Applied Data Science with R: Effective Visualizations

Matthias Haber

03 April 2019

Last week's homework

Create lists of urls

```
library(rvest)
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 3.5.2
```

```
parsedURL <- paste0("https://comicvine.gamespot.com/",
"profile/arthurcbps/lists/top-200-heroes-of-marvel/14088/") %>%
  read_html()

urls <- html_nodes(parsedURL, "a") %>%
  html_attr("href") %>%
  str_extract("/.*/\\d{4}-\\d+") %>%
  .[!is.na(.)] %>%
  unique()
```

Loop through URLs and collect info

```r
library(tidyverse)
baseurl <- "https://comicvine.gamespot.com/"
allList <- list()

for(i in 1:21){
heroeHtml <- paste0(baseurl, urls[i]) %>%
  read_html()
variables <- heroeHtml %>%
  html_nodes("th") %>%
  html_text() %>%
  str_squish()
bio <- heroeHtml %>%
  html_nodes(".bar") %>%
  html_text() %>%
  str_squish() %>%
  str_replace_all("n/a", "")
df <- data.frame(variables, bio, stringsAsFactors = FALSE) %>%
  spread(variables, bio)
allList[[i]] <- df
}
```

Number of issues appearances

```r
dfAll <- reshape2::melt(allList)

dfAll %>%
  dplyr::mutate(`Appears in` = as.numeric(
    gsub(" issues", "", `Appears in`))) %>%
  top_n(1, `Appears in`) %>%
  select(`Super Name`)
```

```
##   Super Name
## 1 Spider-Man
```

Character Type

```
dfAll %>%
  group_by(`Character Type`) %>%
  summarise(n())
```

```
## # A tibble: 5 x 2
##   `Character Type` `n()`
##   <chr>            <int>
## 1 Alien                1
## 2 God/Eternal          1
## 3 Human                7
## 4 Mutant               7
## 5 Radiation            5
```

Never died

```r
dfAll %>%
  filter(Died == "None") %>%
  count()
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     3
```

Effective Visualization with R

Packages

```
library(tidyverse)
library(broom) # Tidy Model output
library(extrafont) # Custom fonts package
loadfonts() # Register custom fonts
library(gapminder) # Example GDP dataset
library(hrbrthemes) # Custom theme package
library(margins) # Compute marginal effects
library(MASS) # Statistical models package
library(scales) # Adjust scales
library(stargazer) # Produce beautiful tables
library(survival) # For survival models
```

Why Visualization is Important

“At their best, graphics are instruments for reasoning about quantitative information.” Tufte (1983)

“There is no statistical tool that is as powerful as a well-chosen graph.” Chambers et al. (1983)

“Diagrams prove nothing, but bring outstanding features readily to the eye.” Fisher (1925)

“Graphics should report the results of careful data analysis—rather than be an attempt to replace it.” Tukey (1993)

Goals

- Discovery goals:
  - Giving an overview—a qualitative sense of what is in a dataset
  - Conveying the sense of the scale and complexity of a dataset
- Communication goals:
  - Communication to self and others: Displaying information from the dataset in a readily understandable way
  - Telling a story
  - Attracting attention and stimulating interest

Interpreting a graph depends on expectations

- If readers have a lot of background knowledge, they will view the graphic differently don't assume you already have the reader's interest and involvement
- Making graphics attractive can help motivate readers to understand them

Graphics are part of a story

- A graphic does not live on its own
- There can be annotations, a legend, a title, a caption, accompanying text, an overall story, and a headline

Seven Rules for Better Figures (Rougier et al. 2014)

1. Know your audience
   - Who is the figure for?
2. Identify your message
   - What is the role of the figure?
3. Captions are not optional
   - Always use captions, explaining how to read a figure
4. Use color effectively
   - Color can be your greatest ally or your worst enemy (Tufte 1983)

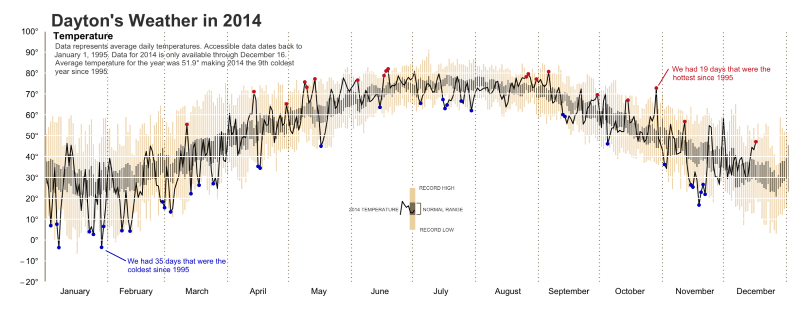
Seven Rules for Better Figures (Rougier et al. 2014)

5. Do not mislead the reader
    - A scientific figure is tied to the data
6. Avoid chartjunk
    - Get rid of any unnecessary non-data-ink
7. Get the right tool
    - Use R!

ggplot2

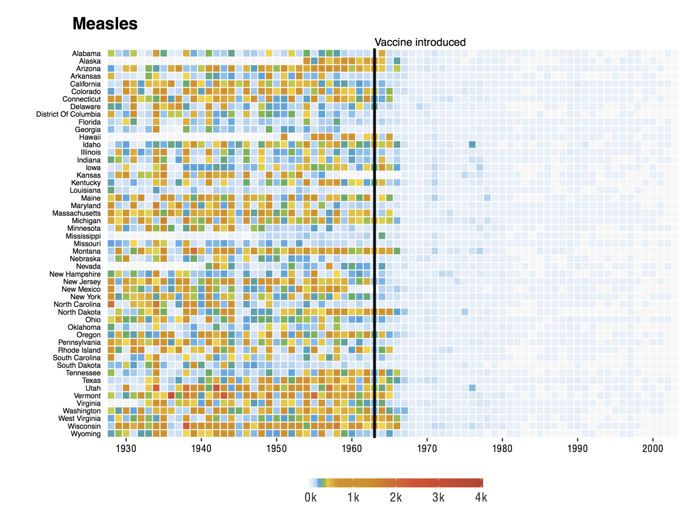
R has several systems for making graphs, but `ggplot2` is one of the most elegant and most versatile. `ggplot2` implements the grammar of graphics, a coherent system for describing and building graphs.

ggplot2 examples



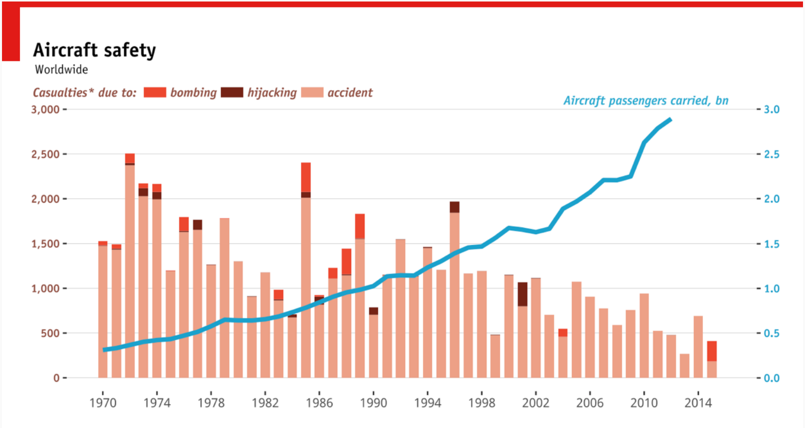
Source

ggplot2 examples



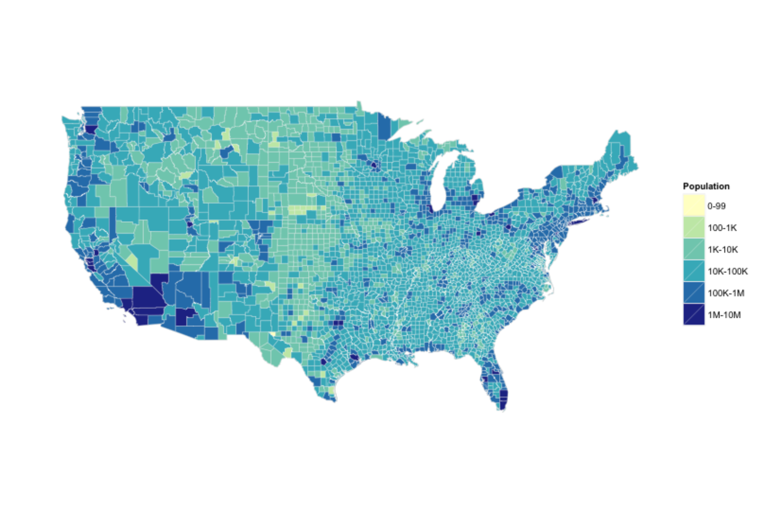
Source

ggplot2 examples



Source

ggplot2 examples



Source

The grammar of graphics

- Each plot is made of layers. Layers include the coordinate system (x-y), points, labels, etc.
- Each layer has aesthetics (`aes`) including x & y, size, shape, and color.
- The main layer types are called geometrics(`geom`) and include lines, points, etc.

The grammar of graphics

A `ggplot` is build piece by piece

1. Tidy Data

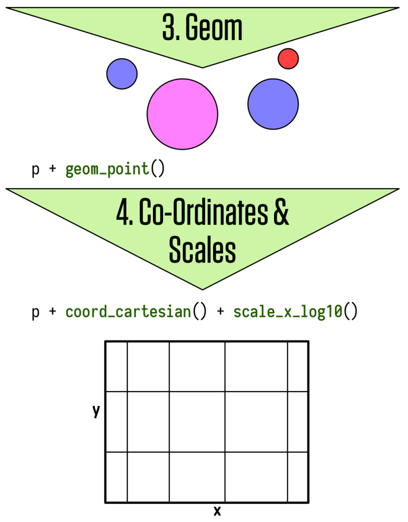
```
p <- ggplot(data = gapminder, ...
```

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

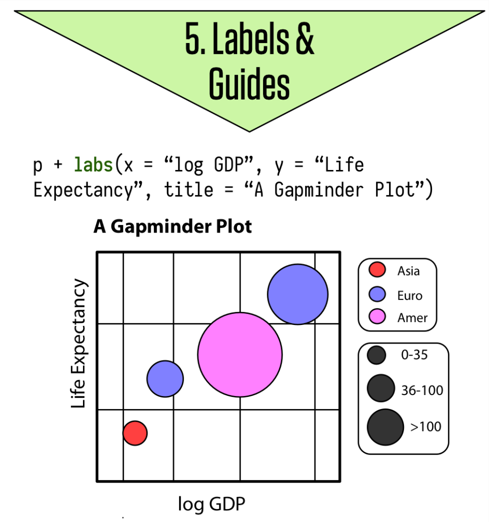
2. Mapping

```
p <- ggplot(data = gapminder, mapping =
        aes(x = gdp, y = lifexp, size = pop,
          color = continent))
```

The grammar of graphics



The grammar of graphics



ggplot workflow

1. Tell the `ggplot()` function what your data are.
2. Tell `ggplot` what relationships we want to see.
3. Tell `ggplot` how you want to see the relationships in your data.
4. Add additional layers to the p object one at a time.
5. Use additional functions to adjust scales, labels, tick marks.

Components of a ggplot2 graph

- data: Variables mapped to aesthetic attributes
- aesthetic: Visual property of the plot objects
- geom: Geometrical object used to represent data
- stats: Statistical transformations of the data
- scales: Values mapped to aesthetic attributes
- coord: Coordinate system
- facets: Subplots that each display one subset of the data

Tidy data

`ggplot` requires data to be tidy, with observations in rows and variables grouped in *key* | *value* columns.

Person	treatmentA	treatmentB
John Smith		2
Jane Doe	16	11

Person	treatment	result
John Smith	a	
Jane Doe	a	16
John Smith	b	2
Jane Doe	b	11

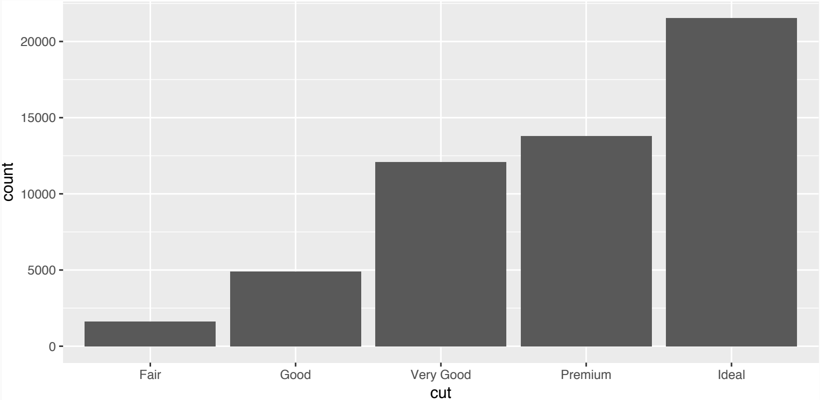
Plotting Distributions

Variation

- Variation is the difference between expected output to observed output.
- Visualization of the distribution is different for categorical (`fctr`, `chr`) and continuous (`dbl`, `int`, `dttm`) variables

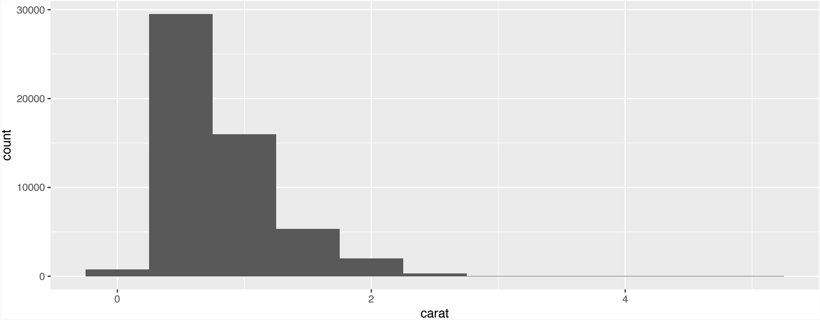
Distributions of categorical data

```
ggplot(data=diamonds) +
  geom_bar(mapping = aes(x = cut))
```



Distributions of continuous data

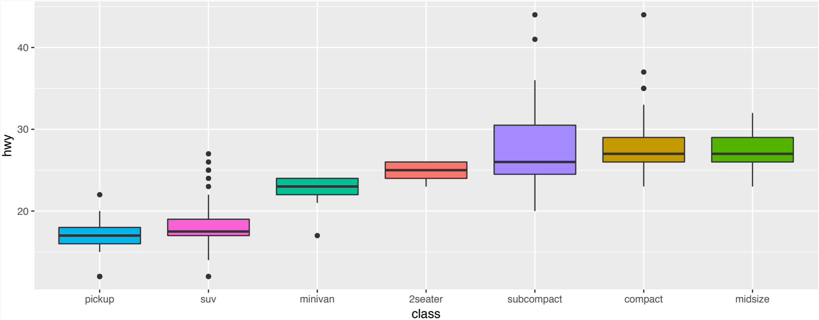
```
ggplot(data=diamonds) +
  geom_histogram(mapping = aes(x = carat),
                 binwidth = 0.5)
```



Boxplot

We can use `geom_boxplot()` to plot covariation between continuous and catagorical variables

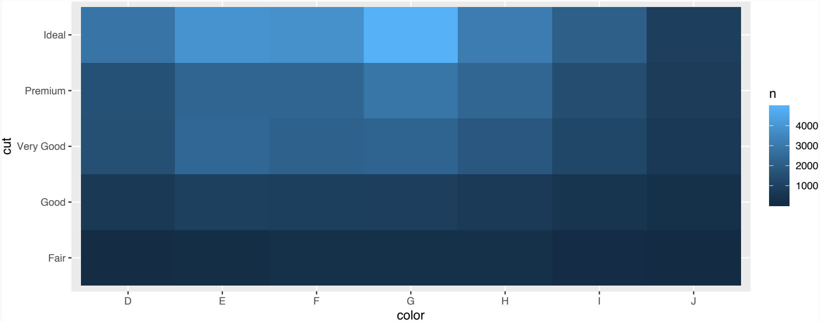
```
ggplot(data = mpg, aes(x = class, y = hwy, fill = class)) +
  geom_boxplot(aes(x=reorder(class, hwy,FUN = median),
                   y = hwy)) +
  theme(legend.position = "none")
```



Tile Plot

We can use `geom_tile` to plot the covariation between two categorical variables

```
diamonds %>%
  count(color, cut) %>%
  ggplot(mapping = aes(x = color, y = cut)) +
    geom_tile(mapping = aes(fill = n))
```

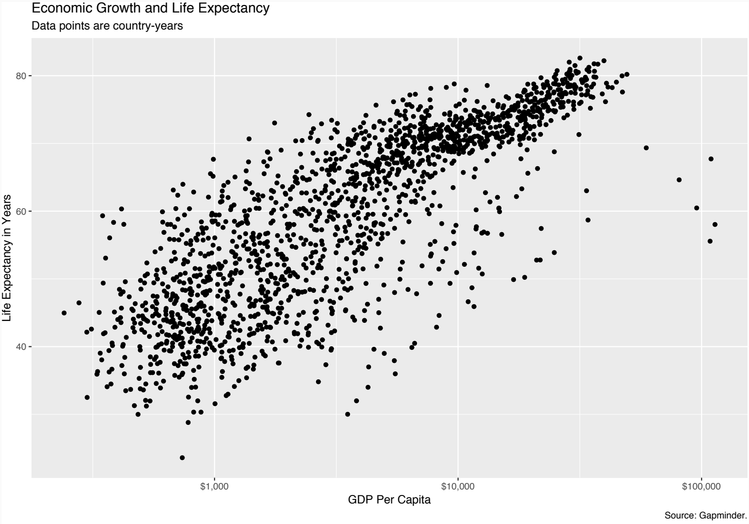


Scatter Plots

The easiest way to visualize the covariation between two continuous variables is to draw a scatterplot with `geom_point()`.

```r
p <- ggplot(data=gapminder, mapping = aes(x = gdpPercap,
                                           y = lifeExp)) +
  geom_point() +
  scale_x_log10(labels = scales::dollar) +
  labs(x = "GDP Per Capita",
       y = "Life Expectancy in Years",
       title = "Economic Growth and Life Expectancy",
       subtitle = "Data points are country-years",
       caption = "Source: Gapminder.")
```

Scatter Plots



Presenting model-based graphics

Goals

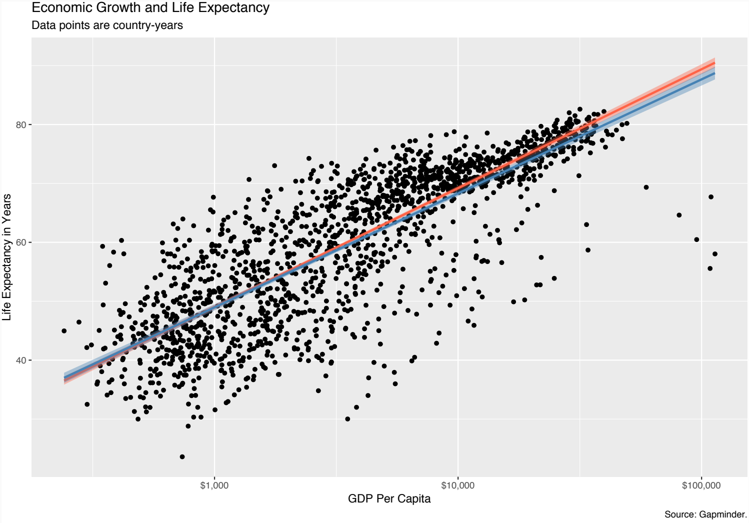
1. Show how `ggplot` can use various modeling techniques directly within geoms
2. Tidily extract and plot estimates of models that we fit ourselves

OLS vs. Robust Regression

- The `geom_smooth()` function can take a range of method arguments to fit LOESS, OLS, and robust regression lines
- `geom_smooth()` can also be instructed to use different formulas to produce their fits

```
p_ols <- p +
  geom_smooth(color = "tomato", fill="tomato",
                              method = MASS::rlm) +
  geom_smooth(color = "steelblue", fill="steelblue",
              method = "lm")
```

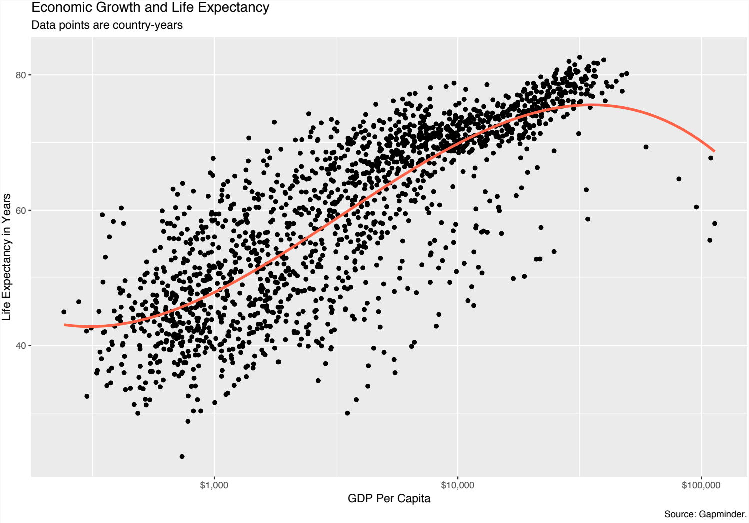
OLS vs. Robust Regression



Polynominal fit

```
p_poly <- p +
  geom_smooth(color = "tomato",
              method = "lm", size = 1.2,
              formula = y ~ splines::bs(x, 3),
              se = FALSE)
```

Polynominal fit



Plot Model Output

- Figures based on statistical models face all the ordinary challenges of effective data visualization
- The more complex the model,a the trickier it becomes to convey this information effectively

Another Look at the Gapminder Data

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

Linear Model of Life Expectancy

```
out <- lm(formula = lifeExp ~ gdpPercap + pop +
             continent, data = gapminder)
summary(out)
```

```
##
## Call:
## lm(formula = lifeExp ~ gdpPercap + pop + continent, data = gapminder
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -49.161  -4.486   0.297   5.110  25.175
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       4.781e+01  3.395e-01 140.819  < 2e-16 ***
## gdpPercap         4.495e-04  2.346e-05  19.158  < 2e-16 ***
## pop               6.570e-09  1.975e-09   3.326 0.000901 ***
## continentAmericas 1.348e+01  6.000e-01  22.458  < 2e-16 ***
```

Present your findings in substantive terms

- Show results in context where other variables are held at sensible values (e.g. mean or median)
- For continuous variables, generate predicted values that cover some meaningful range of the distribution (e.g. 25th to the 75th percentile)
- For unordered categorical variables, predicted values might be presented with respect to the modal category
- Use a scale that readers can easily understand, e.g. use predicted probabilities if your model reports log-odds
- Show confidence intervals and measures of model fit when you present your results

Tidy Data

- we can use the `tidy` function from the `broom` packages to turn our model object into a data frame that we can plot with `ggplot`

```
out_tidy <- tidy(out, conf.int = TRUE)
```

Export Tables to Word

To export tables to Word, follow these general steps:

1. Create a table or data.frame in R.
2. Write this table to a comma-separated .txt file using write.table().
3. Copy and paste the content of the .txt file into Word.
4. In Word,
    - select the text you just pasted from the .txt file
    - go to Table → Convert → Convert Text to Table…
    - make sure “Commas” is selected under “Separate text at”, click OK

Export Tables to Word

```r
write.table(out_tidy, file = "model_out.txt",
            sep = ";", quote = FALSE,
            row.names = FALSE)
```

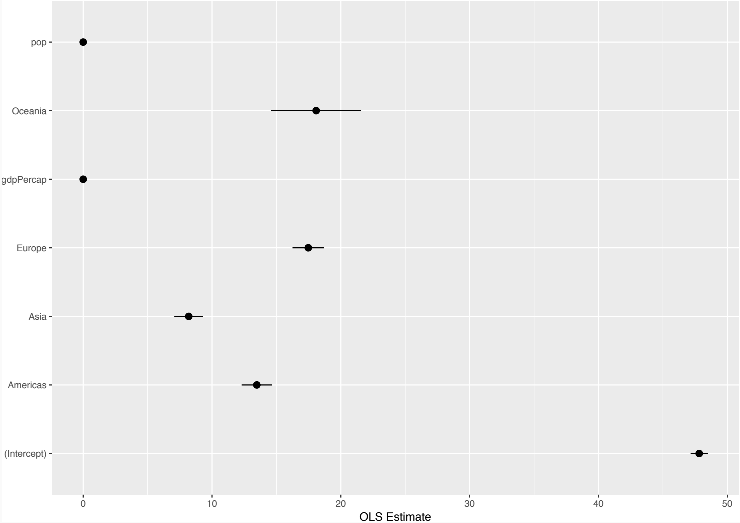
Export with Stargazer

```r
stargazer(out, align = TRUE) # Latex
stargazer(out, type = "text", align = TRUE) # Word
```

Coefficient Plot

```
out_tidy <- out_tidy %>%
  mutate(term = gsub("continent", "", term))
p <- ggplot(out_tidy, mapping = aes(x = term,
                                    y = estimate,
                                    ymin = conf.low,
                                    ymax = conf.high)) +
  geom_pointrange() +
  coord_flip() +
  labs(x="", y="OLS Estimate")
```

Coefficient Plot



Predictions

- Use predictions to get a picture of the estimates your model produces over the range of some particular variable, holding other covariates constant at some sensible values
- For example, predict `gdpPercap` from minimum to maximum, holding `pop` constant at its median and letting `continent` take all of its five available values

Prepare Data For Predictions

```
min_gdp <- min(gapminder$gdpPercap)
max_gdp <- max(gapminder$gdpPercap)
med_pop <- median(gapminder$pop)

pred_df <- expand.grid(gdpPercap =
                         (seq(from = min_gdp,
                              to = max_gdp,
                              length.out = 100)),
                       pop = med_pop,
                       continent = c("Africa",
                                     "Americas",
                                     "Asia", "Europe",
                                     "Oceania"))
```

Generate Predictions

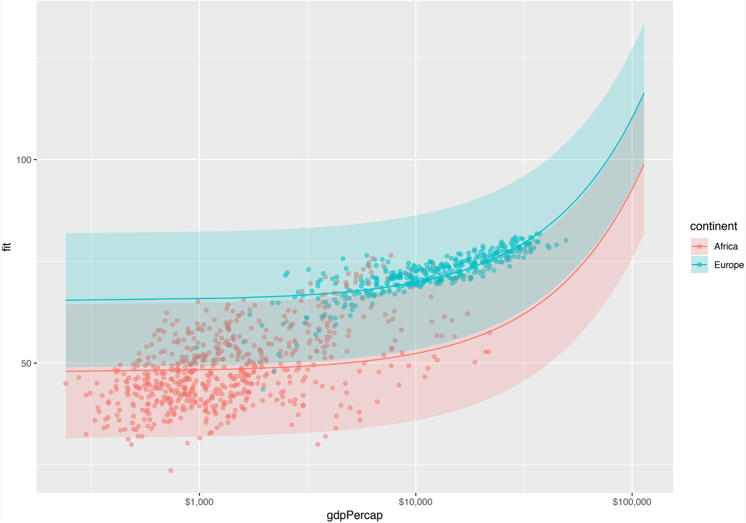
- we can use `predict()` with our new data and model to calculate the fitted values for every row in the data frame and merge the results with `pred_df`

```
pred_out <- stats::predict(object = out,
                           newdata = pred_df,
                           interval = "predict") # 95% CI
pred_full <- cbind(pred_df, pred_out)
```

Plot Predictions For Europe and Africa

```r
p <- ggplot(data = subset(pred_full,
                          continent %in% c("Europe", "Africa")),
            aes(x = gdpPercap, y = fit, ymin = lwr,
                ymax = upr,
                color = continent, fill = continent,
                group = continent)) +
  geom_point(data = subset(gapminder,
                           continent %in% c("Europe", "Africa")),
             aes(x = gdpPercap, y = lifeExp,
                 color = continent),
             alpha = 0.5,
             inherit.aes = FALSE) +
  geom_line() +
  geom_ribbon(alpha = 0.2, color = FALSE) +
  scale_x_log10(labels = scales::dollar)
```

Plot Predictions For Europe and Africa



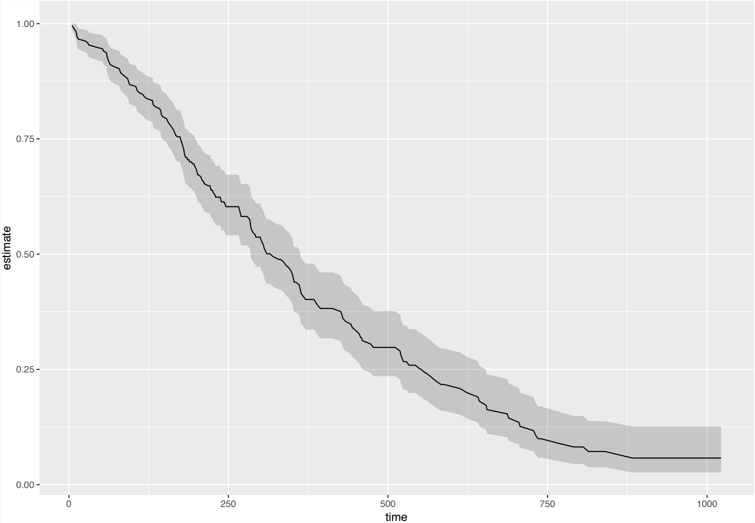
Tidy Results from a Survival Model

```
out_cph <- coxph(Surv(time, status) ~ age + sex,
                 data = lung)
out_surv <- survfit(out_cph)
out_tidy <- tidy(out_surv)
```

Plot Survival Model Output

```
p <- ggplot(data = out_tidy,
            mapping = aes(time, estimate)) +
  geom_line() +
  geom_ribbon(mapping = aes(ymin = conf.low,
                            ymax = conf.high),
              alpha = .2)
```

Plot Survival Model Output



Generate Marginal Effects

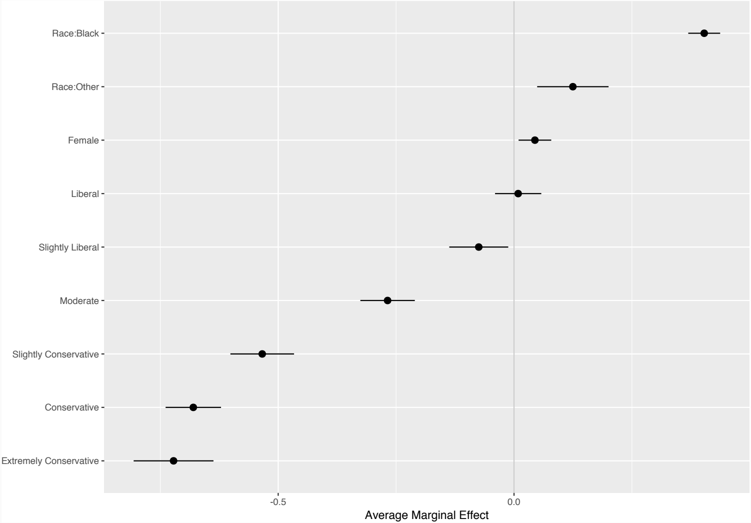
- Using the General Social Survey data let's fit a logistic regression on obama, with age, polviews, race, and sex as predictors.

```
load("gss.RData")
gss_sm$polviews_m <- relevel(gss_sm$polviews,
                              ref = "Moderate")
out_bo <- glm(obama ~ polviews + sex*race,
              family = "binomial", data = gss_sm)
bo_m <- margins(out_bo)
bo_gg <- as.tibble(summary(bo_m)) %>%
  mutate(factor = gsub("polviews|sex", "", factor)) %>%
  mutate(factor = gsub("race", "Race:", factor))
```

Plot Marginal Effects

```r
p <- ggplot(data = bo_gg, aes(x = reorder(factor, AME),
                              y = AME, ymin = lower,
                              ymax = upper)) +
  geom_hline(yintercept = 0, color = "gray80") +
  geom_pointrange() + coord_flip() +
  labs(x = NULL, y = "Average Marginal Effect")
```

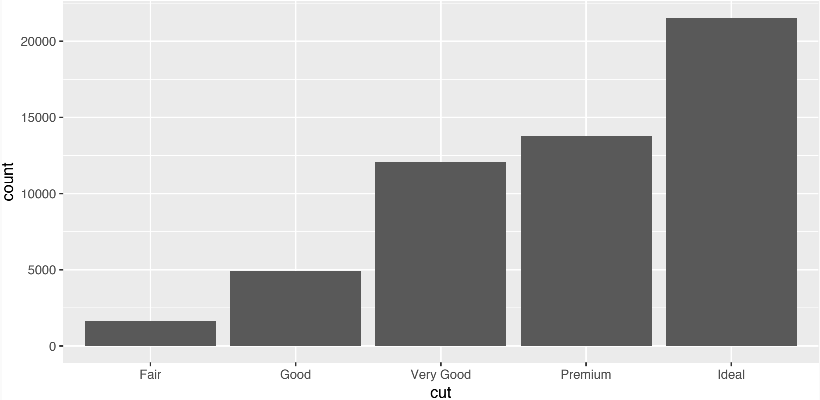
Plot Marginal Effects



Making Plots Pretty

Remember our old barplot

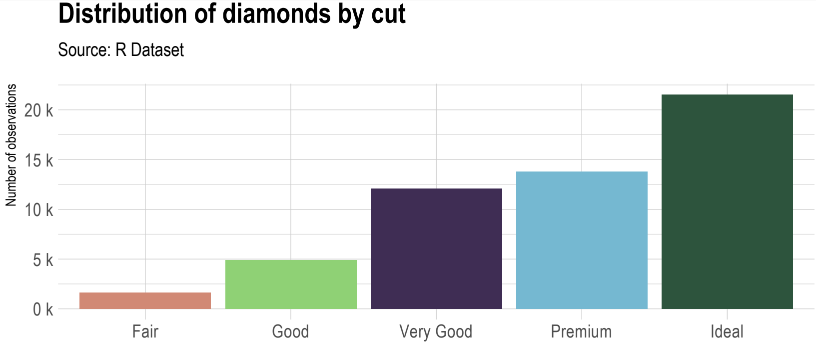
```
ggplot(data=diamonds) +
  geom_bar(mapping = aes(x = cut))
```



Prettier version

```r
library(hrbrthemes)
p <- ggplot(data=diamonds) +
  geom_bar(mapping = aes(x = cut, fill = cut)) +
  theme_ipsum() + # custom theme
  scale_fill_ipsum() + # add colors
  scale_y_continuous(label = scales::unit_format(
    unit = "k", scale = 1e-3)) + # change y-scale
  labs(x="", y="Number of observations",
       title="Distribution of diamonds by cut",
       subtitle = "Source: R Dataset") + # titles
  theme(legend.position =  "none") # remove legend
```

Prettier version



Need Help

ggplot Cheat Sheet

Homework Exercises

Homework Exercises

No Homework this week.

That's it for today. Questions?