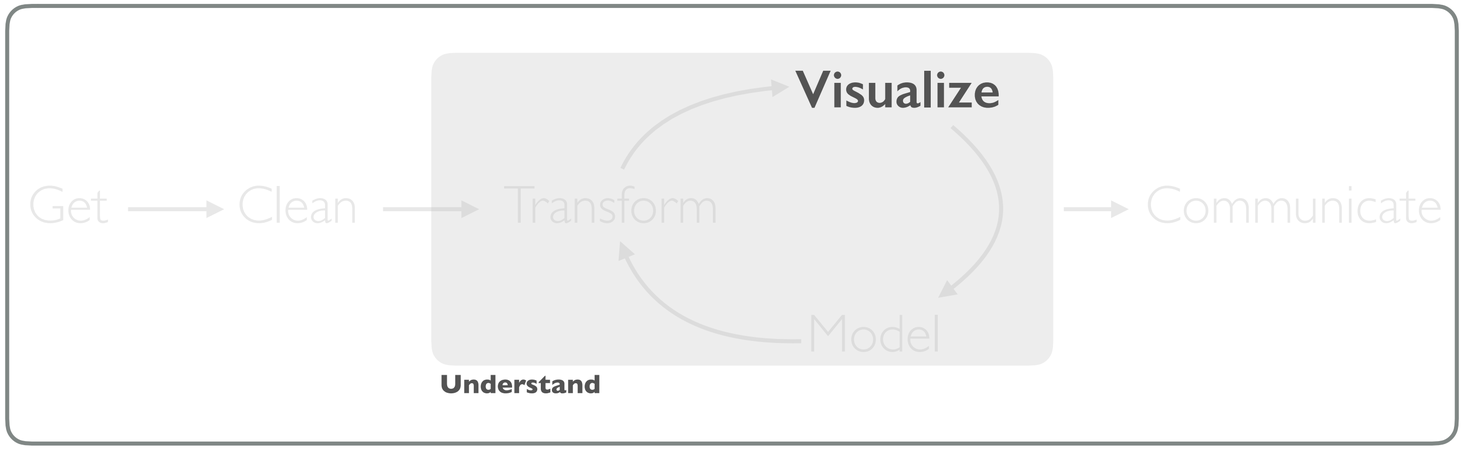
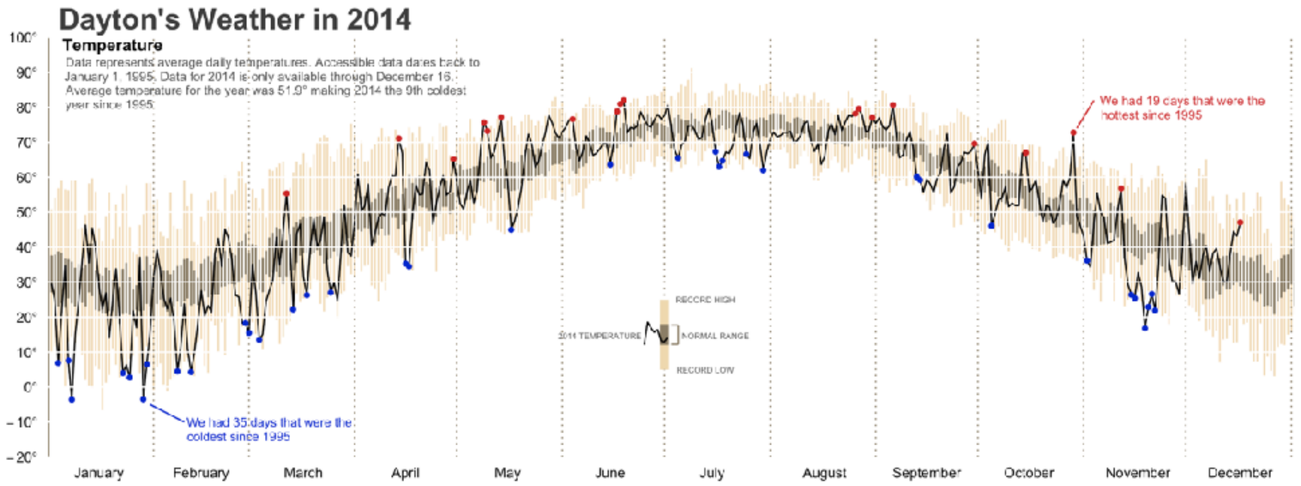
VISUALIZATION

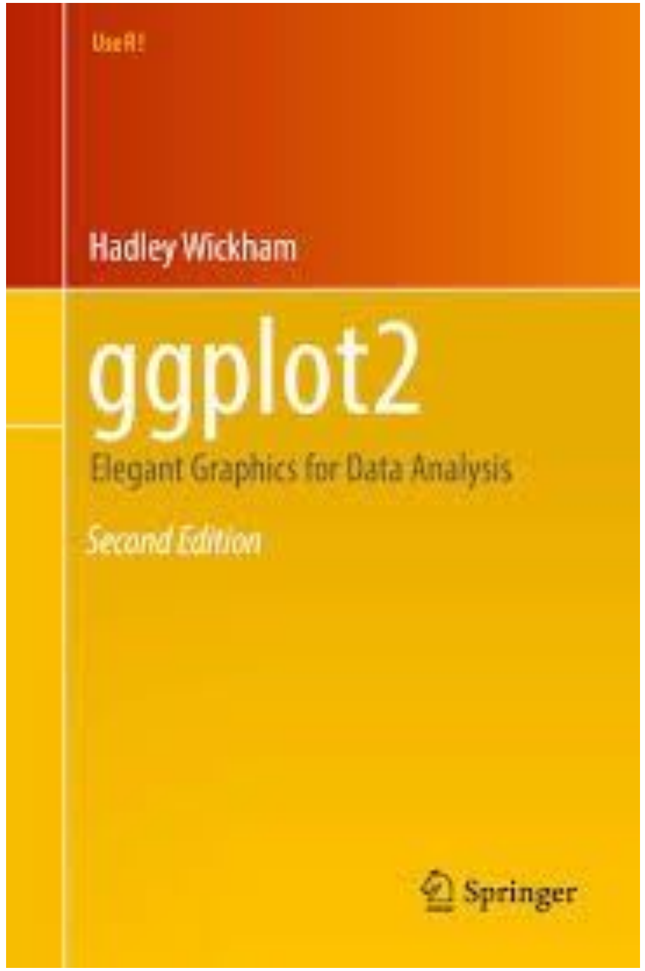


†A modified version of Hadley Wickham's analytic process

ggplot2

- R has several systems for making graphs
- `ggplot2` is the most elegant and versatile
- Implements the grammar of graphics theory behind data visualization





PREREQUISITES

PACKAGE PREREQUISITE

```
library(tidyverse)
#> Loading tidyverse: ggplot2
#> Loading tidyverse: tibble
#> Loading tidyverse: tidyr
#> Loading tidyverse: readr
#> Loading tidyverse: purrr
#> Loading tidyverse: dplyr
#> Conflicts with tidy packages ----------------------------------------
#> filter(): dplyr, stats
#> lag():    dplyr, stats
```

DATA PREREQUISITE

```
mpg
#> # A tibble: 234 × 11
#>   manufacturer model displ  year   cyl      trans   drv   cty   hwy    fl
#>          <chr> <chr> <dbl> <int> <int>      <chr> <chr> <int> <int> <chr>
#> 1         audi    a4   1.8  1999     4   auto(l5)     f    18    29     p
#> 2         audi    a4   1.8  1999     4 manual(m5)     f    21    29     p
#> 3         audi    a4   2.0  2008     4 manual(m6)     f    20    31     p
#> 4         audi    a4   2.0  2008     4   auto(av)     f    21    30     p
#> 5         audi    a4   2.8  1999     6   auto(l5)     f    16    26     p
#> 6         audi    a4   2.8  1999     6 manual(m5)     f    18    26     p
#> # ... with 228 more rows, and 1 more variables: class <chr>
```

*Type **`View(mpg)`** in the console for a spreadsheet view of the data*

YOUR TURN!

Can you find the help file explaining the **`mpg`** *data set?*

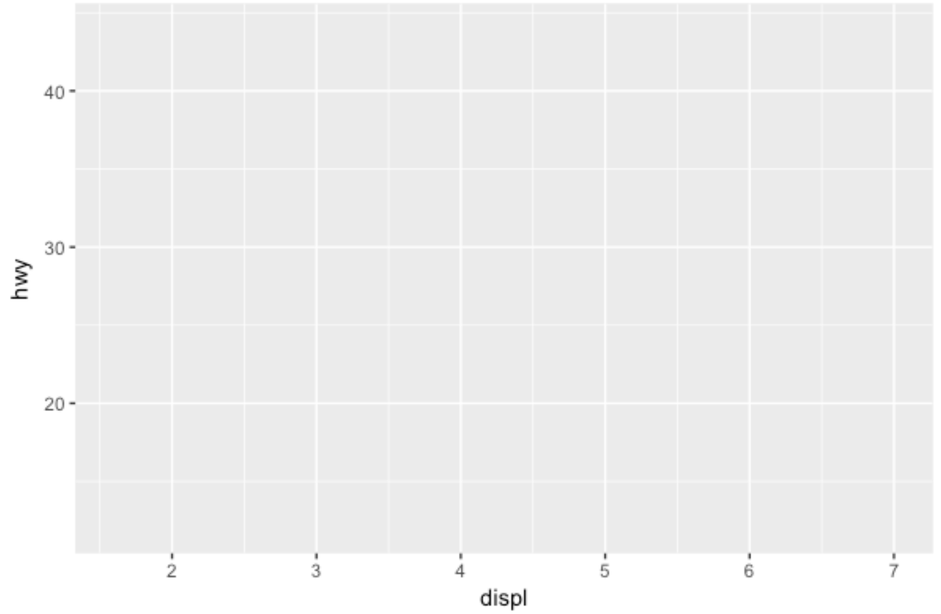
SOLUTION

```
# additional documentation for the mpg data set
?mpg
```

CANVAS

LET'S CREATE OUR "CANVAS"

```
# left
ggplot(data = mpg)

# right
ggplot(data = mpg, aes(x = displ, y = hwy))
```

GEOMS

```
geom_abline       geom_histogram
geom_bar          geom_jitter
geom_bin2d        geom_label
geom_blank        geom_map
geom_boxplot      geom_path
geom_contour      geom_point
geom_count        geom_polygon
geom_hex          geom_quantile
geom_crossbar     geom_raster
geom_density      geom_ribbon
geom_density_2d   geom_rug
geom_dotplot      geom_segment
geom_errorharh    geom_smooth
geom_freqpoly     geom_violin
```

LETS ADD "GEOMS"

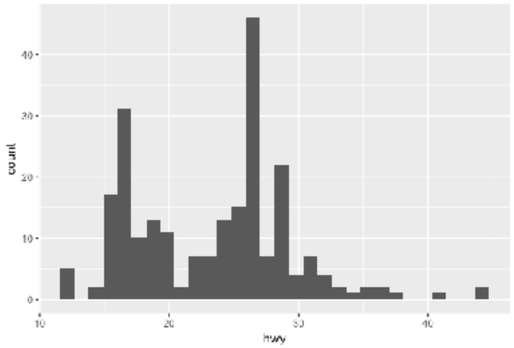
- We display data with geometric shapes
- ~ 30 built-in geoms (with many more offered by other pkgs)

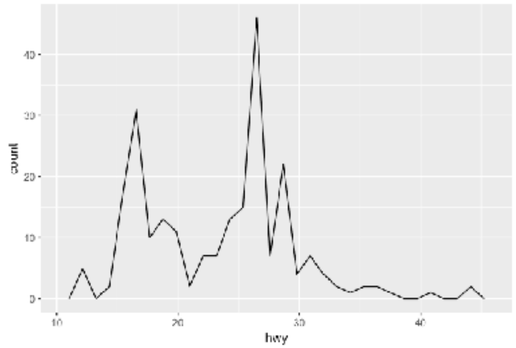
Type **`geom_ +`** *tab in the console*

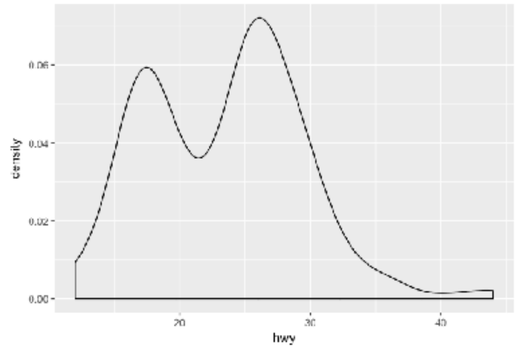
```
geom_abline      geom_histogram
geom_bar         geom_jitter
geom_bin2d       geom_label
geom_blank       geom_map
geom_boxplot     geom_path
geom_contour     geom_point
geom_count       geom_polygon
geom_hex         geom_quantile
geom_crossbar    geom_raster
geom_density     geom_ribbon
geom_density_2d  geom_rug
geom_dotplot     geom_segment
geom_errorharh   geom_smooth
geom_freqpoly    geom_violin
```

UNIVARIATE GEOMS

```
ggplot(data = mpg, aes(x = hwy)) +
  geom_histogram()

ggplot(data = mpg, aes(x = hwy)) +
  geom_freqpoly()

ggplot(data = mpg, aes(x = hwy)) +
  geom_density()
```



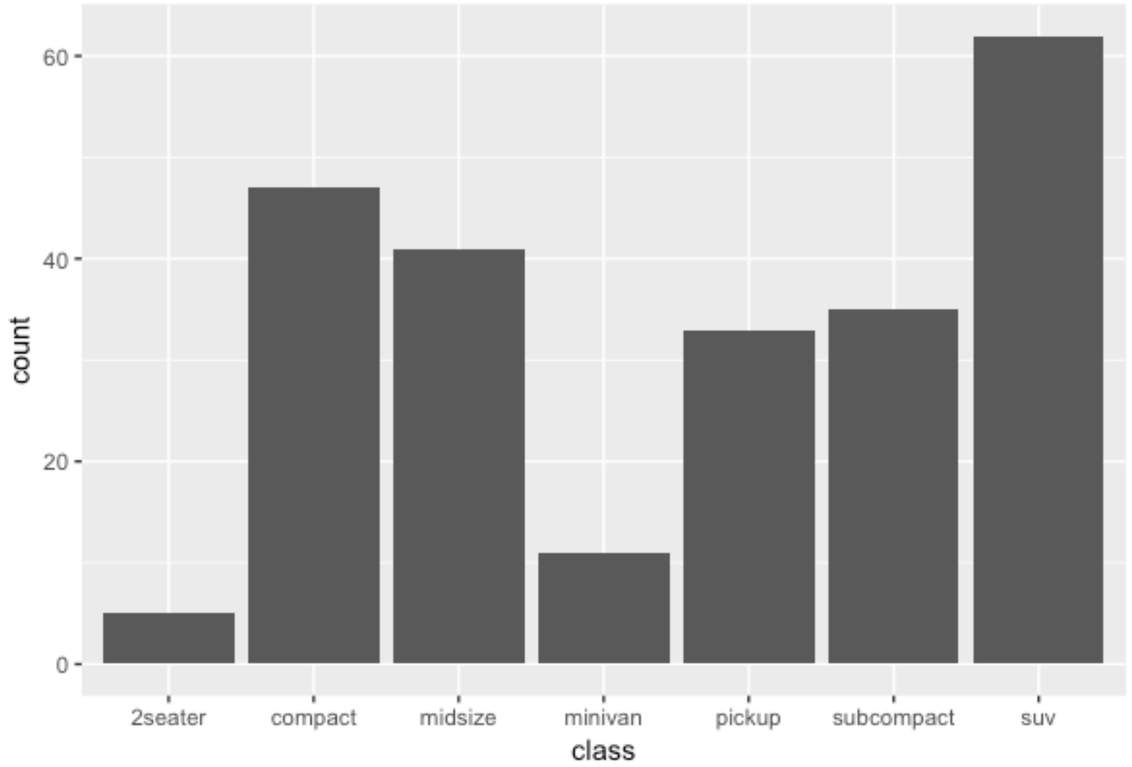




geom_abline
geom_bar
geom_bin2d
geom_blank
geom_boxplot
geom_contour
geom_count
geom_hex
geom_crossbar
geom_density
geom_density_2d
geom_dotplot
geom_errorharh
geom_freqpoly
geom_histogram
geom_jitter
geom_label
geom_map
geom_path
geom_point
geom_polygon
geom_quantile
geom_raster
geom_ribbon
geom_rug
geom_segment
geom_smooth
geom_violin

UNIVARIATE GEOMS

```
ggplot(data = mpg, aes(x = class)) +
  geom_bar()
```



geom_abline
geom_bar
geom_bin2d
geom_blank
geom_boxplot
geom_contour
geom_count
geom_hex
geom_crossbar
geom_density
geom_density_2d
geom_dotplot
geom_errorharh
geom_freqpoly
geom_histogram
geom_jitter
geom_label
geom_map
geom_path
geom_point
geom_polygon
geom_quantile
geom_raster
geom_ribbon
geom_rug
geom_segment
geom_smooth
geom_violin

UNIVARIATE GEOMS

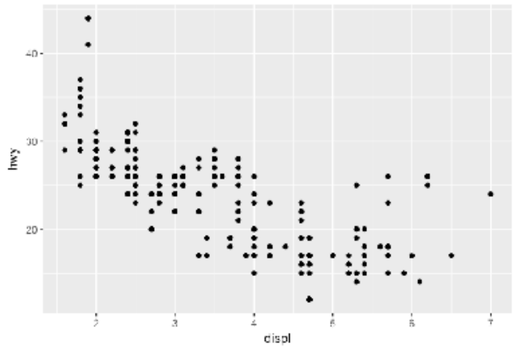
```
ggplot(data = mpg, aes(x = class)) +
  geom_bar()
```

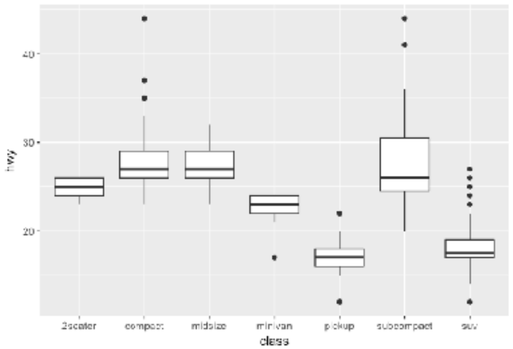
- This is called an **aes**thetic mapping argument
- Every geom requires a mapping argument
  - Some geoms require just one (x variable)
  - While other geoms require two (x & y variable)

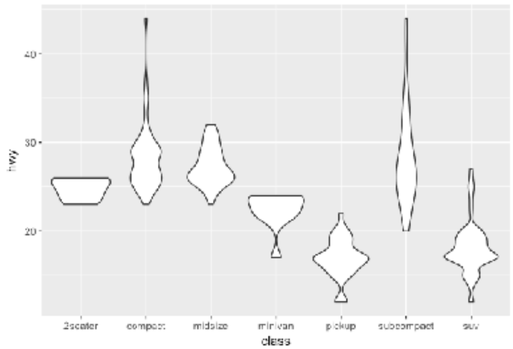
```
geom_abline      geom_histogram
geom_bar         geom_jitter
geom_bin2d       geom_label
geom_blank       geom_map
geom_boxplot     geom_path
geom_contour     geom_point
geom_count       geom_polygon
geom_hex         geom_quantile
geom_crossbar    geom_raster
geom_density     geom_ribbon
geom_density_2d  geom_rug
geom_dotplot     geom_segment
geom_errorharh   geom_smooth
geom_freqpoly    geom_violin
```

BIVARIATE GEOMS

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point()

ggplot(data = mpg, aes(x = class, y = hwy)) +
  geom_boxplot()

ggplot(data = mpg, aes(x = class, y = hwy)) +
  geom_violin()
```







geom_abline
geom_bar
geom_bin2d
geom_blank
geom_boxplot
geom_contour
geom_count
geom_hex
geom_crossbar
geom_density
geom_density_2d
geom_dotplot
geom_errorharh
geom_freqpoly
geom_histogram
geom_jitter
geom_label
geom_map
geom_path
geom_point
geom_polygon
geom_quantile
geom_raster
geom_ribbon
geom_rug
geom_segment
geom_smooth
geom_violin

YOUR TURN!

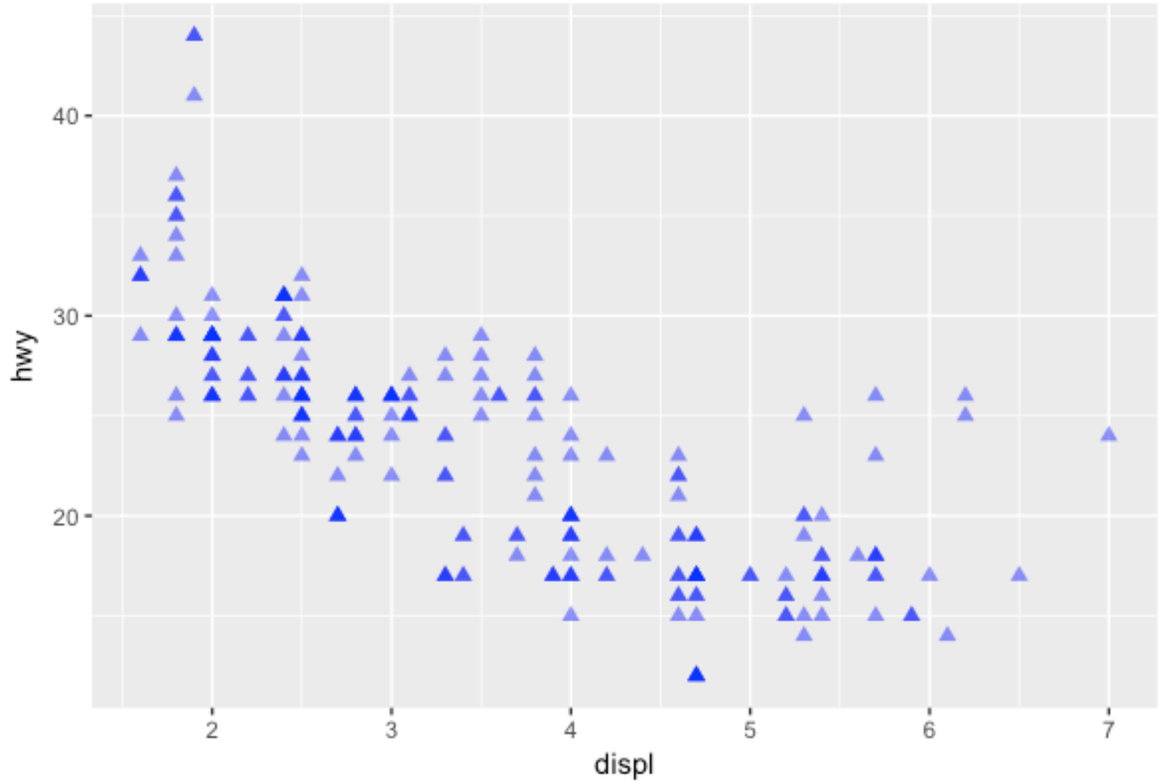
1. Import the CustomerData.csv file
2. Create a chart that illustrates the distribution of the `DebtToIncomeRatio` variable.
3. Create a chart that shows the counts for each `JobCategory`
4. Create a scatter plot of `HHIncome` vs `CardSpendMonth`

SOLUTION

```
#1: import data
customer <- read_csv("data/CustomerData.csv")

#2: distribution of DebtToIncomeRatio variable
ggplot(data = customer, aes(x = DebtToIncomeRatio)) +
  geom_histogram()

#3: distribution of JobCategory variable
ggplot(data = customer, aes(x = JobCategory)) +
  geom_bar()

#4: scatter plot for HHIncome vs CardSpendMonth
ggplot(data = customer, aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point()
```

NON-MAPPING AESTHETICS

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point(color = "blue", size = 2, shape = 17, alpha = .5)
```



We can also change other visual aesthetics in our graphics

- color
- size
- sh△pe (0-25 **?pch**)
- opacity

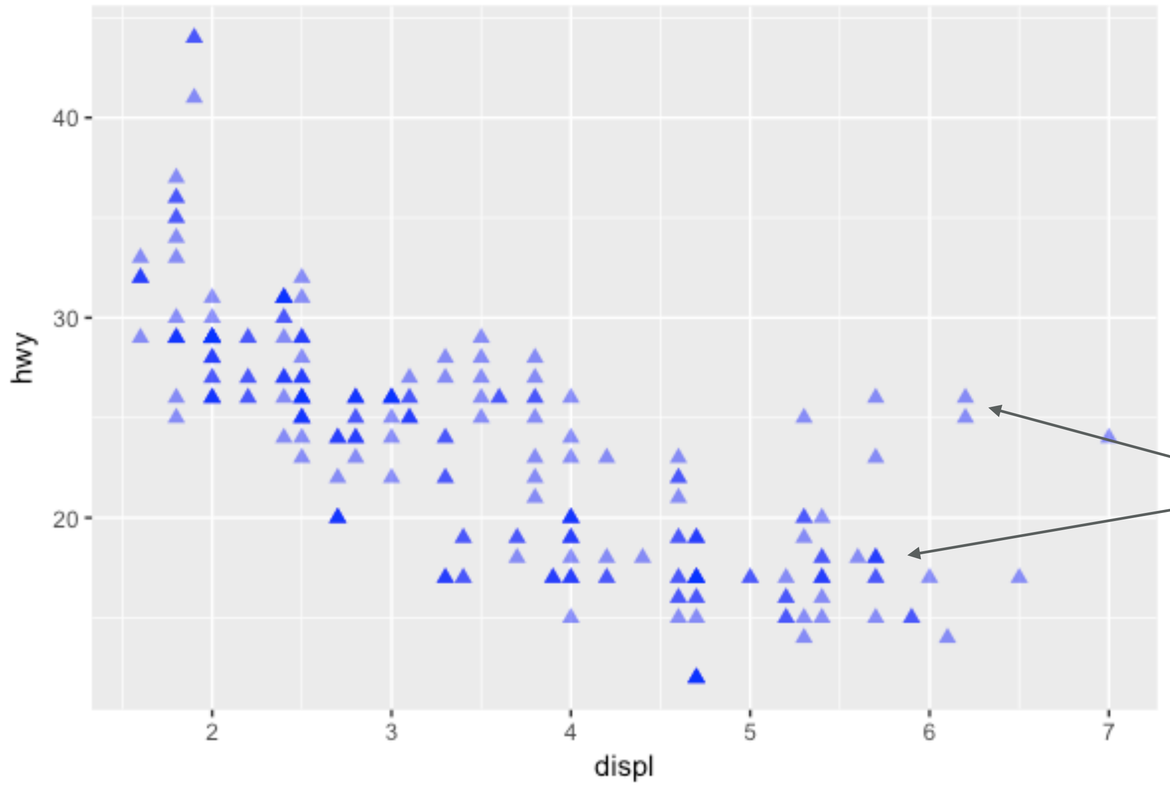
Lots of color and sh◆pe options; just google and you'll find plenty of references

NON-MAPPING AESTHETICS

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point(color = "blue", size = 2, shape = 17, alpha = .5)
```

We can also change other visual aesthetics in our graphics

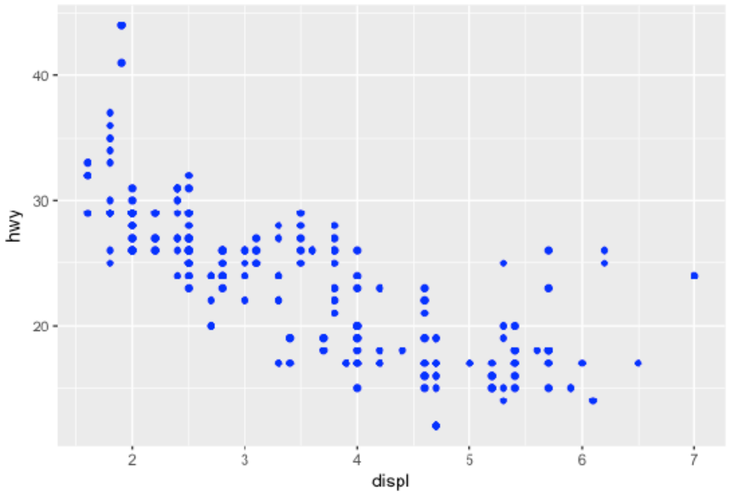
- color
- size
- sh△pe
- opacity



- Why are some points darker than others?
- Try **`geom_jitter`** in place of **`geom_point`**
- What do you think **`geom_jitter`** does?

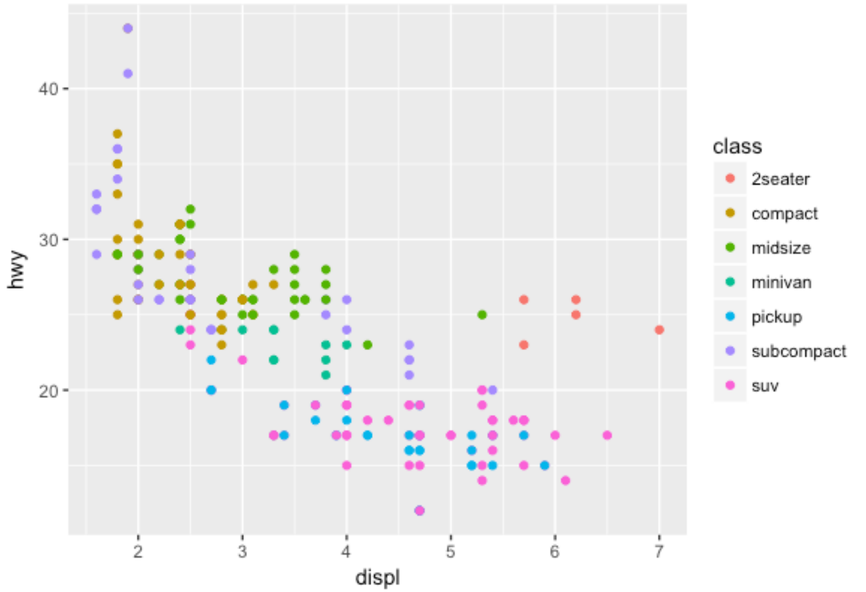
ADDING A 3RD DIMENSION

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point(color = "blue")
```



By moving the **`color`** argument to within `aes()`, we can map a 3rd variable to our plot

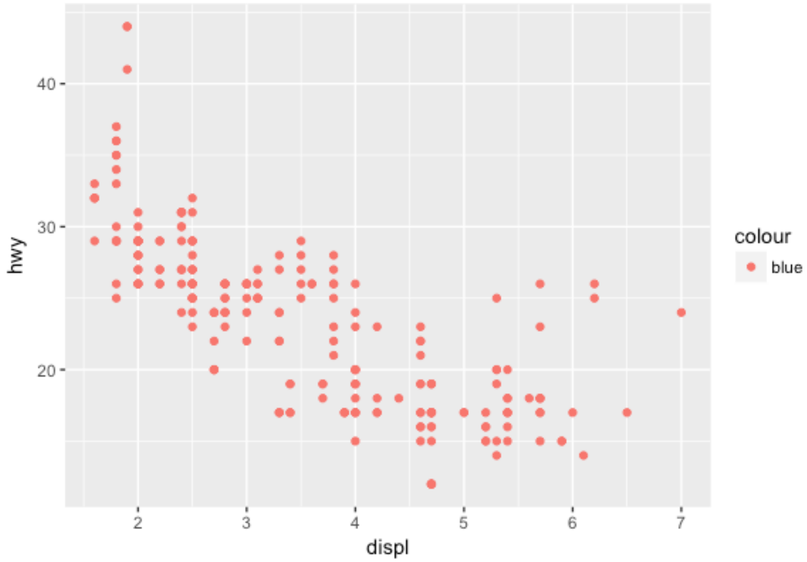
```
ggplot(data = mpg, aes(x = displ, y = hwy, color = class)) +
  geom_point()
```



ADDING A 3RD DIMENSION

A common error…what happened???

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = “blue”)) +
  geom_point()
```



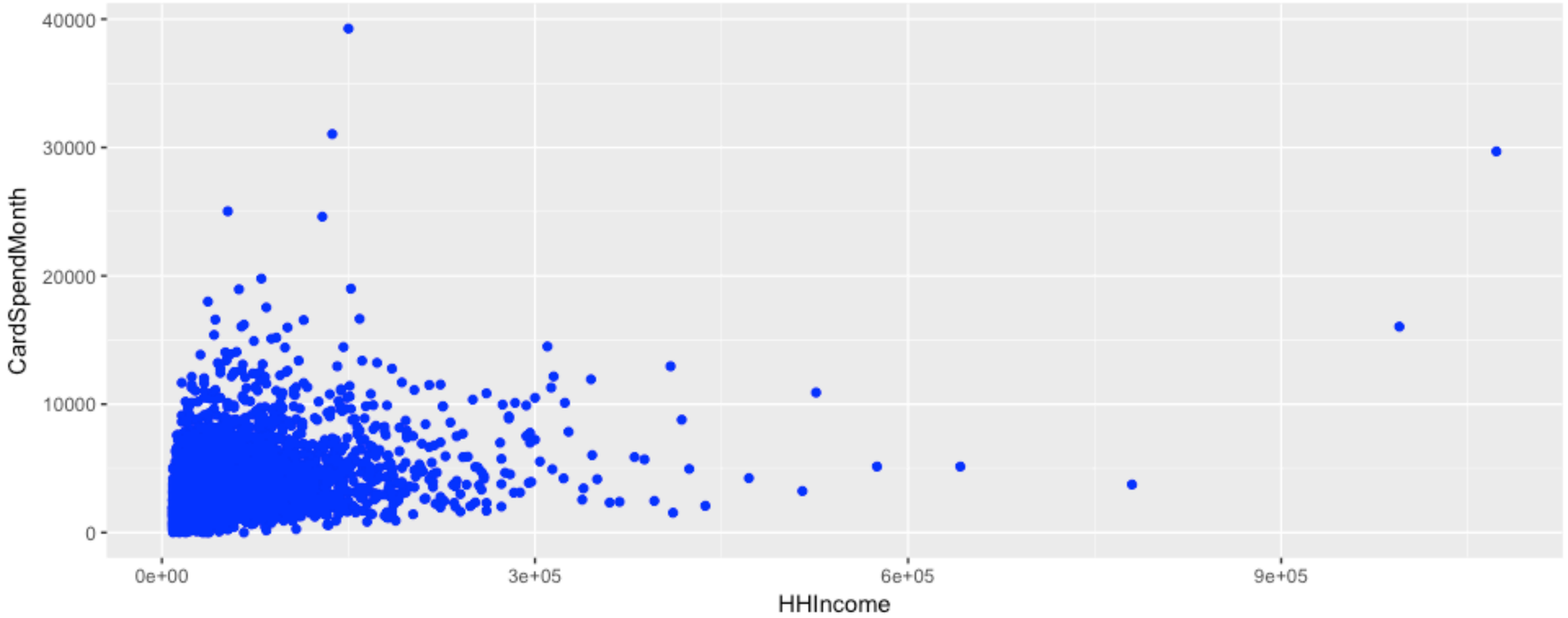
YOUR TURN!

1. Create a scatter plot of **`HHIncome`** vs **`CardSpendMonth`** and color all points blue.

2. Create a scatter plot of **`HHIncome`** vs **`CardSpendMonth`** and color all points based on whether or not the customer is retired.

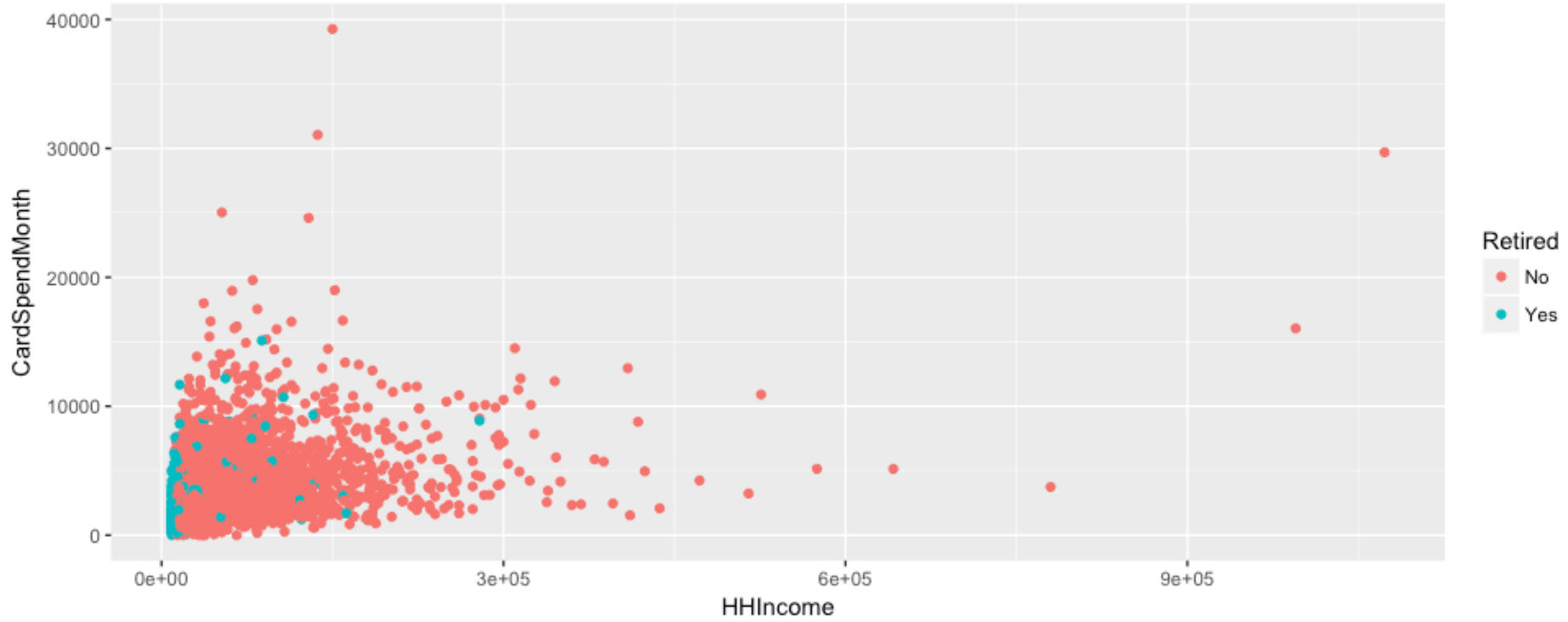
SOLUTION

```
#1. Create a scatter plot of HHIncome vs CardSpendMonth and color all points blue.
ggplot(customer, aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point(color = "blue")
```



SOLUTION

```
#2. Create a scatter plot of HHIncome vs CardSpendMonth and color all points based on
#   whether or not the customer is retired..
ggplot(customer, aes(x = HHIncome, y = CardSpendMonth, color = Retired)) +
  geom_point()
```

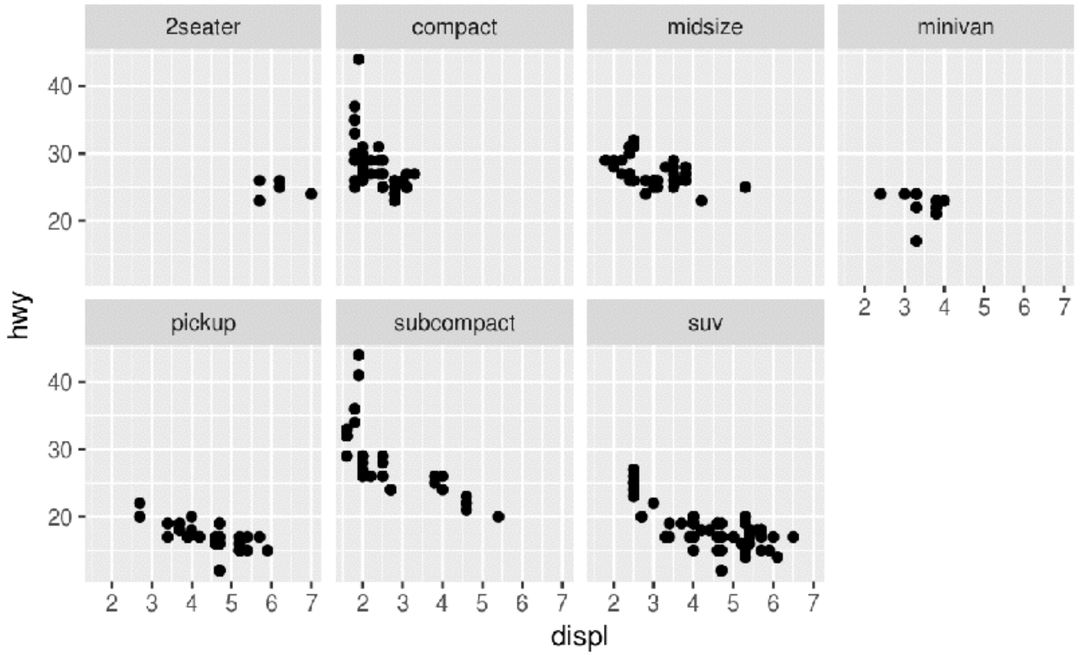


FACETS

FACETS = SMALL MULTIPLES

- The `facet` functions provide a simple way to create small multiples
  - `facet_wrap`: primarily used to create small multiples based on a single variable
  - `facet_grid`: primarily used to create a small multiples grid based on two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  facet_wrap(~ class, nrow = 2)
```

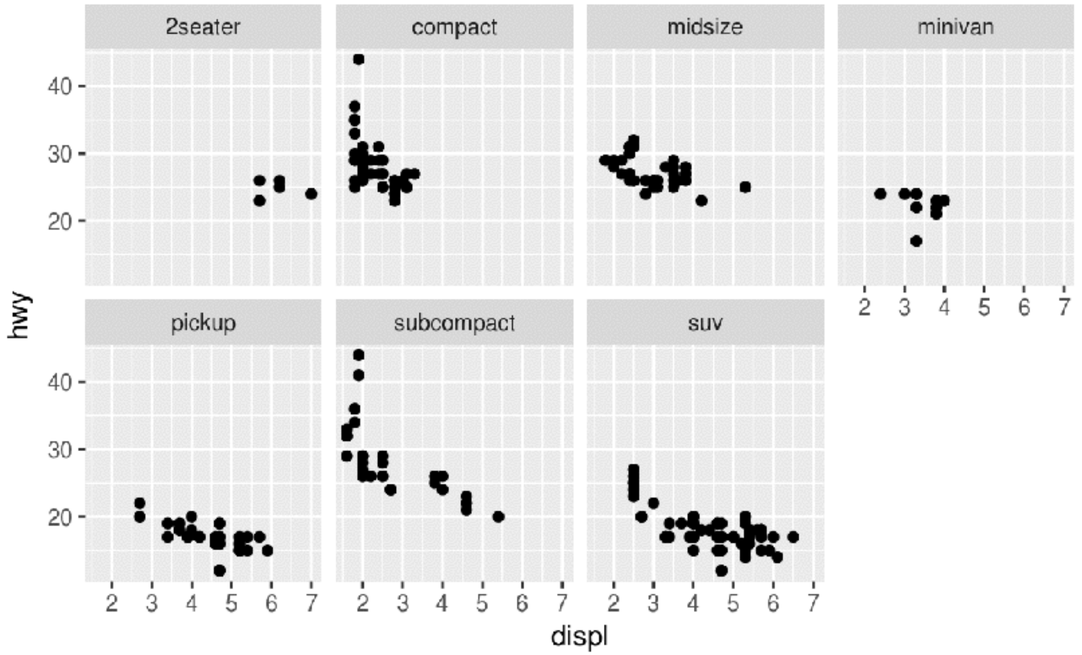


FACETS = SMALL MULTIPLES

- The `facet` functions provide a simple way to create small multiples
  - `facet_wrap`: primarily used to create small multiples based on a single variable
  - `facet_grid`: primarily used to create a small multiples grid based on two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  facet_wrap(~ class, nrow = 2)
```

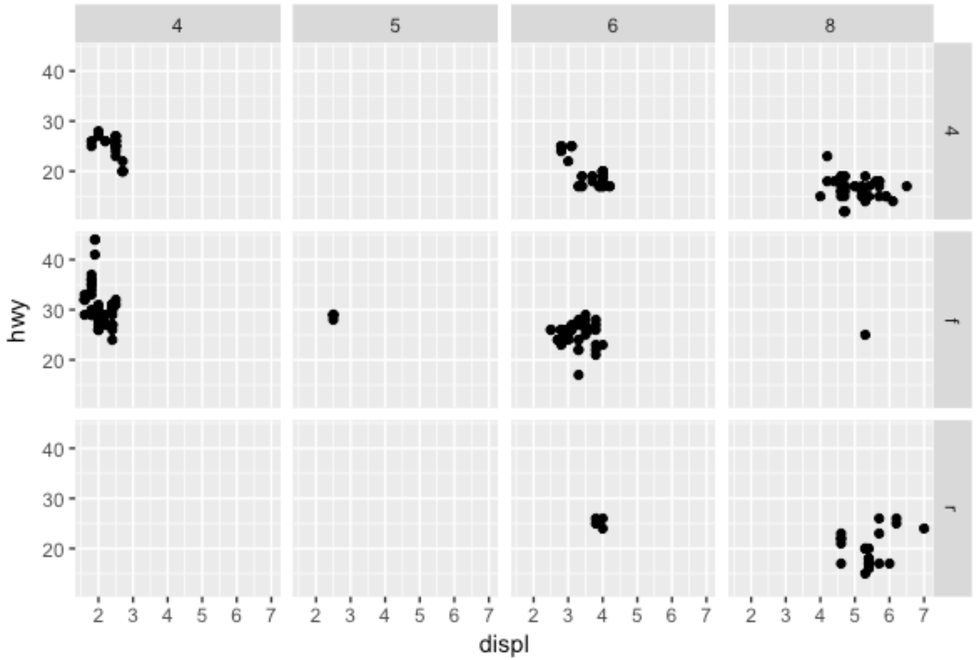
- use `nrow` or `ncol` to specify dimensions
- `?facet_wrap` to see other arguments to control the output



FACETS = SMALL MULTIPLES

- The `facet` functions provide a simple way to create small multiples
  - `facet_wrap`: primarily used to create small multiples based on a single variable
  - `facet_grid`: primarily used to create a small multiples grid based on two variables

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  facet_grid(drv ~ cyl)
```



YOUR TURN!

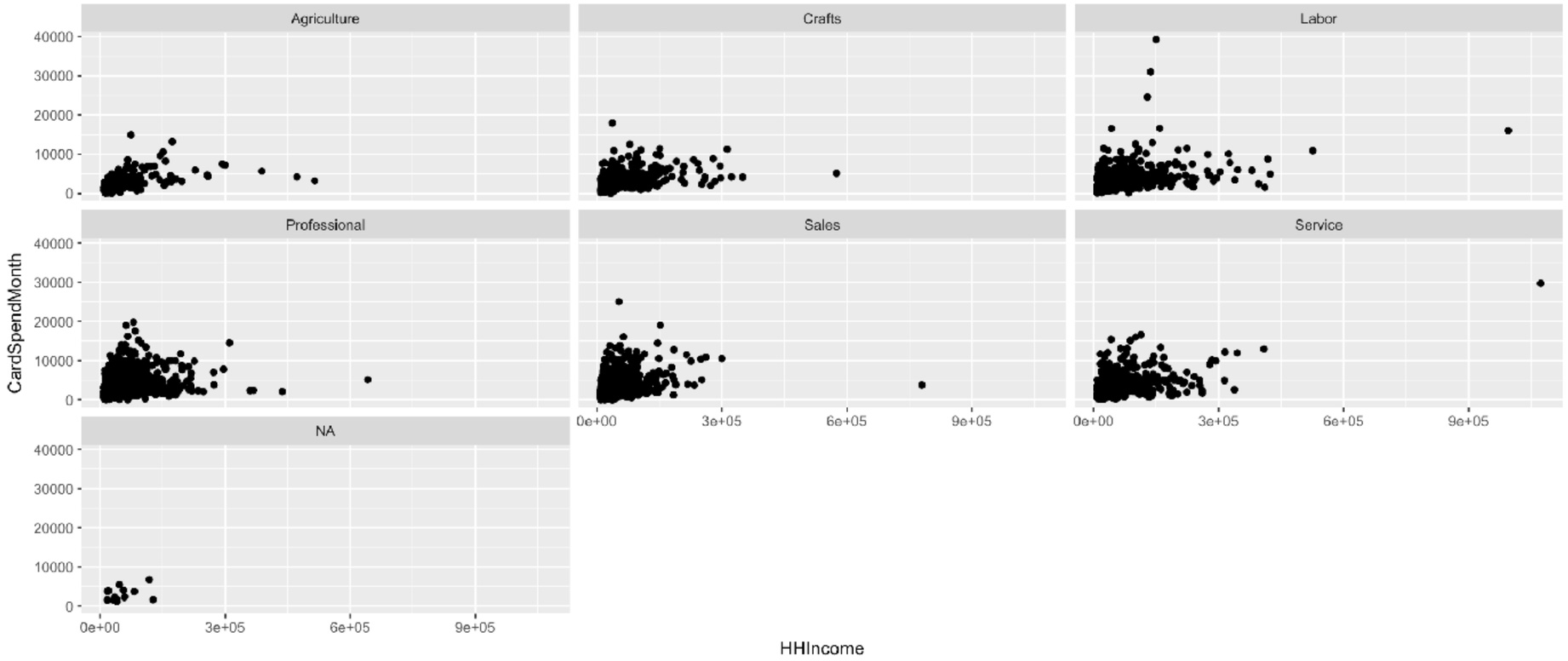
1. Create a scatter plot of ***HHIncome*** vs ***CardSpendMonth*** facetted by ***JobCategory***.

2. Create a scatter plot of ***HHIncome*** vs ***CardSpendMonth*** facetted by ***JobCategory*** and ***Gender***.

3. Assess ***UnionMember*** across each ***JobCategory***.

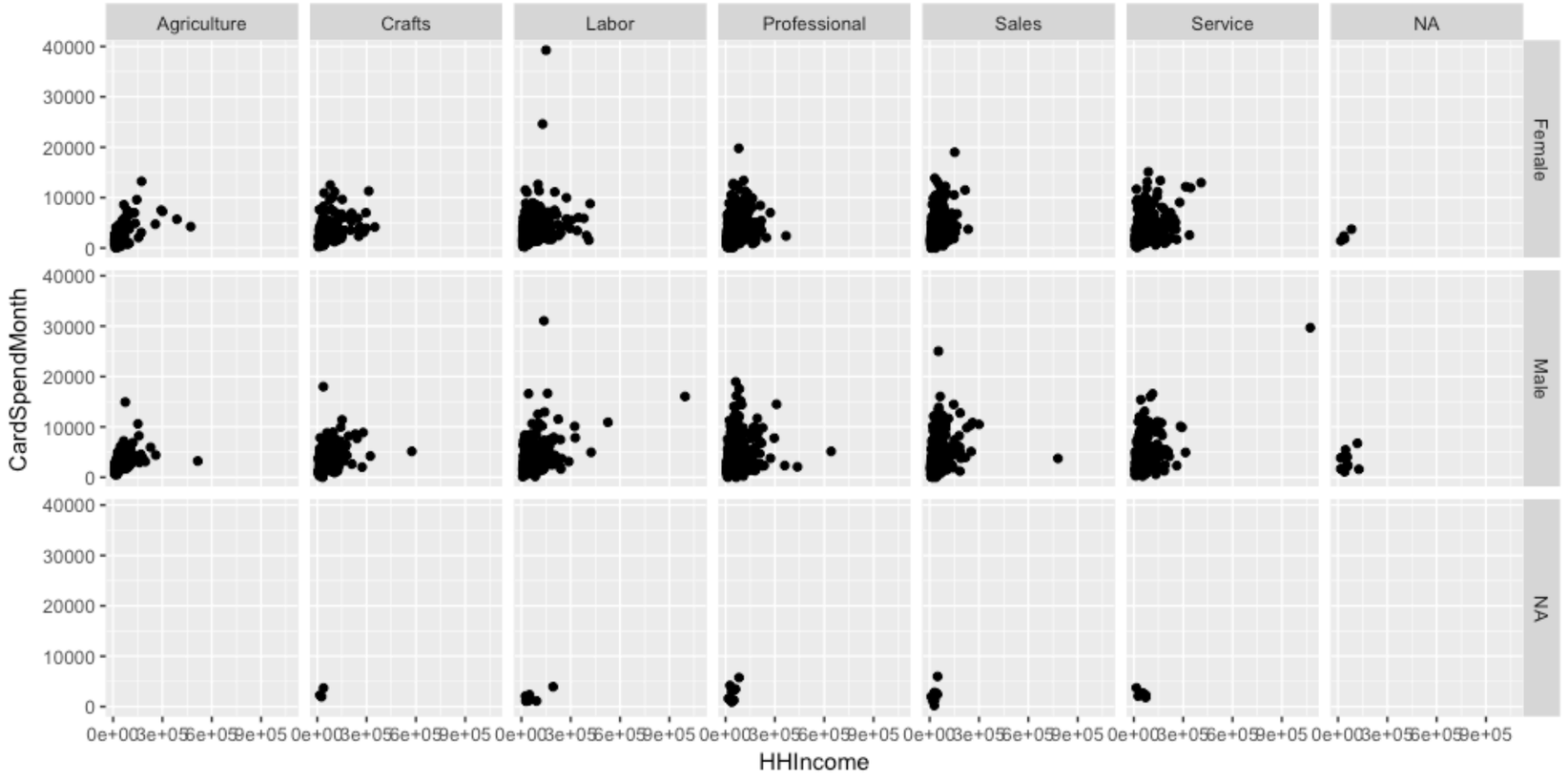
SOLUTION

```
# 2. Create a scatter plot of HHIncome vs CardSpendMonth facetted by JobCategory & Gender.
ggplot(customer, aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point() +
  facet_wrap(~ JobCategory)
```



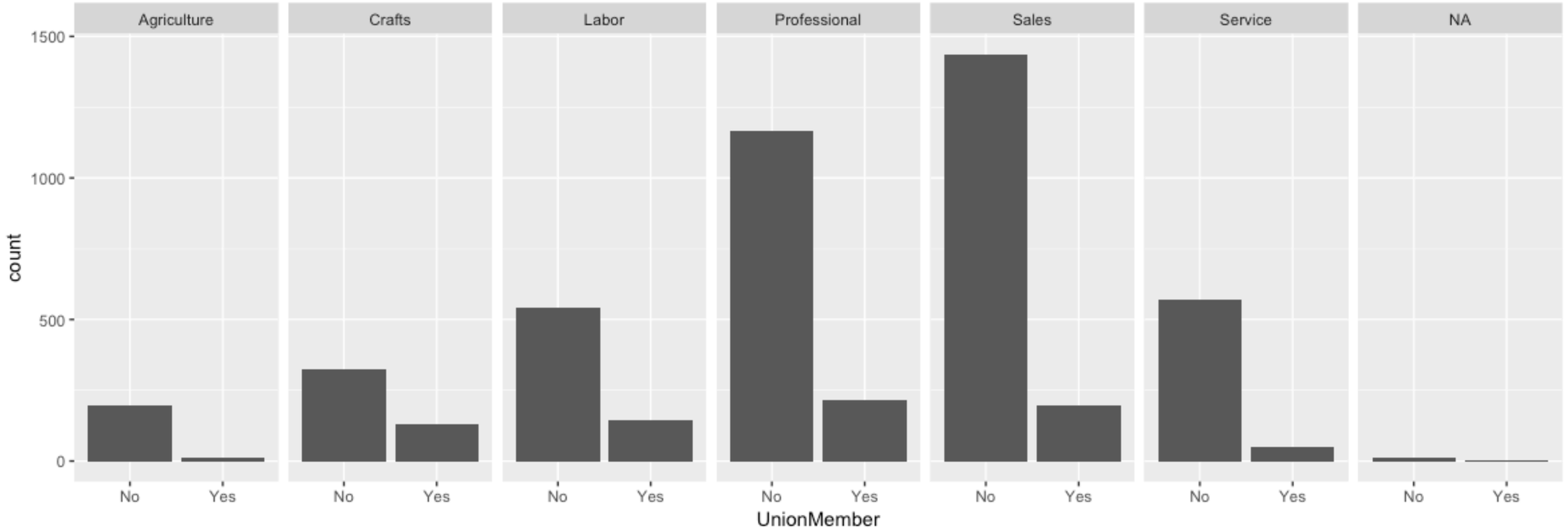
SOLUTION

```
# 2. Create a scatter plot of HHIncome vs CardSpendMonth facetted by JobCategory & Gender.
ggplot(customer, aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point() +
  facet_grid(Gender ~ JobCategory)
```



SOLUTION

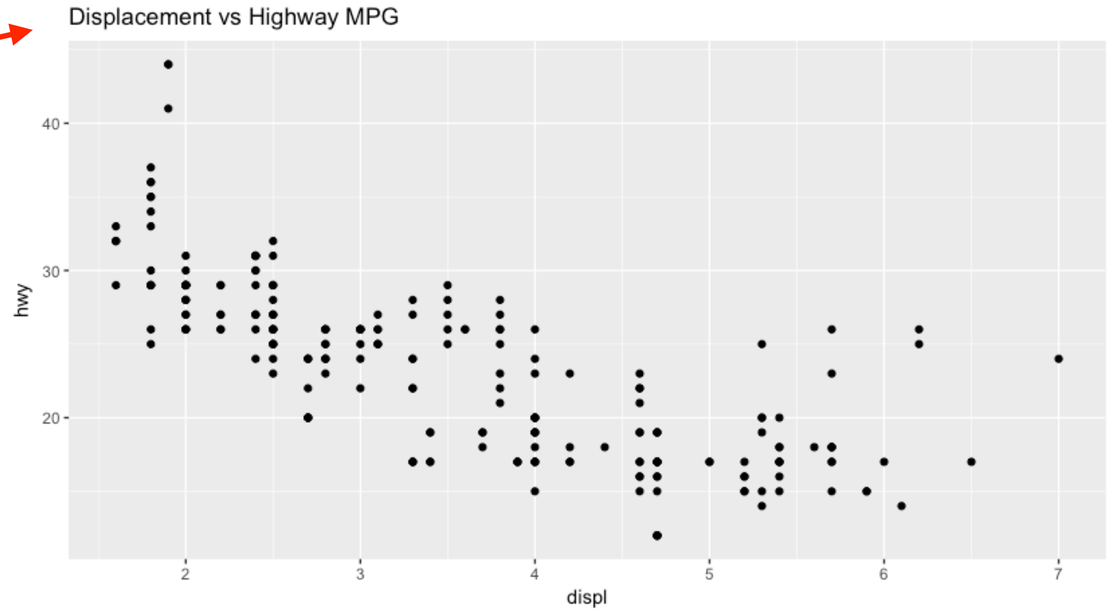
```
# 3. Assess UnionMember across each JobCategory
ggplot(customer, aes(x = UnionMember)) +
  geom_bar() +
  facet_wrap(~ JobCategory)
```

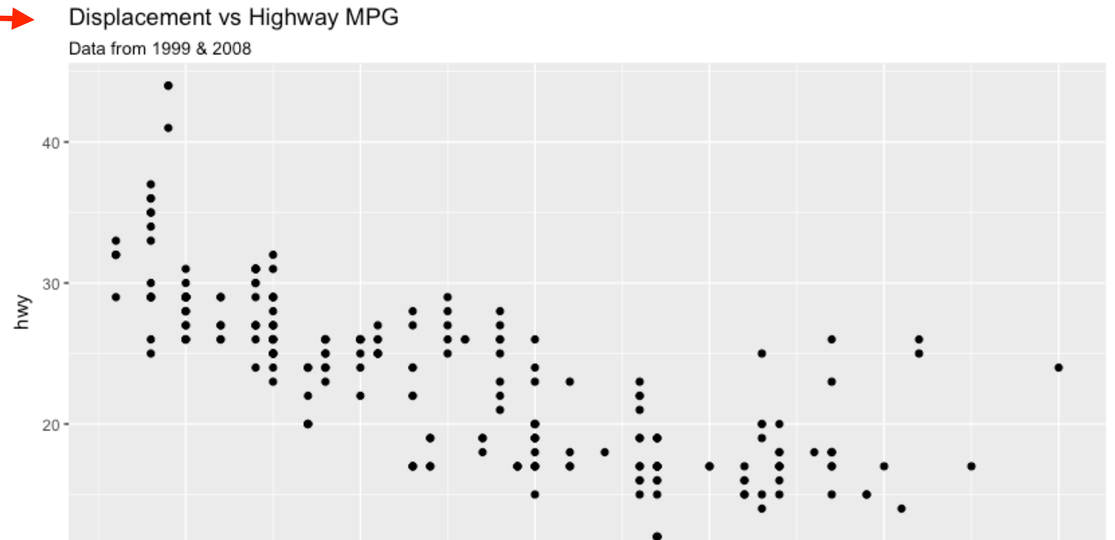


TITLES & AXES

ADDING TITLES

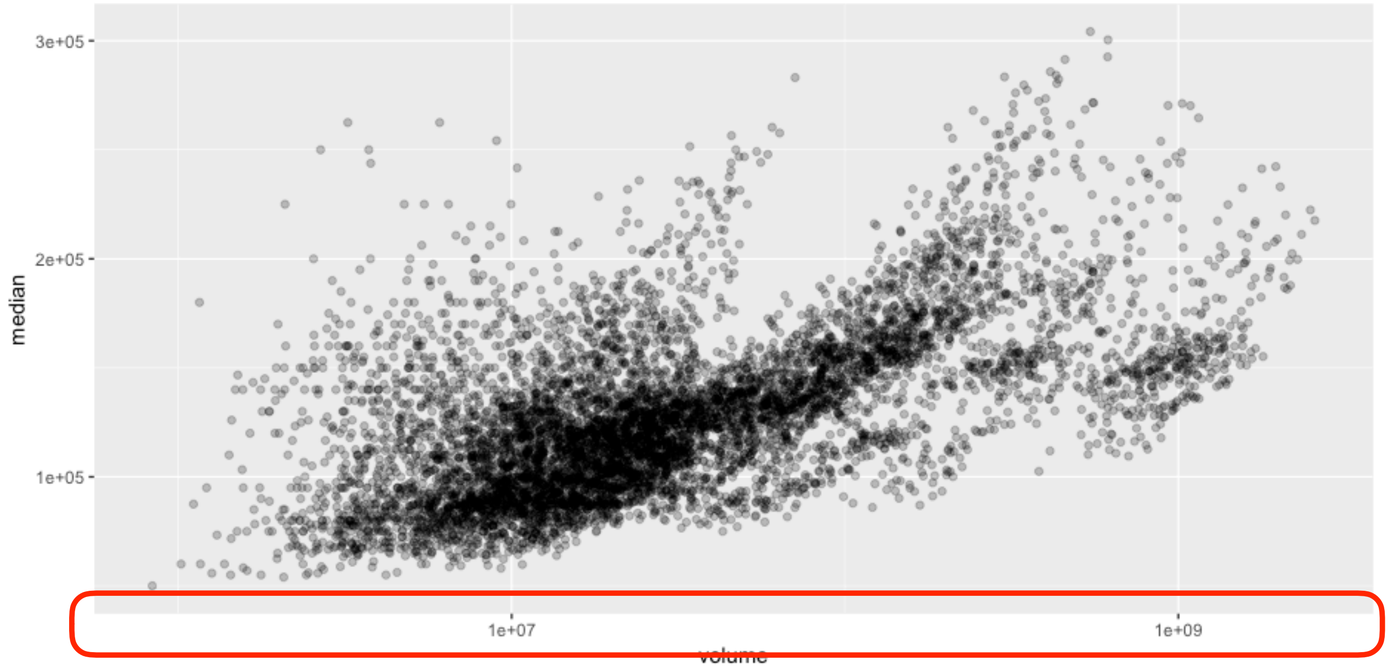
```
# top
ggplot(data = mpg, aes(x = class, y = hwy)) +
  geom_point() +
  ggtitle("Displacement vs Highway MPG")

# bottom
ggplot(data = mpg, aes(x = class, y = hwy)) +
  geom_boxplot() +
  ggtitle("Displacement vs Highway MPG",
          subtitle = "Data from 1999 & 2008")
```





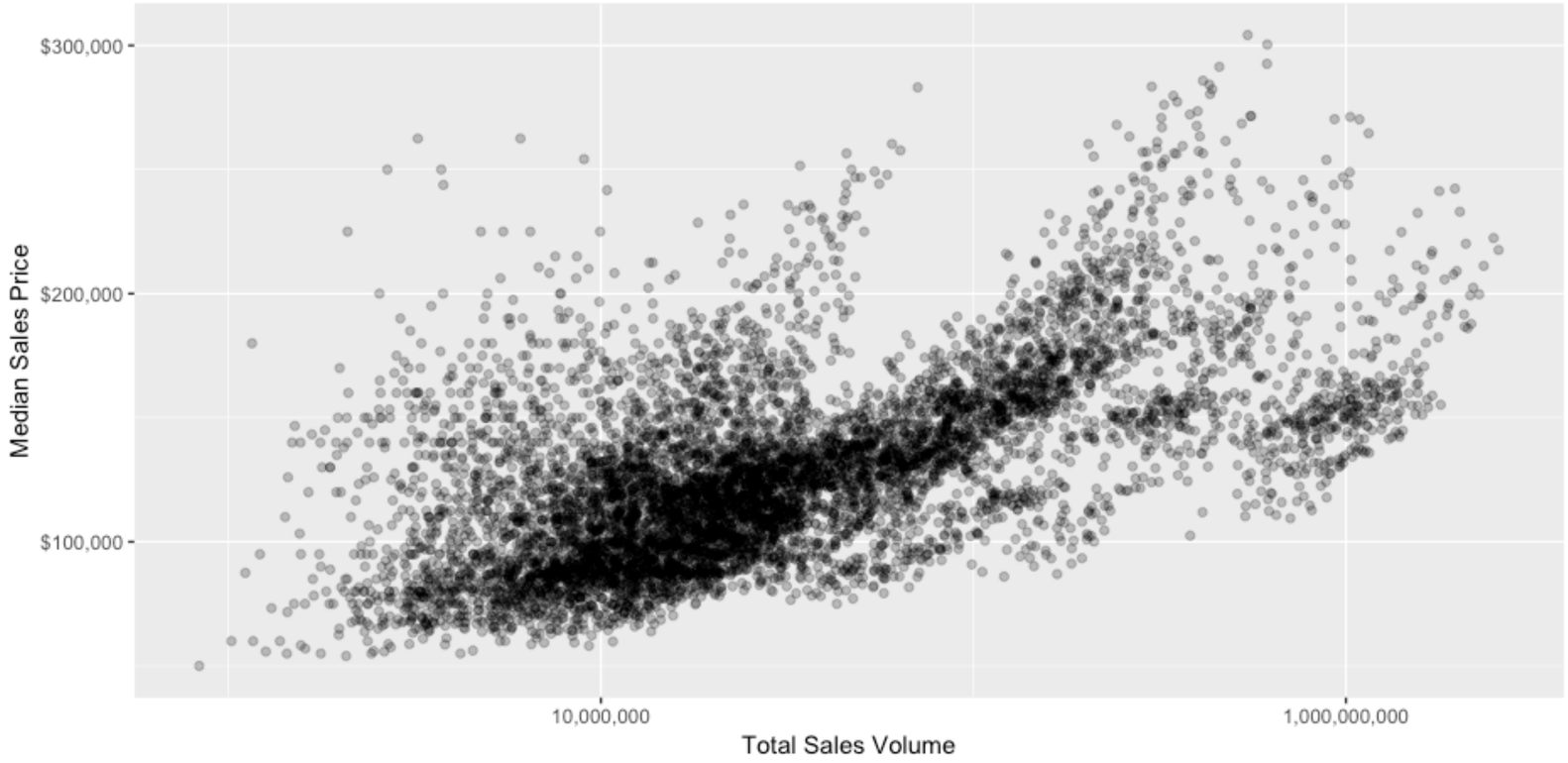
ADJUSTING AXIS SCALES

```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25) +
  scale_x_log10()
```



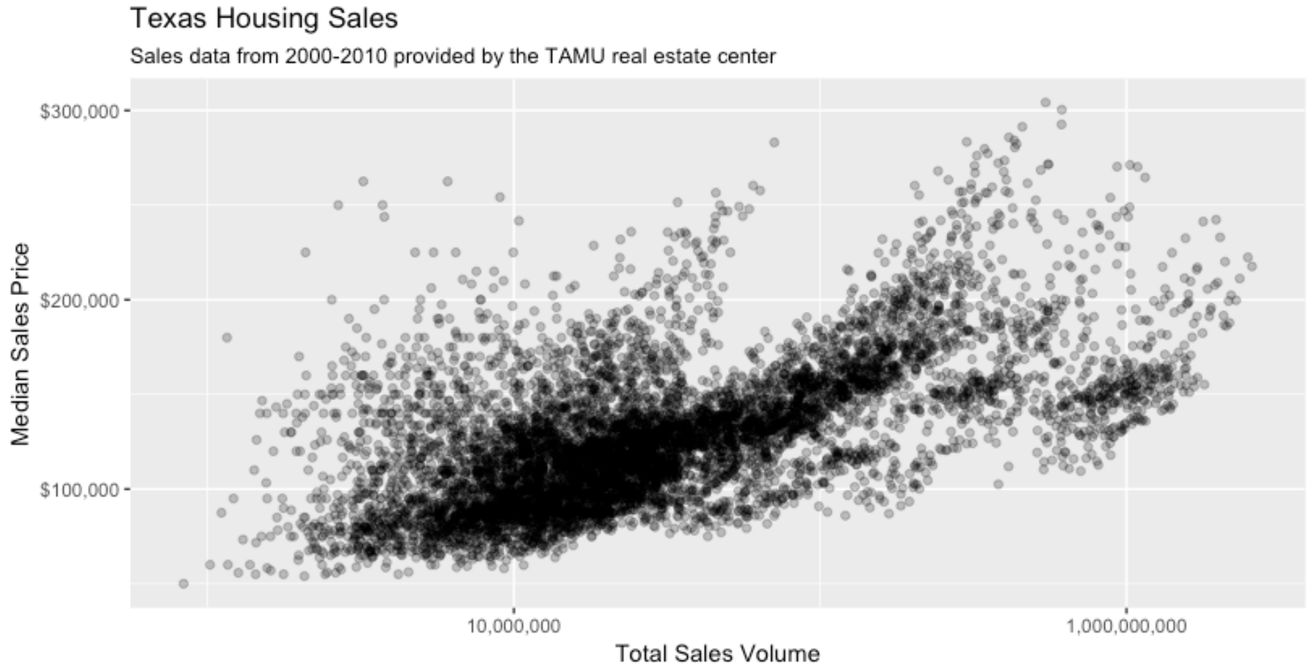
ADJUSTING AXIS TITLES & LABELS

```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25)  +
  scale_y_continuous(name = "Median Sales Price", labels = scales::dollar) +
  scale_x_log10(name = "Total Sales Volume", labels = scales::comma)
```



PUT IT ALL TOGETHER

```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25) +
  scale_y_continuous(name = "Median Sales Price", labels = scales::dollar) +
  scale_x_log10(name = "Total Sales Volume", labels = scales::comma) +
  ggtitle("Texas Housing Sales",
          subtitle = "Sales data from 2000-2010 provided by the TAMU real estate center")
```



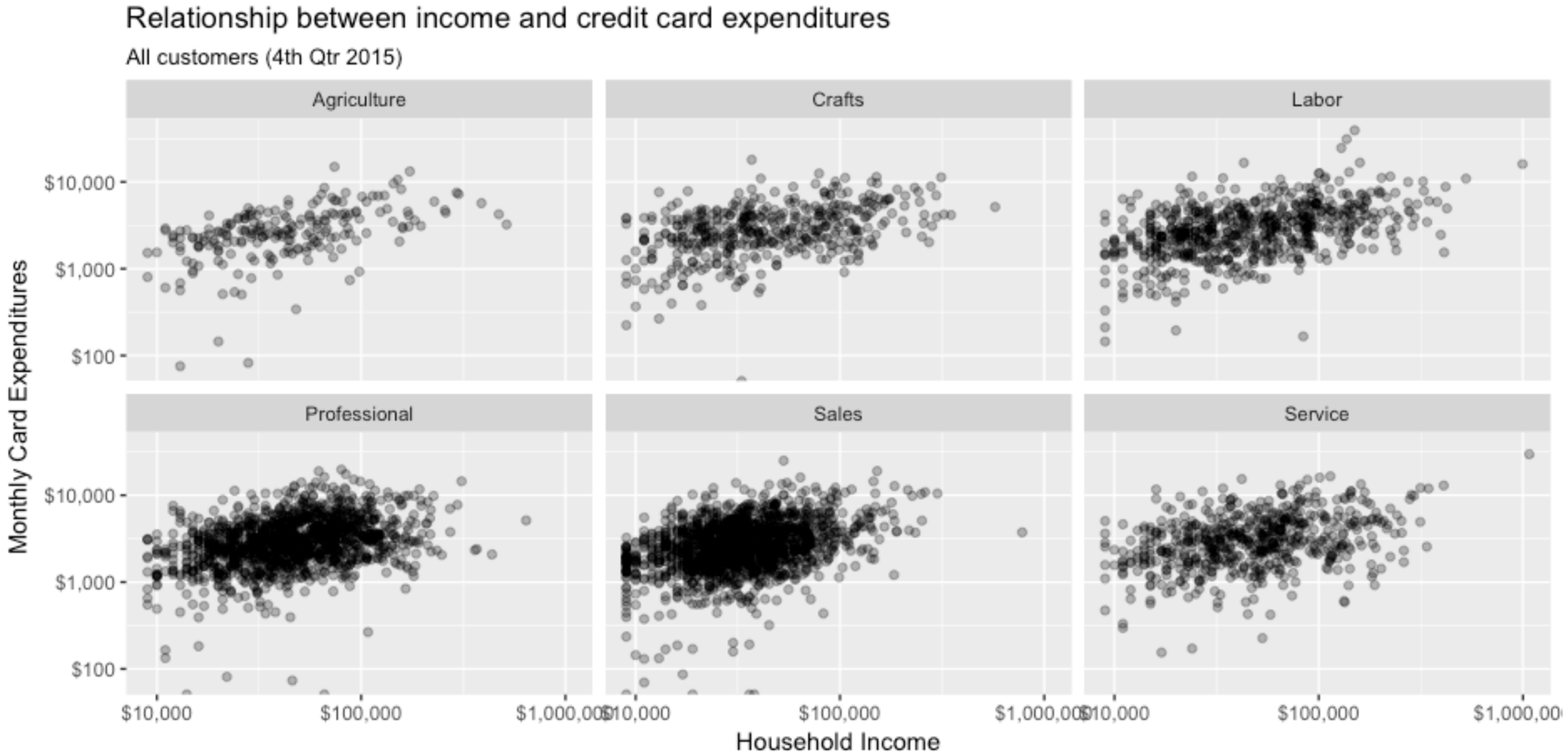
YOUR TURN!

1. Remove all missing values from the customer data and then…

2. Create a scatter plot of ***HHIncome*** vs ***CardSpendMonth*** facetted by ***JobCategory*** and…

3. add a title, subtitle, and nicely format the axes.

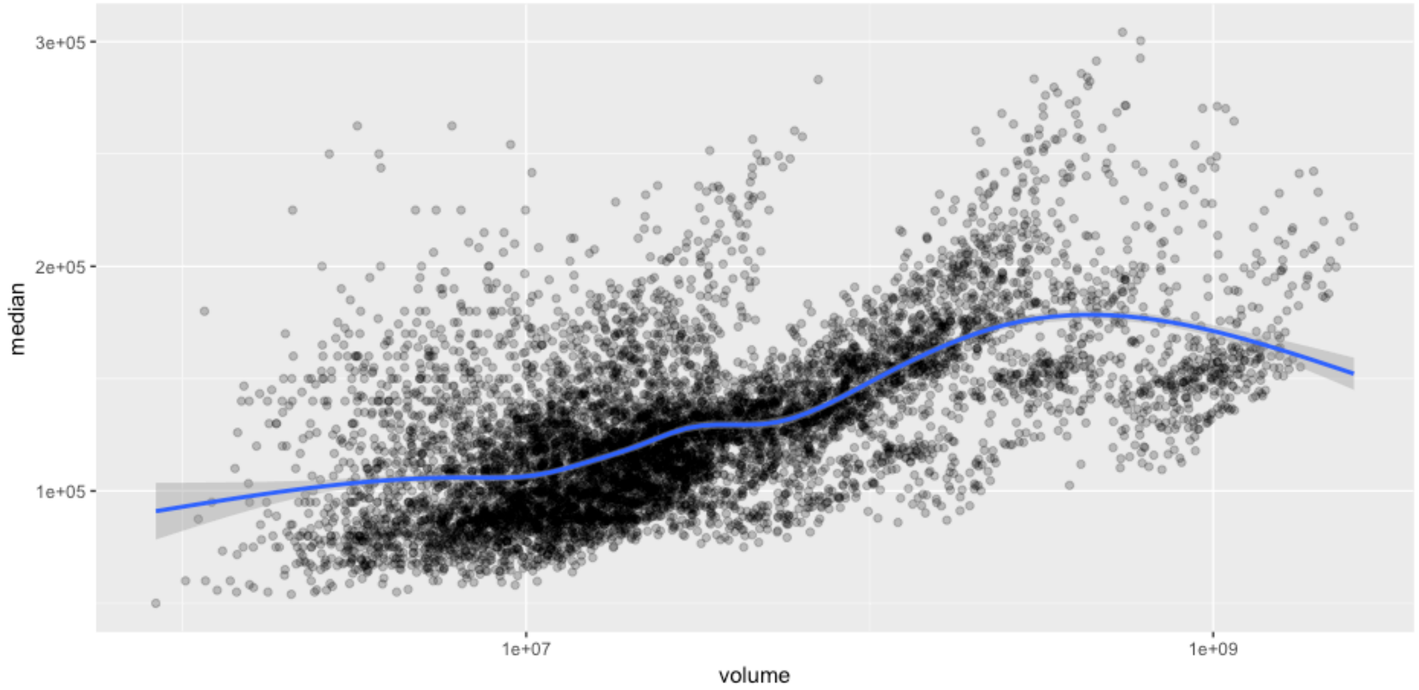
```
customer %>%
  na.omit() %>%
  ggplot(aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point(alpha = .3) +
  facet_wrap(~ JobCategory) +
  scale_x_log10("Household Income", labels = scales::dollar) +
  scale_y_log10("Monthly Card Expenditures", labels = scales::dollar) +
  ggtitle("Relationship between income and credit card expenditures",
          subtitle = "All customers (4th Qtr 2015)")
```



OVERPLOTTING

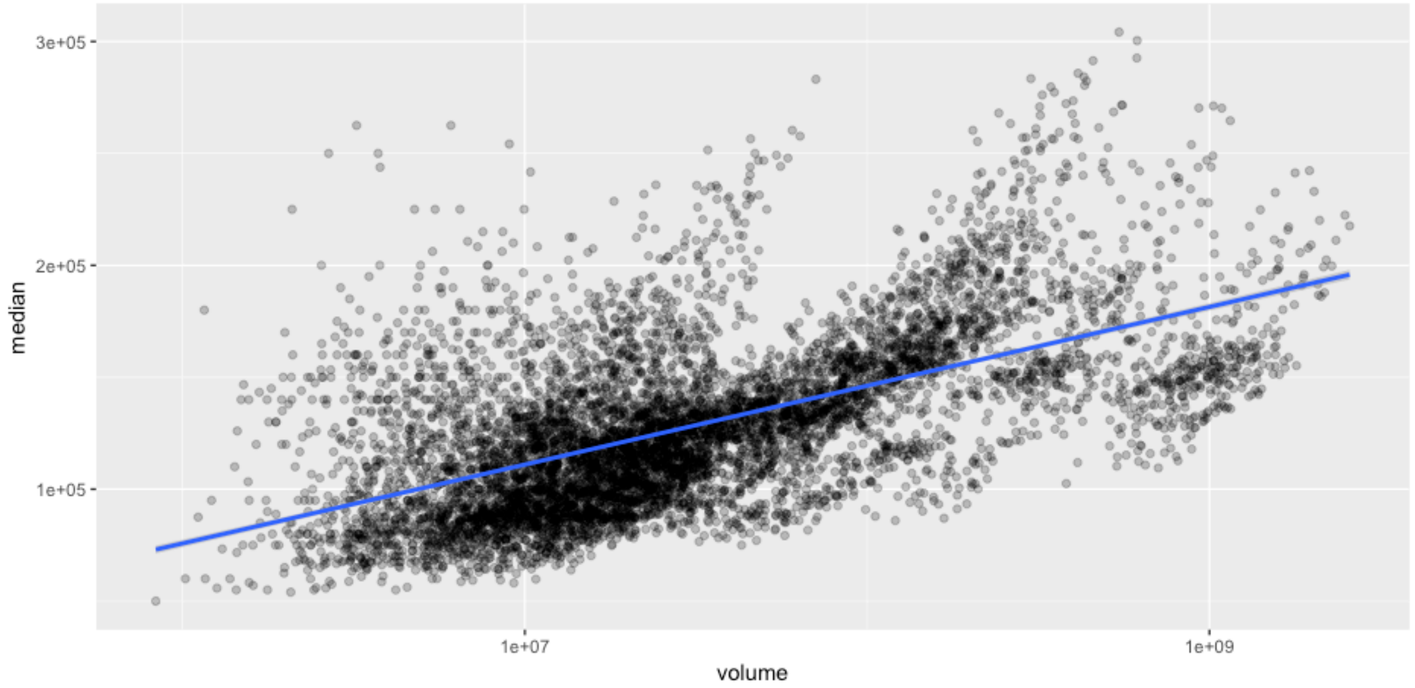
LAYERING HELPS DISPLAY PATTERNS

```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25)  +
  scale_x_log10() +
  geom_smooth()
```



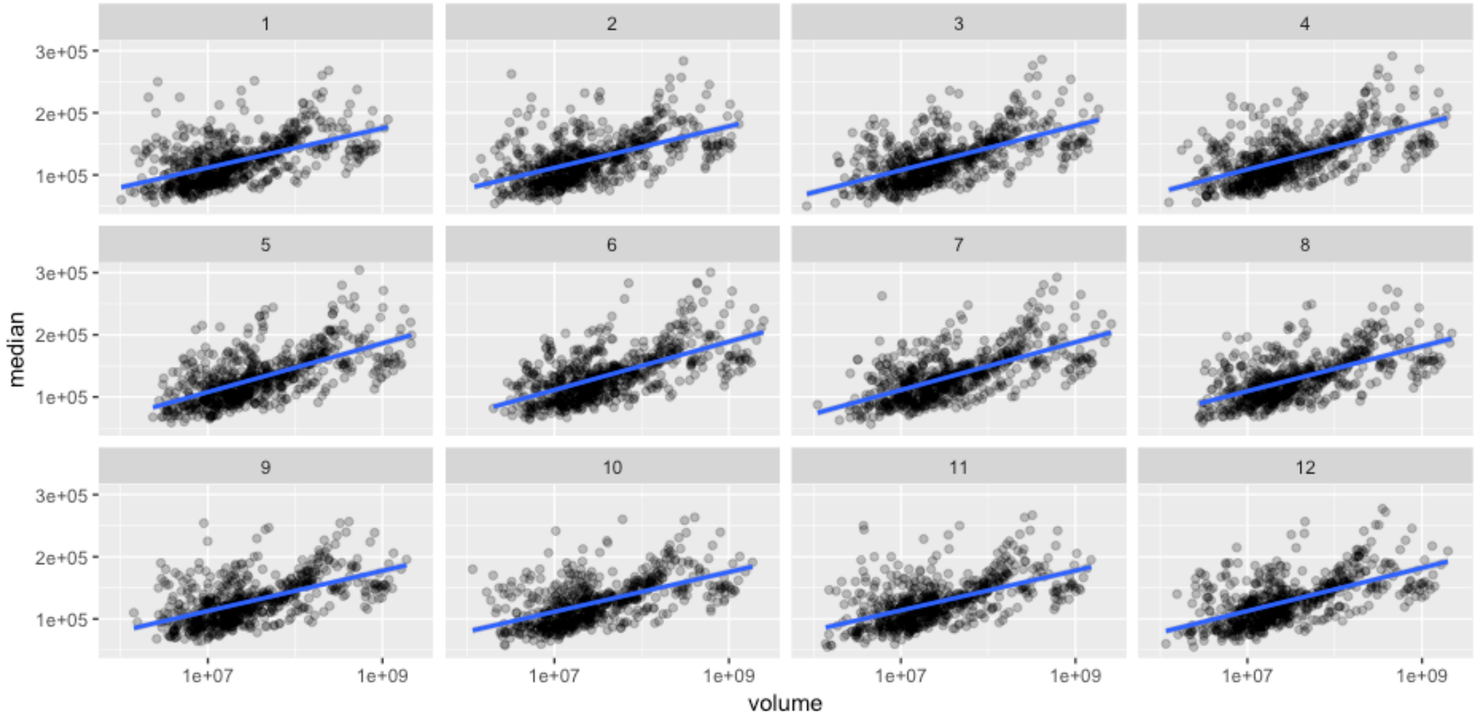
LAYERING HELPS DISPLAY PATTERNS

```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25)  +
  scale_x_log10() +
  geom_smooth(method = "lm")
```



LAYERING HELPS DISPLAY PATTERNS

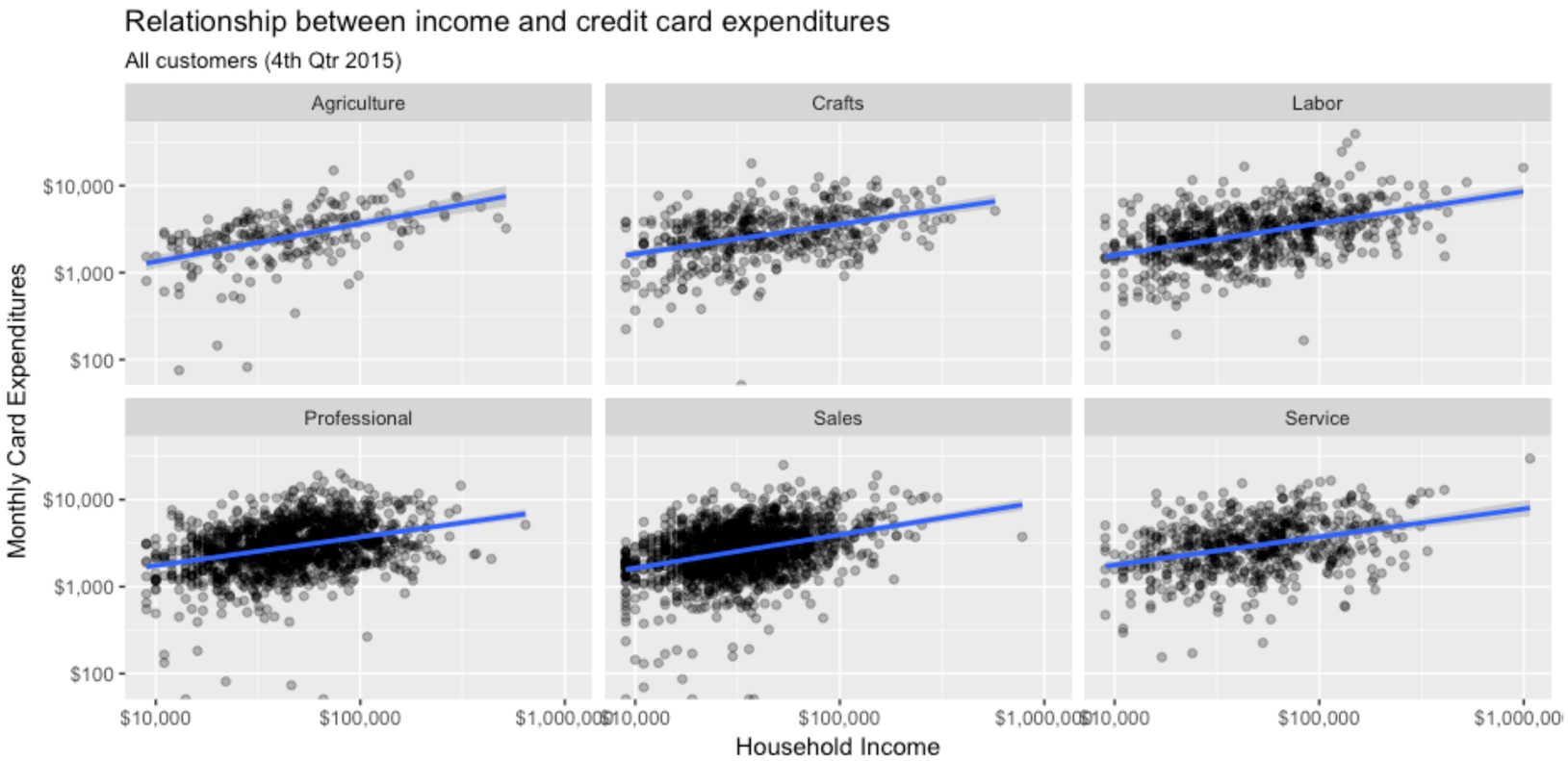
```
ggplot(data = txhousing, aes(x = volume, y = median)) +
  geom_point(alpha = .25)  +
  scale_x_log10() +
  geom_smooth(method = "lm") +
  facet_wrap(~ month)
```



YOUR TURN!

1. Remove all missing values from the customer data and then…
2. Create a scatter plot of ***HHIncome*** vs ***CardSpendMonth*** facetted by ***JobCategory*** and…
3. add a title, subtitle, and nicely format the axes and…
4. add a linear line to assess if the slope changes across ***JobCategory***

```
customer %>%
  na.omit() %>%
  ggplot(aes(x = HHIncome, y = CardSpendMonth)) +
  geom_point(alpha = .3) +
  geom_smooth(method = "lm") +
  facet_wrap(~ JobCategory) +
  scale_x_log10("Household Income", labels = scales::dollar) +
  scale_y_log10("Monthly Card Expenditures", labels = scales::dollar) +
  ggtitle("Relationship between income and credit card expenditures",
          subtitle = "All customers (4th Qtr 2015)")
```



LEVERAGE HELP AS YOU'RE LEARNING

Help >> Cheatsheets >> Data Visualization with ggplot2

Data Visualization with ggplot2 Cheat Sheet

RStudio

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system.**

To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.

Build a graph with **ggplot()** or **qplot()**

Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

Graphical Primitives

```
a <- ggplot(seals, aes(x = long, y = lat))
b <- ggplot(economics, aes(date, unemploy))
```

a + **geom_blank()**
(Useful for expanding limits)

a + **geom_curve**(aes(yend = lat + delta_lat, xend = long + delta_long, curvature = z))
x, xend, y, yend, alpha, angle, color, curvature, linetype, size

b + **geom_path**(lineend="butt", linejoin="round", linemitre=1)
x, y, alpha, color, group, linetype, size

b + **geom_polygon**(aes(group = group))
x, y, alpha, color, fill, group, linetype, size

a + **geom_rect**(aes(xmin = long, ymin = lat, xmax= long + delta_long, ymax = lat + delta_lat))
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

b + **geom_ribbon**(aes(ymin=unemploy - 900, ymax=unemploy + 900))
x, ymax, ymin, alpha, color, fill, group, linetype, size

a + **geom_segment**(aes(yend=lat + delta_lat, xend = long + delta_long))
x, xend, y, yend, alpha, color, linetype, size

a + **geom_spoke**(aes(yend = lat + delta_lat, xend = long + delta_long))
x, y, angle, radius, alpha, color, linetype, size

One Variable

Two Variables

Continuous X, Continuous Y

```
e <- ggplot(mpg, aes(cty, hwy))
```

e + **geom_label**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + **geom_jitter**(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

e + **geom_point()**
x, y, alpha, color, fill, shape, size, stroke

e + **geom_quantile()**
x, y, alpha, color, group, linetype, size, weight

e + **geom_rug**(sides = "bl")
x, y, alpha, color, linetype, size

e + **geom_smooth**(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + **geom_text**(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE)
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

```
f <- ggplot(mpg, aes(class, hwy))
```

f + **geom_bar(stat = "identity")**
x, y, alpha, color, fill, linetype, size, weight

Continuous Bivariate Distribution

```
h <- ggplot(diamonds, aes(carat, price))
```

h + **geom_bin2d**(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h + **geom_density2d()**
x, y, alpha, colour, group, linetype, size

h + **geom_hex()**
x, y, alpha, colour, fill, size

Continuous Function

```
i <- ggplot(economics, aes(date, unemploy))
```

i + **geom_area()**
x, y, alpha, color, fill, linetype, size

i + **geom_line()**
x, y, alpha, color, group, linetype, size

i + **geom_step**(direction = "hv")
x, y, alpha, color, group, linetype, size

Visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))
```

j + **geom_crossbar**(fatten = 2)
x, y, ymax, ymin, alpha, color, fill, group, linetype, size

WHAT TO REMEMBER

FUNCTIONS TO REMEMBER

Operator/Function	Description
`ggplot()`	Initializes a ggplot object (creates the blank canvas)
`aes()`	Creates aesthetic mappings
`geom_xx`	Geometric shapes to plot the data
`color, shape, size, alpha, etc`	Aesthetic parameters
`facet_wrap, facet_grid`	Create small multiples
`position`	Position argument (primarily used with bar charts)
`coord_xx`	Functions to adjust the coordinate system
`scale_xx`	Functions to adjust x and y axis