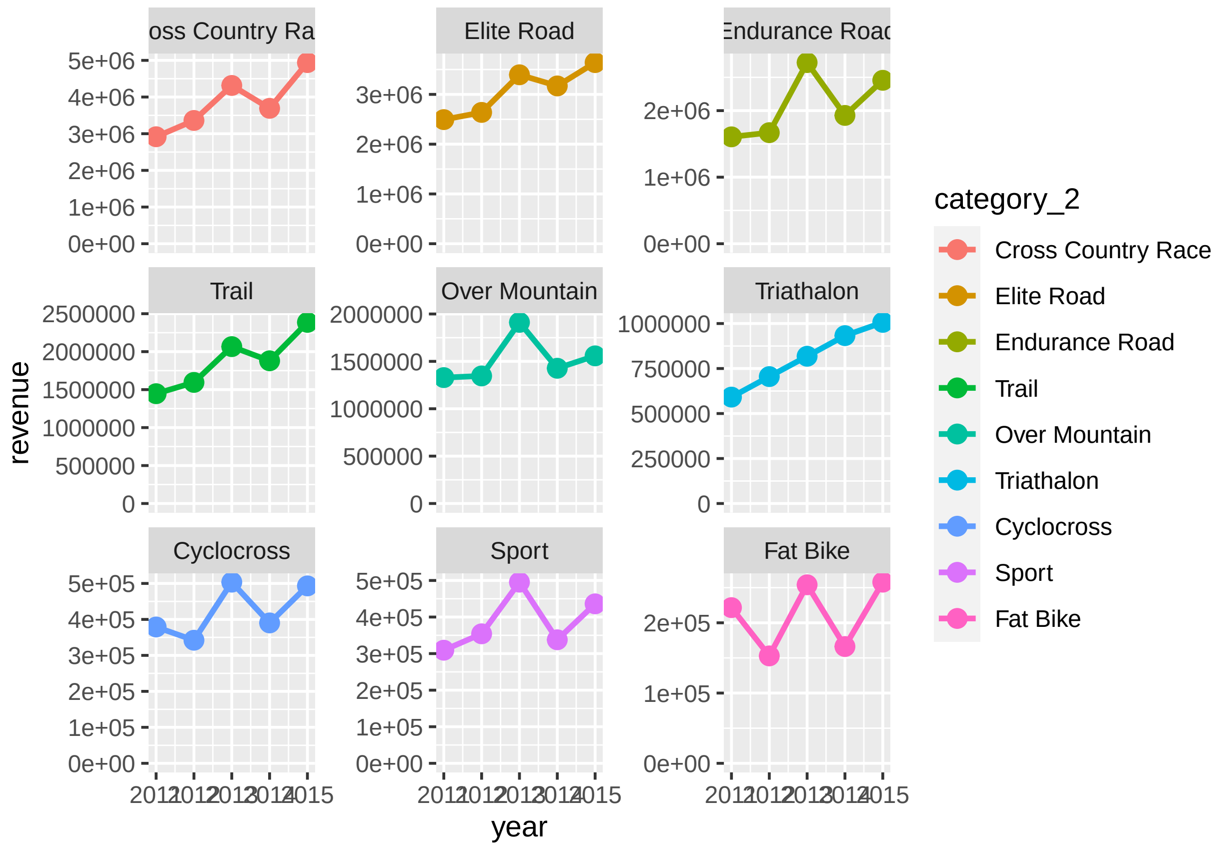
Plot 1: Faceted Plot, Colour = Continous Scale

```
g_facet_continous <- sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, colour =revenue)) +
    geom_line(size = 1) +
    geom_point(size = 3) +

    facet_wrap(~ category_2, scales = "free_y") +
    expand_limits(y = 0) +
    theme_minimal()

g_facet_continous
```



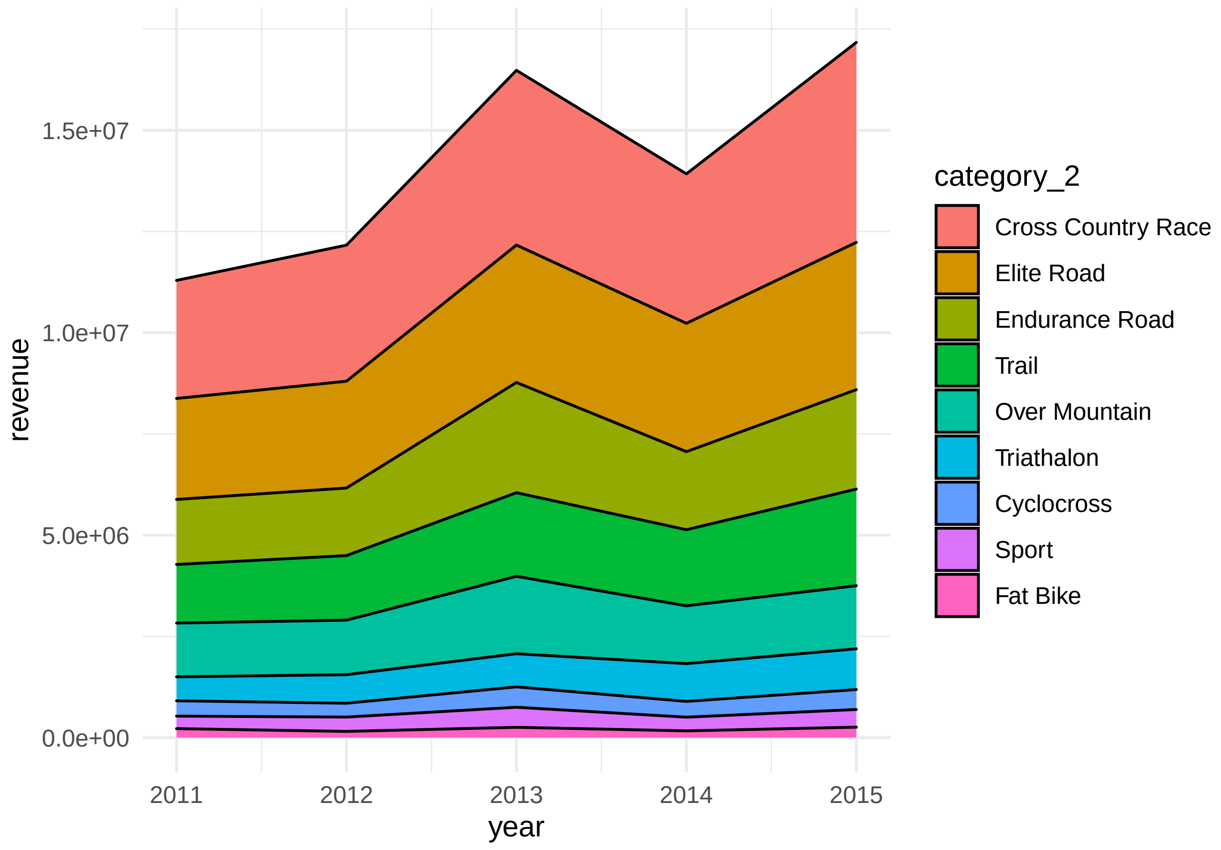
Plot 2: Faceted Plot, Colour = Discrete Scale

```
g_facet_discrete <- sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, colour = category_2)) +
    geom_line(size = 1) +
    geom_point(size = 3) +
    facet_wrap(~category_2, scales = "free_y") +
    expand_limits(y = 0)

g_facet_discrete
```



Plot 2: Stacked Area plot

```r
g_area_discrete <- sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, fill = category_2)) +
    geom_area(colour = "black") +
    theme_minimal()

g_area_discrete
```

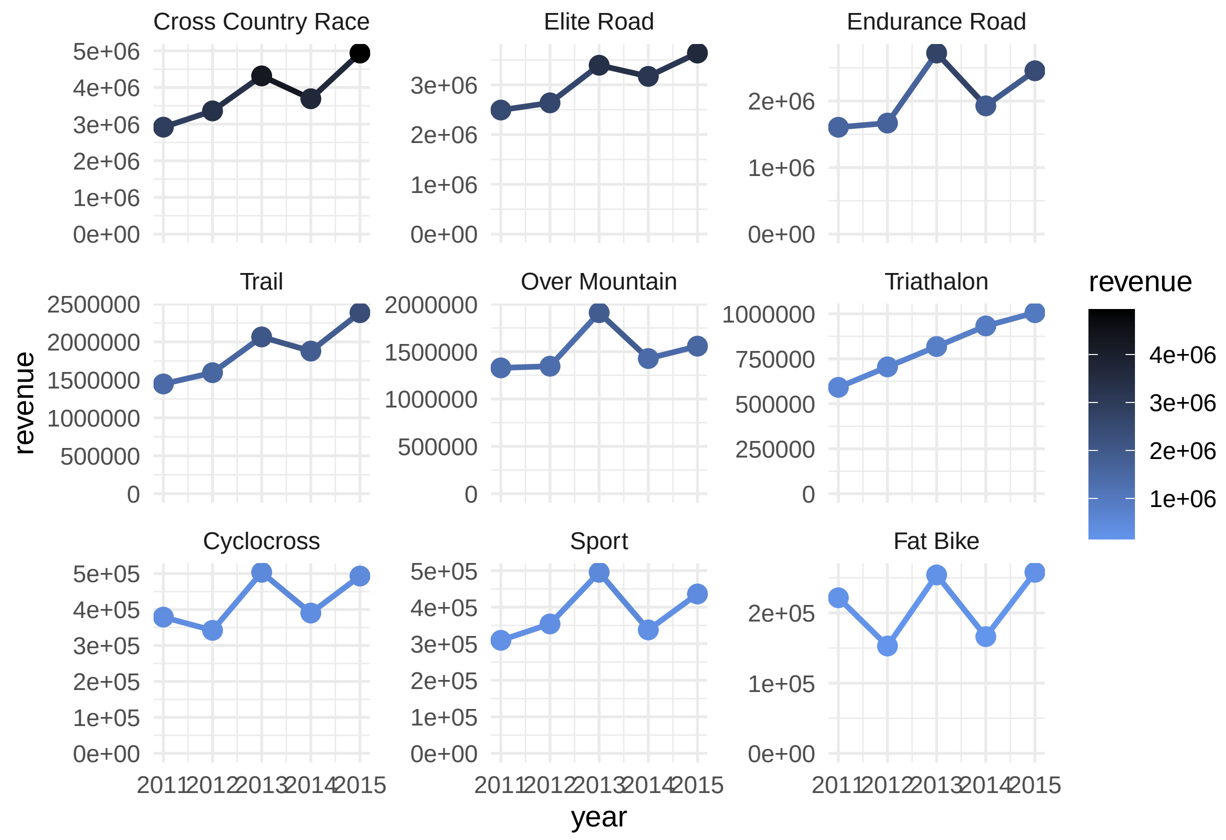


7.2 Scale Colours & Fills

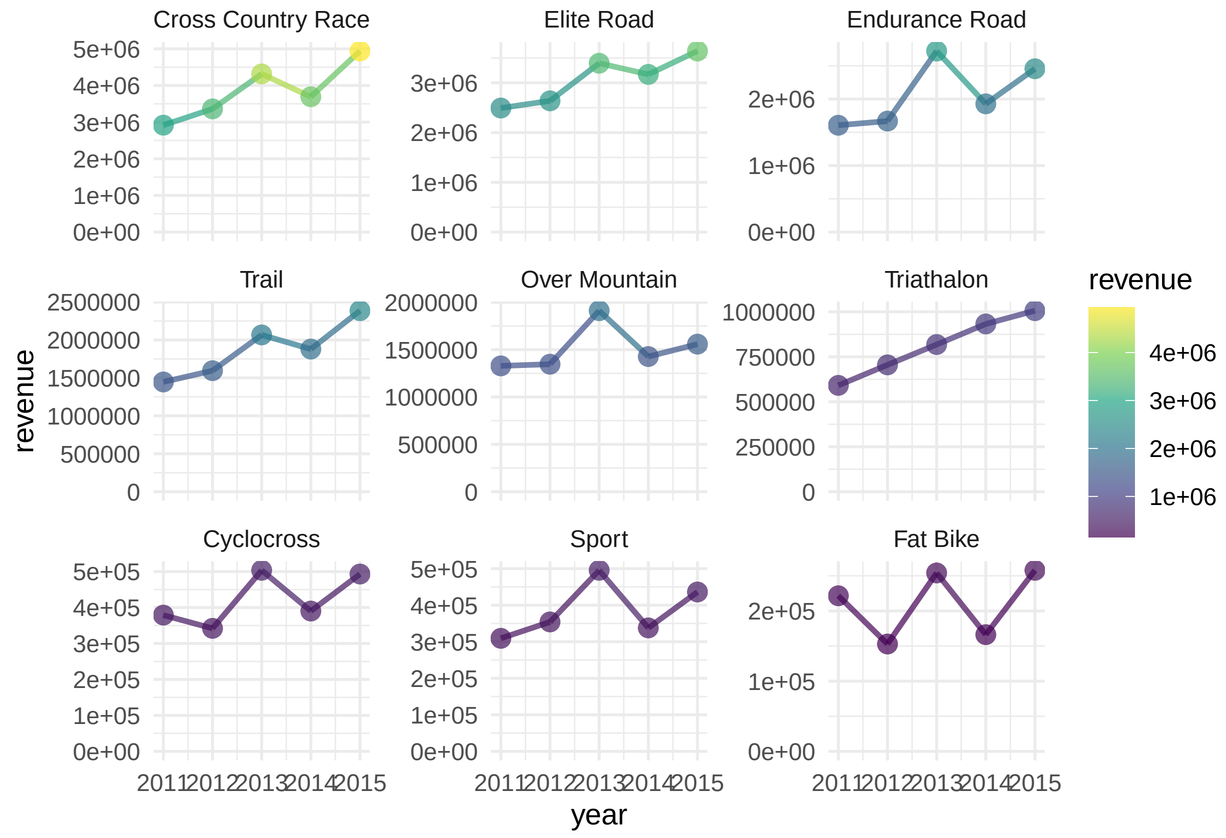
- Awesome way to show variation by groups (discrete) and by values (continuous)
- Because we have colour = category_2

colour by Revenue (continuous Scale): adjusting colour gradient

```
g_facet_continous +
    scale_color_continuous(
        low  = "cornflowerblue",
        high = "black"
    )
```

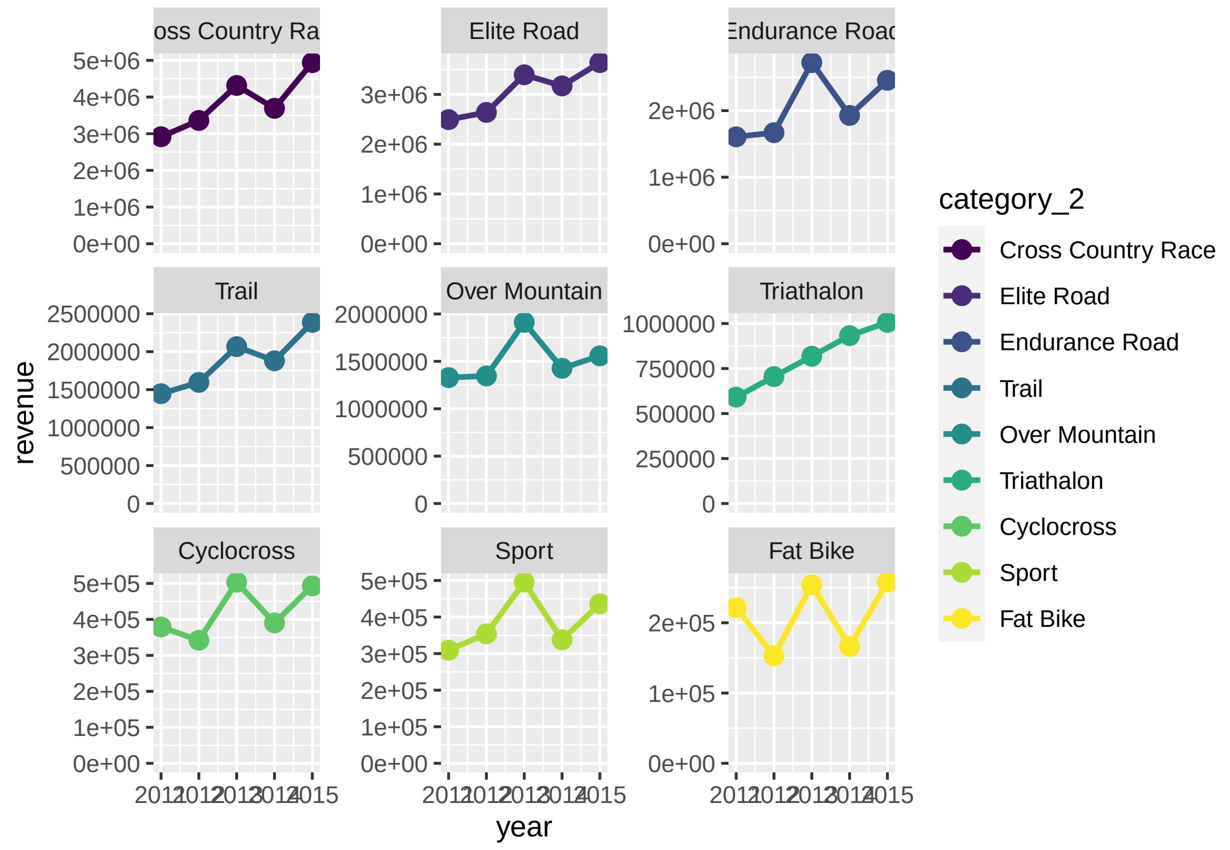


```
g_facet_continous +
    scale_color_viridis_c(alpha = 0.7)
```

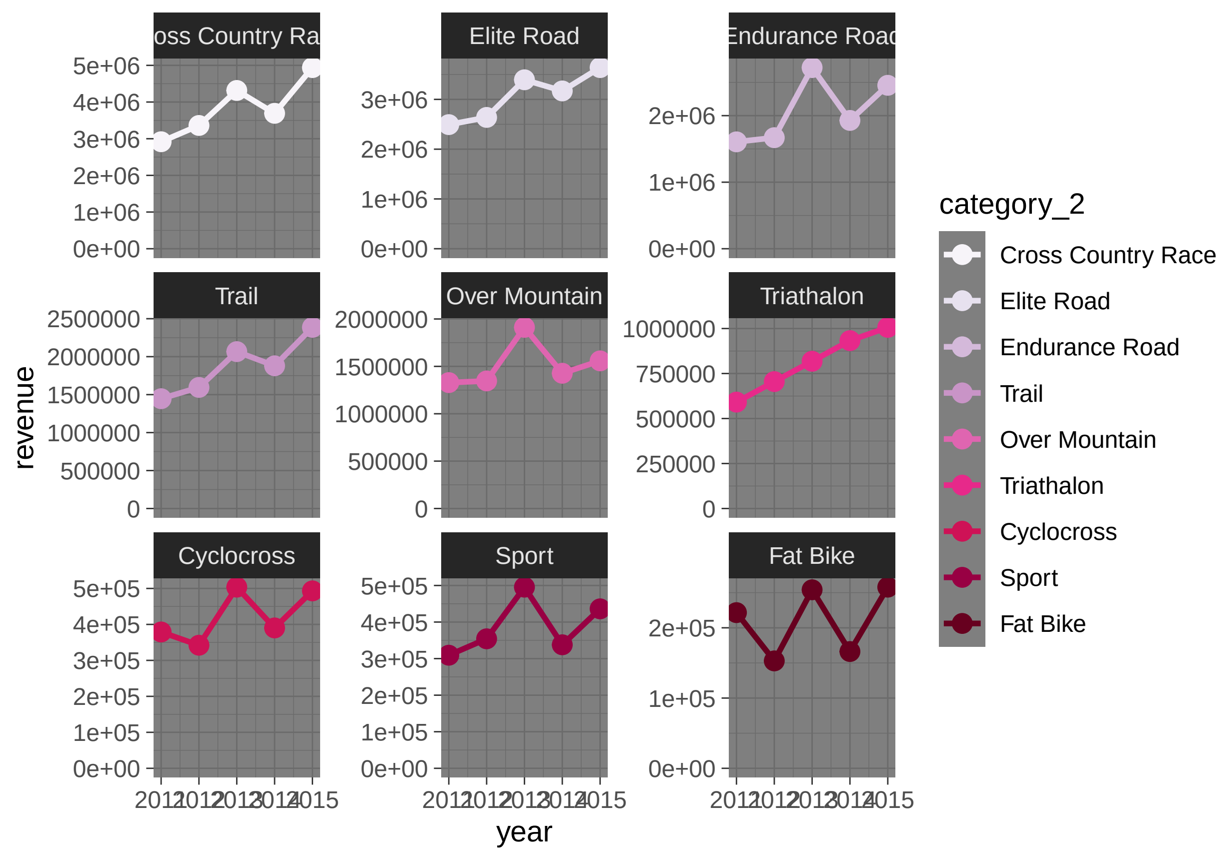


colour by Category 2: **discrete** Scale

```
g_facet_discrete +
    scale_color_viridis_d()
```



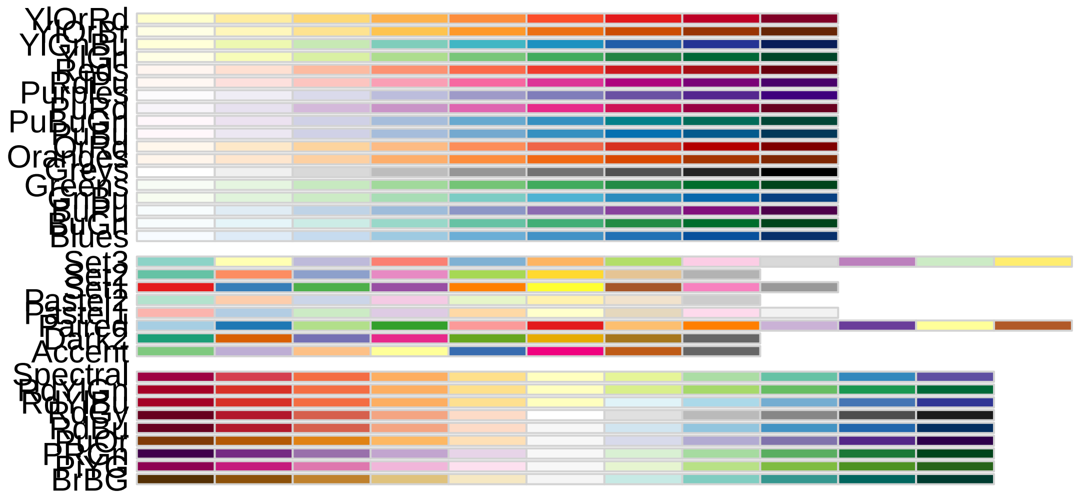
```
g_facet_discrete +
    scale_color_brewer(palette = "PuRd") +
    theme_dark()
```



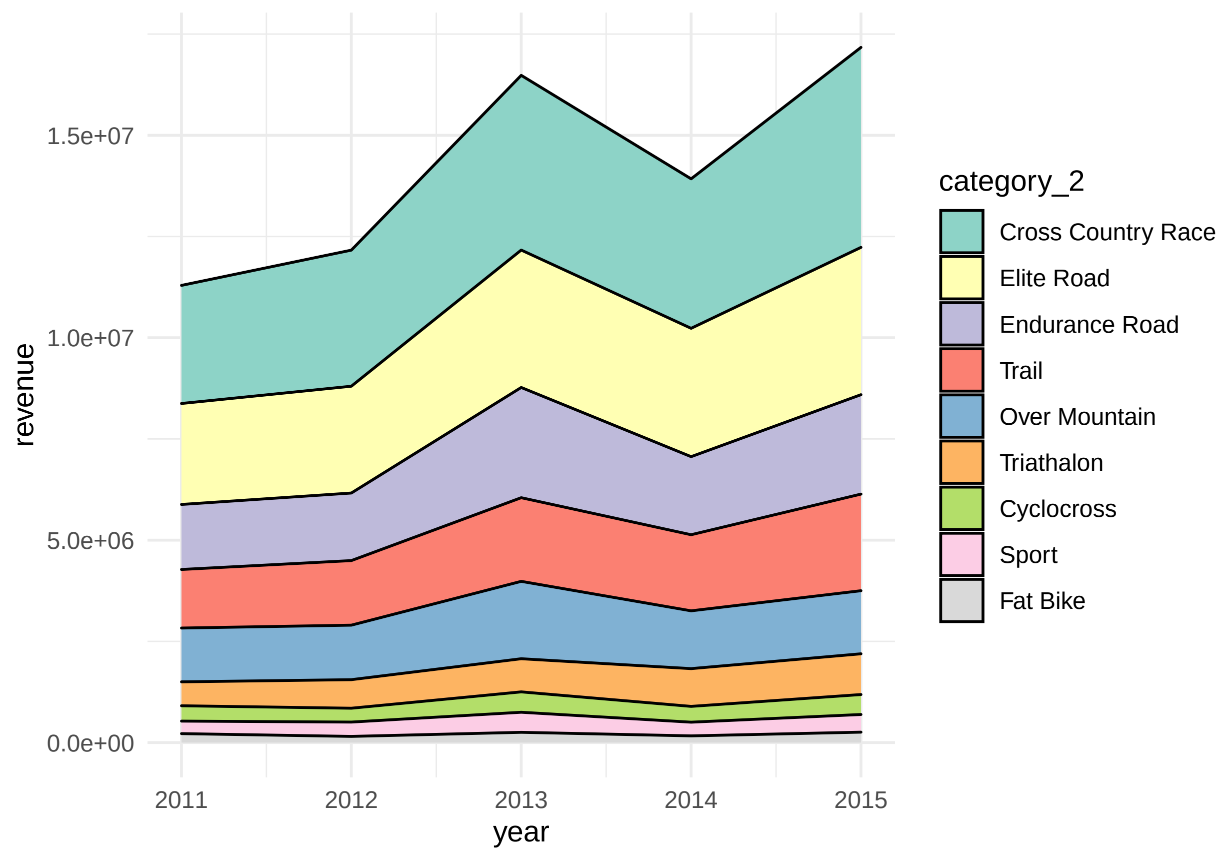
```
# RColorBrewer::display.brewer.all()
```

Fill by Category 2

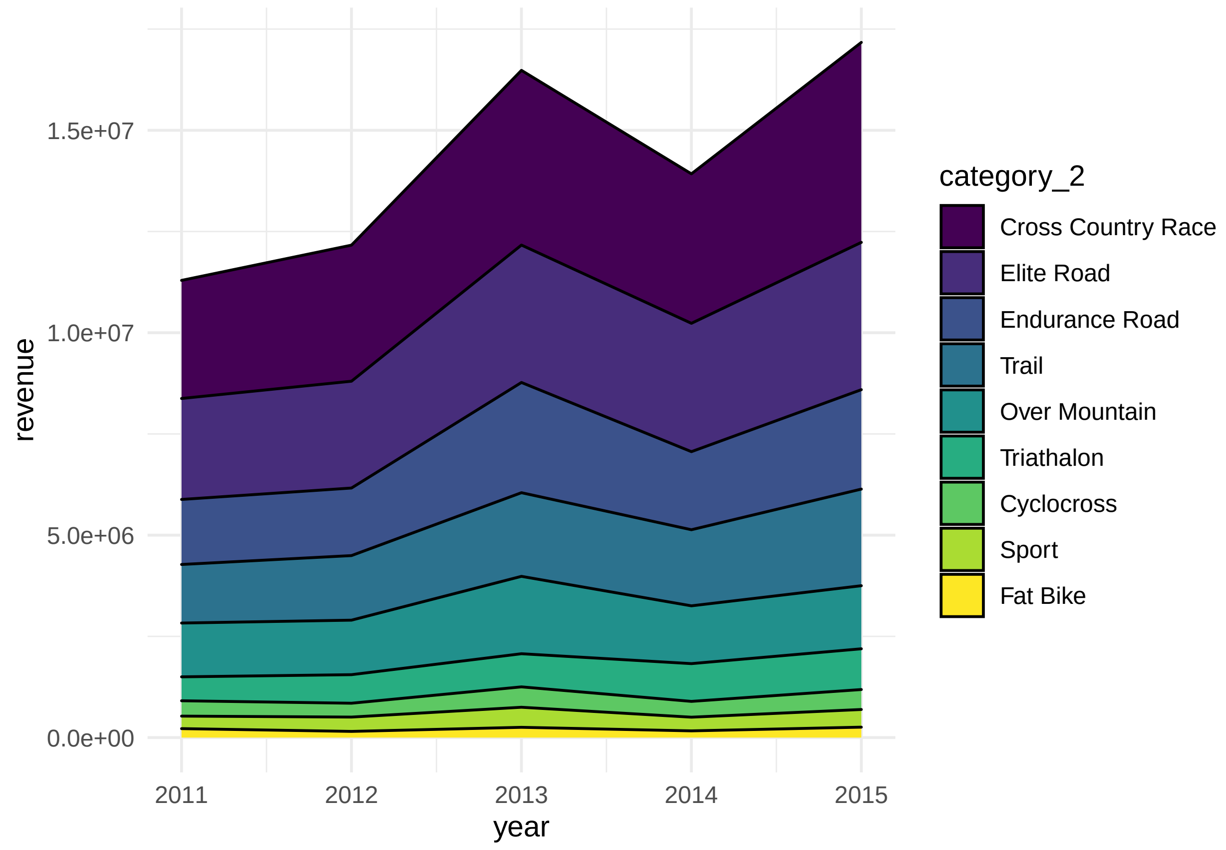
```
RColorBrewer::display.brewer.all()
```



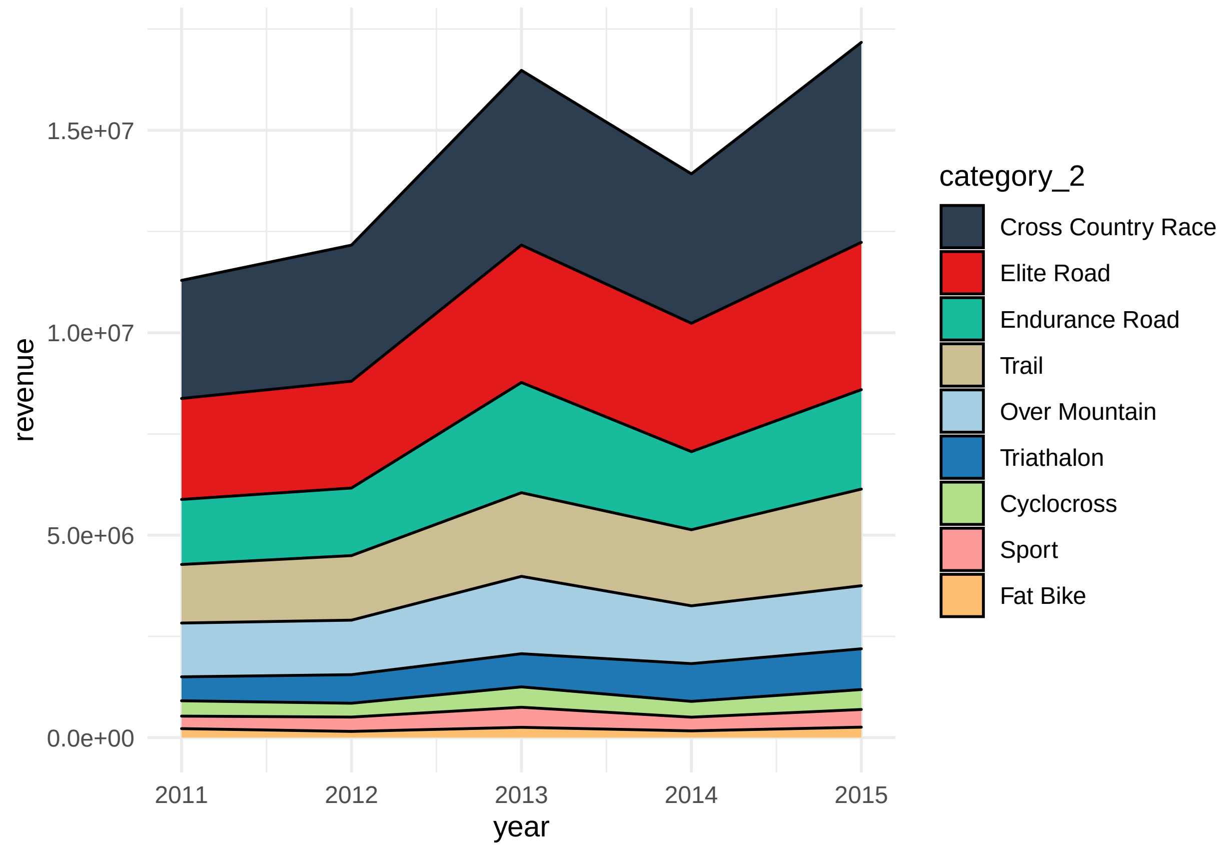
```r
g_area_discrete +
    scale_fill_brewer(palette = "Set3")
```



```
g_area_discrete +
    scale_fill_viridis_d()
```

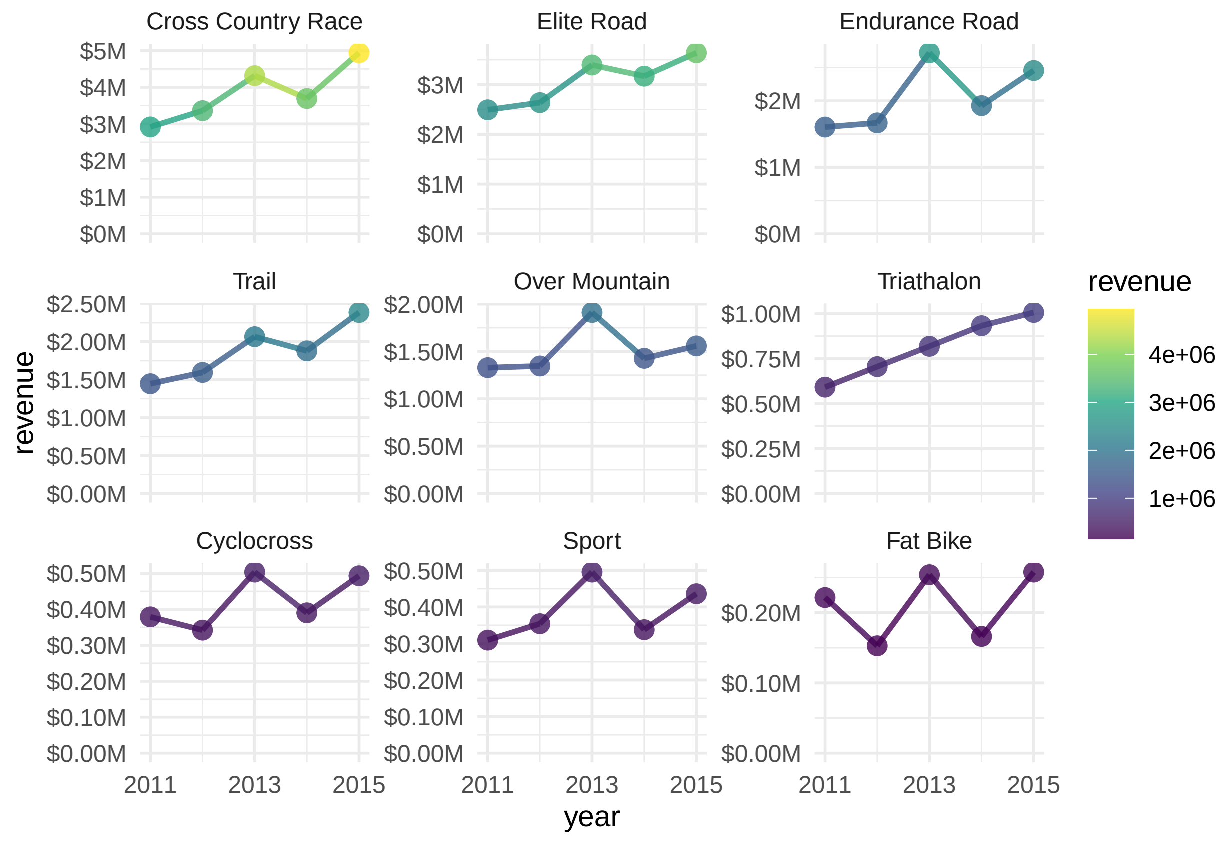


```
g_area_discrete +
    scale_fill_tq()
```



7.3 Axis Scales

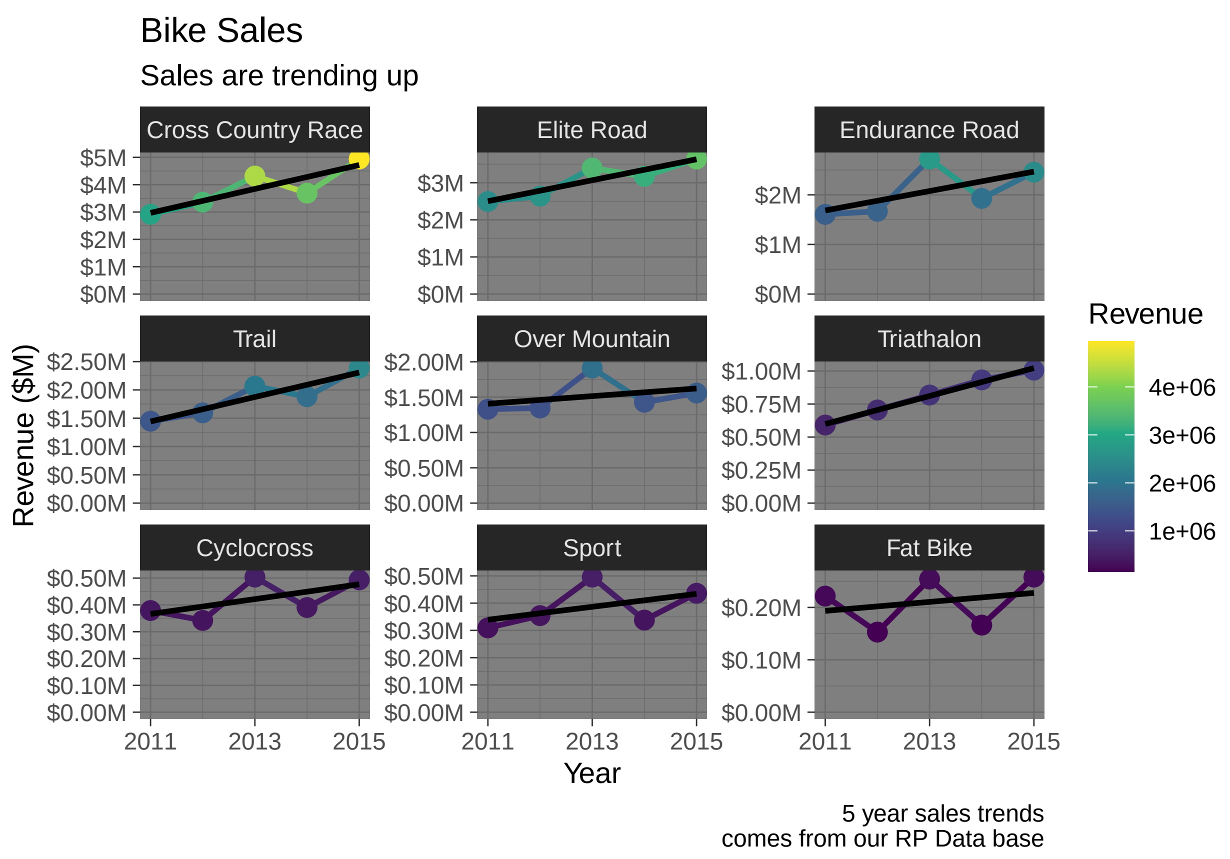
```
g_facet_continous +
    scale_color_viridis_c(alpha = 0.8) +
    # gives more room to breath on the x-axis
    scale_x_continuous(breaks = seq(2011, 2015, by = 2)) +
    scale_y_continuous(labels = scales::dollar_format(scale = 1e-6, suffix = "M")) +
    theme_minimal()
```



8.0 Labels

```
g_facet_continous +
    scale_x_continuous(breaks = seq(2011, 2015, by = 2)) +
    scale_y_continuous(labels = scales::dollar_format(scale = 1e-6, suffix = "M")) +
    geom_smooth(method = "lm", se = FALSE, color = "black") +

    scale_color_viridis_c() +
    theme_dark() +

    labs(
        title = "Bike Sales",
        subtitle = "Sales are trending up",
        caption = "5 year sales trends\ncomes from our RP Data base",
        x = "Year",
        y = "Revenue ($M)",
        colour = "Revenue"
    )
```

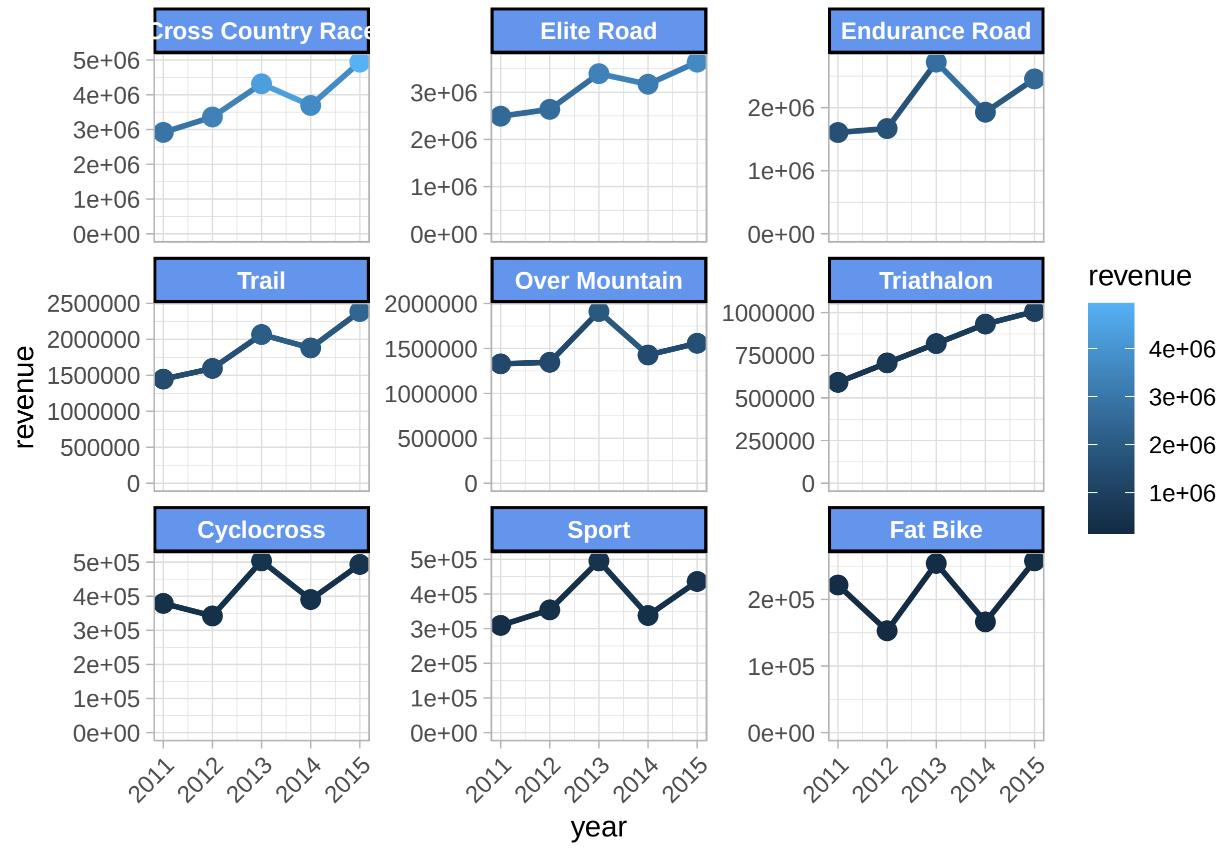
```
## `geom_smooth()` using formula 'y ~ x'
```



9.0 Labels

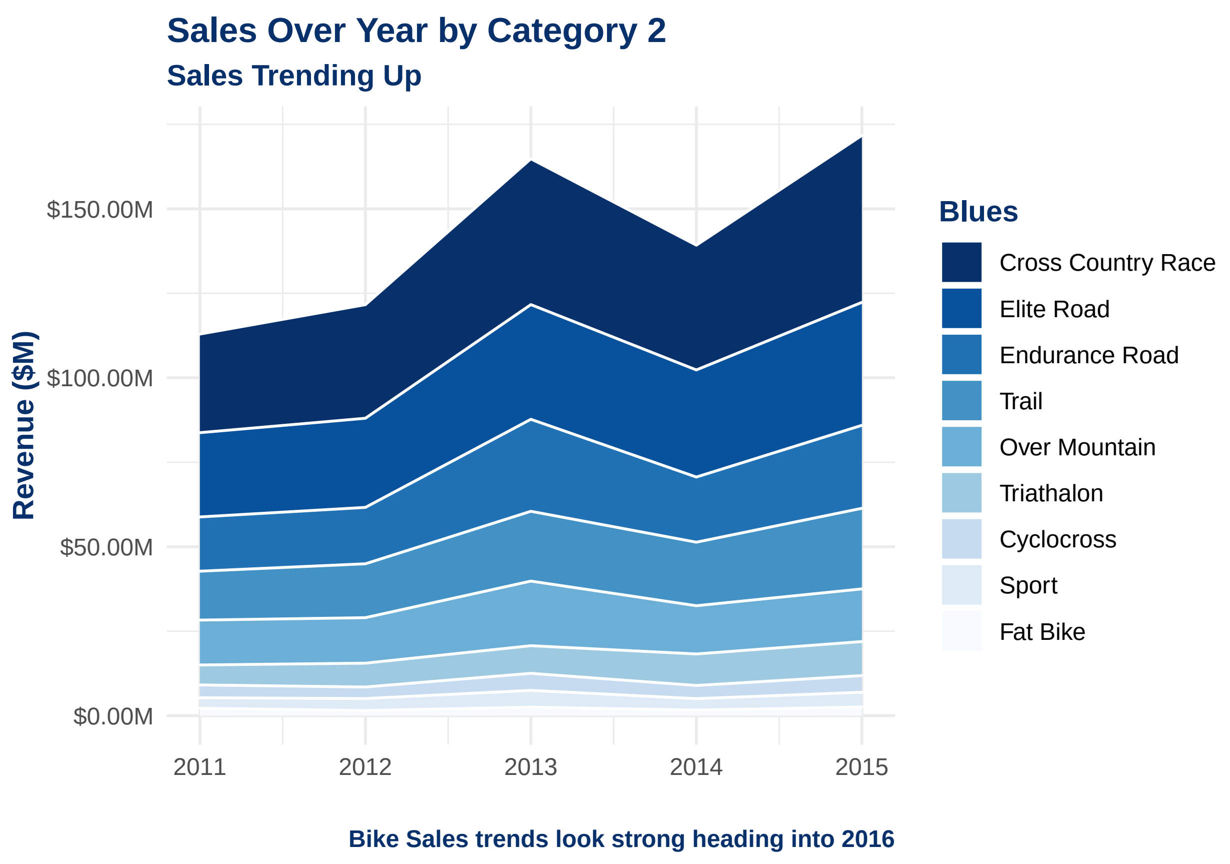
theme_light(): Pre-set theme theme(): The function used to adjust every theme element that is part of a ggplot object

```
g_facet_continous +
    theme_light() +
    theme(
        axis.text.x = element_text(
            angle = 45,
            hjust = 1
            ),
        strip.background = element_rect(
            colour = "black",
            fill   = "cornflowerblue",
            size   = 1
            ),
        strip.text = element_text(
            face = "bold",
            colour = "White"
        )
    )
```



10.0 Putting it All Together

```r
sales_by_year_category_2_tbl %>%
    ggplot(aes(year, revenue, fill = category_2)) +
    geom_area(colour = "white") +

    scale_fill_brewer("Blues", direction = -1) +
    scale_y_continuous(label = scales::dollar_format(scale = 10e-6, suffix = "M")) +

    labs(
        title = "Sales Over Year by Category 2",
        subtitle = "Sales Trending Up",
        caption = "Bike Sales trends look strong heading into 2016",
        x = "",
        y = "Revenue ($M)",
        fill = "2nd Category"
    ) +
    theme_minimal() +
    theme(
        title = element_text(face = "bold", colour = "#08306B")
    )
```


Sales Over Year by Category 2
Sales Trending Up
Revenue ($M)
$150.00M
$100.00M
$50.00M
$0.00M
2011
2012
2013
2014
2015
Blues
Cross Country Race
Elite Road
Endurance Road
Trail
Over Mountain
Triathalon
Cyclocross
Sport
Fat Bike
Bike Sales trends look strong heading into 2016