STAT 408: Week 3

Data Visualization with ggplot

2/1/2022

Tidyverse Overview

Tidyverse overview

- The `tidyverse` contains `dplyr`, `ggplot2` and a set of other useful R packages.
- Most of the packages in the tidyverse were created by Hadley Wickham.
- The tidyverse is a modern way to express R code for data wrangling, storage, and visualization.

Installing and loading packages

There are two steps for using packages: installation and loading.

```
install.packages('tidyverse')
library(tidyverse)
```

Images sourced from https://www.wikihow.com/Change-a-Light-Bulb

Reading data into R with `readr`

The readr package is very useful for reading data into R.

```
okcupid <-
  read_csv('https://math.montana.edu/shancock/data/OKCupid_profil
```

Data manipulation with dplyr

dplyr is useful for data manipulation and can be characterized by a set of verbs:

- select
- filter
- group_by
- mutate
- sample. . .

```
sample_n(okcupid, 2)
```

```
## # A tibble: 2 x 10
##     age body_type diet          drinks drugs ethnicity height job   s
##   <dbl> <chr>     <chr>         <chr>  <chr> <chr>       <dbl> <chr> <
## 1    29 curvy     mostly anyth~ socia~ never white          69 other f
## 2    32 athletic  mostly anyth~ rarely never indian         66 comp~ m
```

Piping with %>%

The symbol %>% is a piping operator that can be used to connect commands.

```
okcupid %>% group_by(sex) %>% summarize(average_age = mean
```

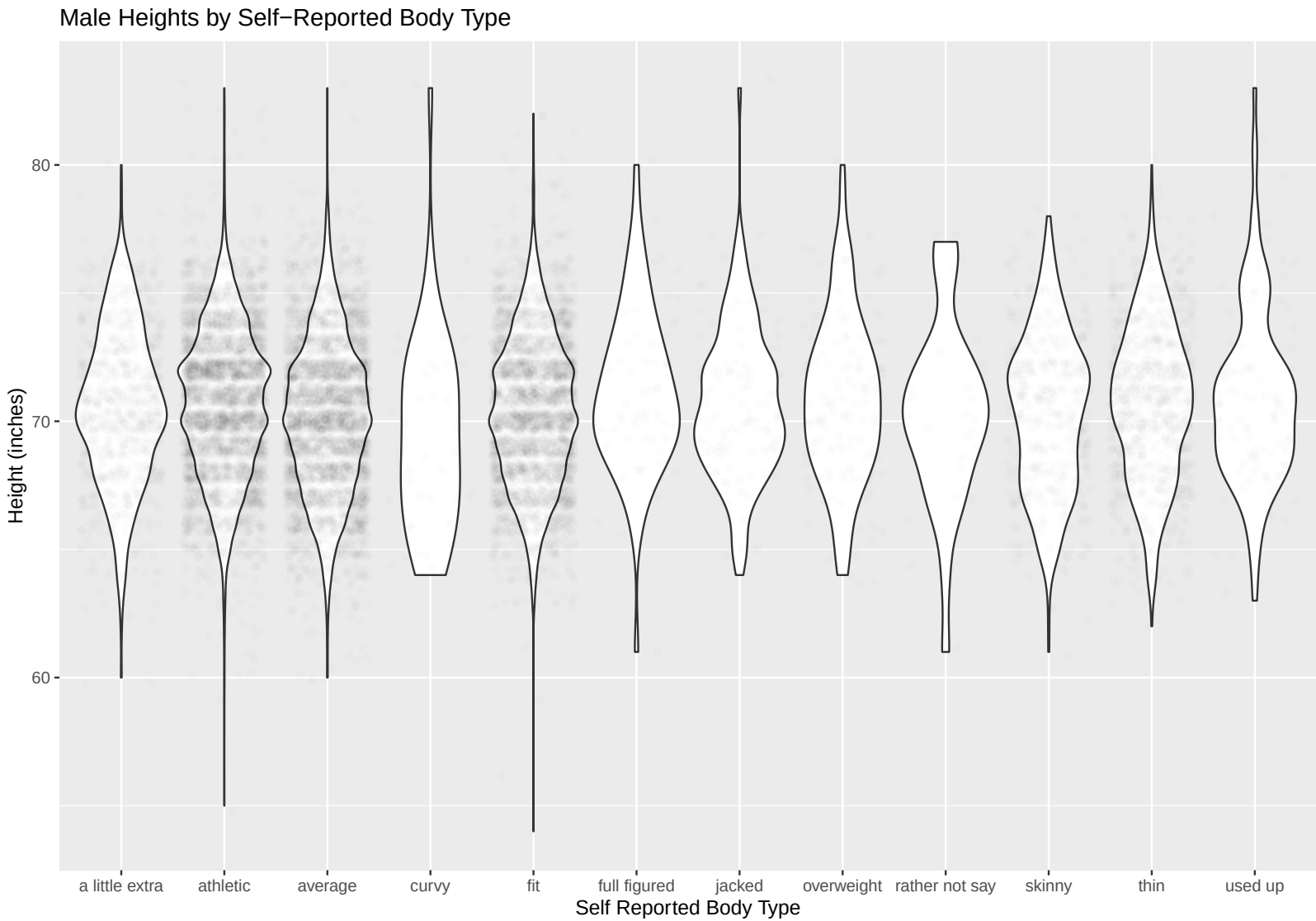
```
## # A tibble: 2 x 2
##   sex   average_age
##   <chr>       <dbl>
## 1 f            33.5
## 2 m            31.9
```

Graphics with ggplot2

- ggplot stands for the grammar of graphics and can be used to create figures in R.
- Layers of ggplot components are layered on top of each other using the + operator.

```
okcupid %>%
  filter(sex == 'm') %>%
  ggplot(aes(x = body_type, y = height)) +
  geom_violin() +
  ggtitle("Male Heights by Self-Reported Body Type") +
  xlab('Self Reported Body Type') +
  ylab('Height (inches)') +
  geom_jitter(alpha = .01)
```

Graphics with ggplot2



ggplot2 Overview

Why ggplot2?

- consistent underlying grammar of graphics (Wilkinson, 2005)
- plot specification at a high level of abstraction
- very flexible
- theme system for polishing plot appearance

Grammar of graphics

The basic idea: independently specify plot building blocks and combine them to create just about any kind of graphical display you want.

Building blocks of a graph include:

- data
- aesthetic mapping
- geometric object
- statistical transformations
- faceting

ggplot2 VS base R graphics

Compared to base graphics, ggplot2

- is more verbose for simple / canned graphics
- is less verbose for complex / custom graphics
- does not have methods (data should always be in a data.frame)
- uses a different system for adding plot elements

Aesthetic mapping

Aesthetics are things that you can see. Examples include:

- position (i.e., on the x and y axes)
- color ("outside" color)
- fill ("inside" color)
- shape (of points)
- linetype
- size

Aesthetic mappings are set with the `aes()` function.

Geometric objects (geom)

Geometric objects are the actual marks we put on a plot. Examples include:

- points (`geom_point`)
- lines (`geom_line`)
- boxplot (`geom_boxplot`)

A plot must have at least one geom; there is no upper limit. You can add a geom to a plot using the + operator.

Example: NCAA Basketball Data

NCAA basketball data

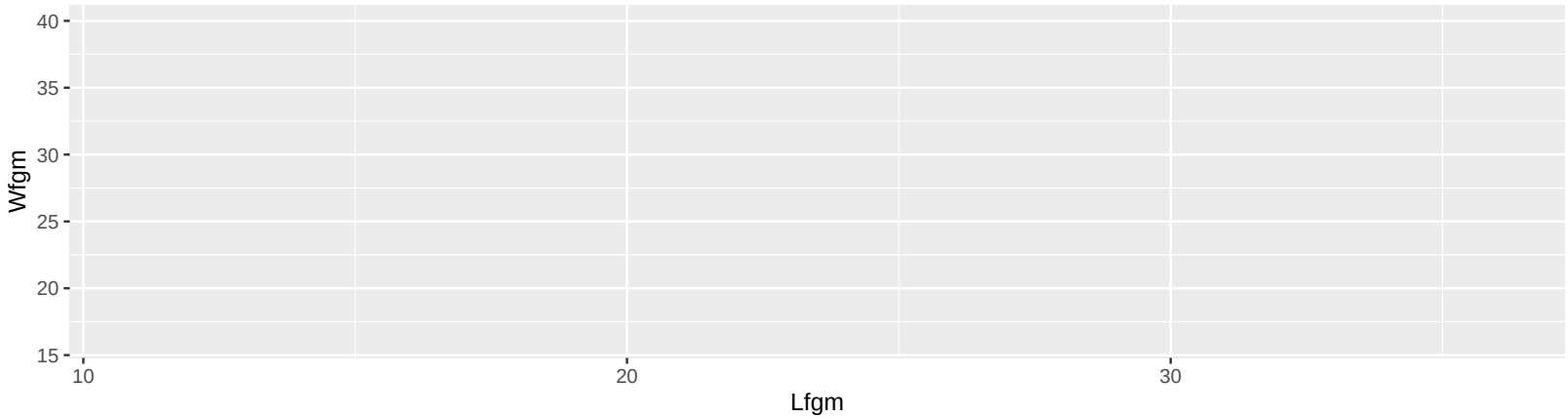
We will use data from the NCAA basketball tournament from 2011–2016.

```
hoops <- read_csv('https://math.montana.edu/shancock/data/TourneyDetaile
hoops_2011 <- hoops %>% filter(Season >= 2011)
hoops_2011
```

```
## # A tibble: 402 x 34
##    Season Daynum Wteam Wscore Lteam Lscore Wloc  Numot  Wfgm  Wfga Wf
##     <dbl>  <dbl> <dbl>  <dbl> <dbl>  <dbl> <chr> <dbl> <dbl> <dbl> <d
##  1   2011    134  1155     70  1412     52 N         0    26    50
##  2   2011    134  1421     81  1114     77 N         1    27    54
##  3   2011    135  1427     70  1106     61 N         0    23    54
##  4   2011    135  1433     59  1425     46 N         0    20    59
##  5   2011    136  1139     60  1330     58 N         0    22    54
##  6   2011    136  1140     74  1459     66 N         0    24    61
##  7   2011    136  1153     78  1281     63 N         0    29    54
##  8   2011    136  1163     81  1137     52 N         0    32    66
##  9   2011    136  1196     79  1364     51 N         0    29    53
## 10   2011    136  1211     86  1385     71 N         0    28    52
## # ... with 392 more rows, and 22 more variables: Wftm <dbl>, Wfta <db
## #   Wor <dbl>, Wdr <dbl>, Wast <dbl>, Wto <dbl>, Wstl <dbl>, Wblk <db
## #   Wpf <dbl>, Lfgm <dbl>, Lfga <dbl>, Lfgm3 <dbl>, Lfga3 <dbl>, Lftm
```

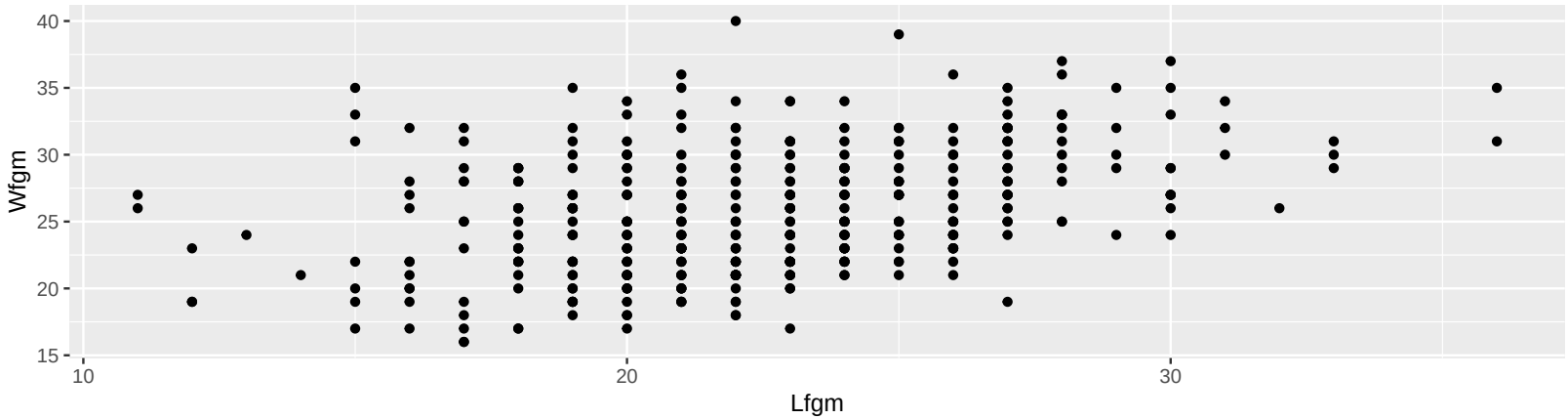
Graphical primitives: ggplot()


```
graph.a <- ggplot(data = hoops_2011, aes(Lfgm,Wfgm))
graph.a
```



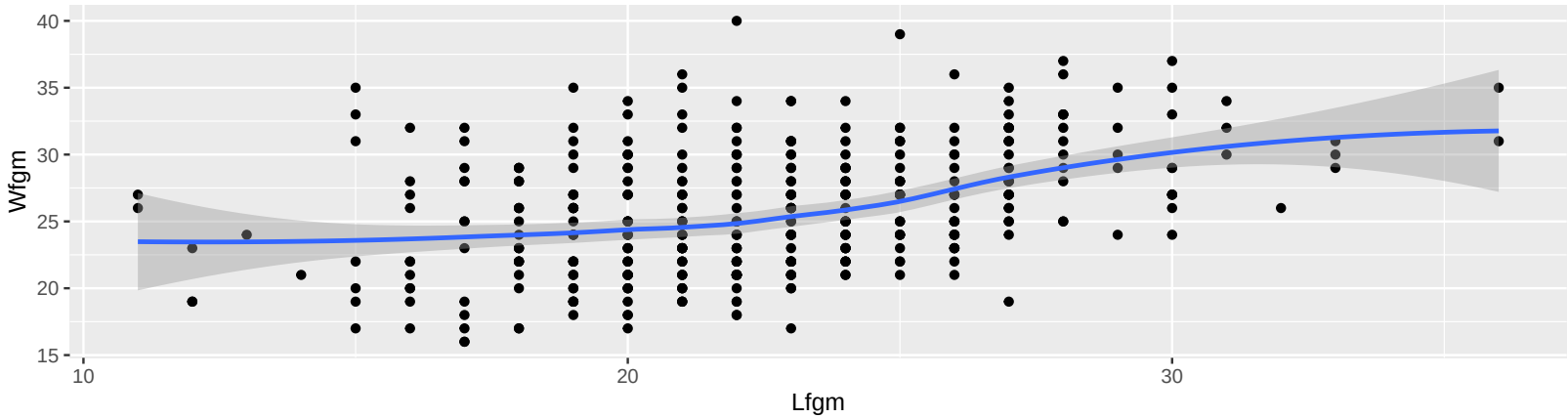
Adding geoms: geom_point()


```
graph.a + geom_point()
```



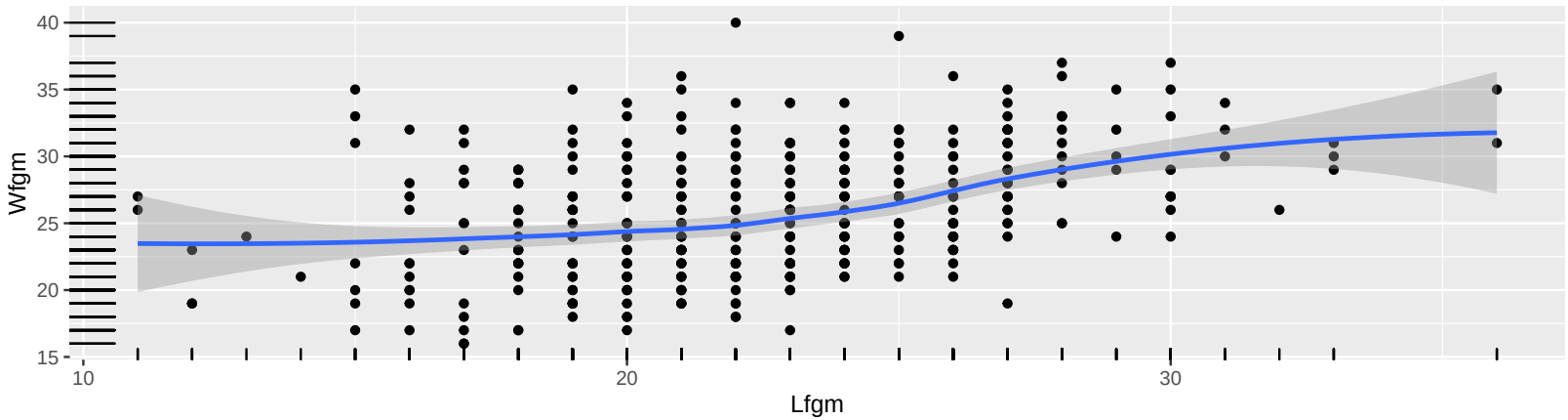
Adding geoms: geom_smooth()


```
graph.a + geom_point() +
  geom_smooth(method = 'loess', formula = 'y ~ x')
```



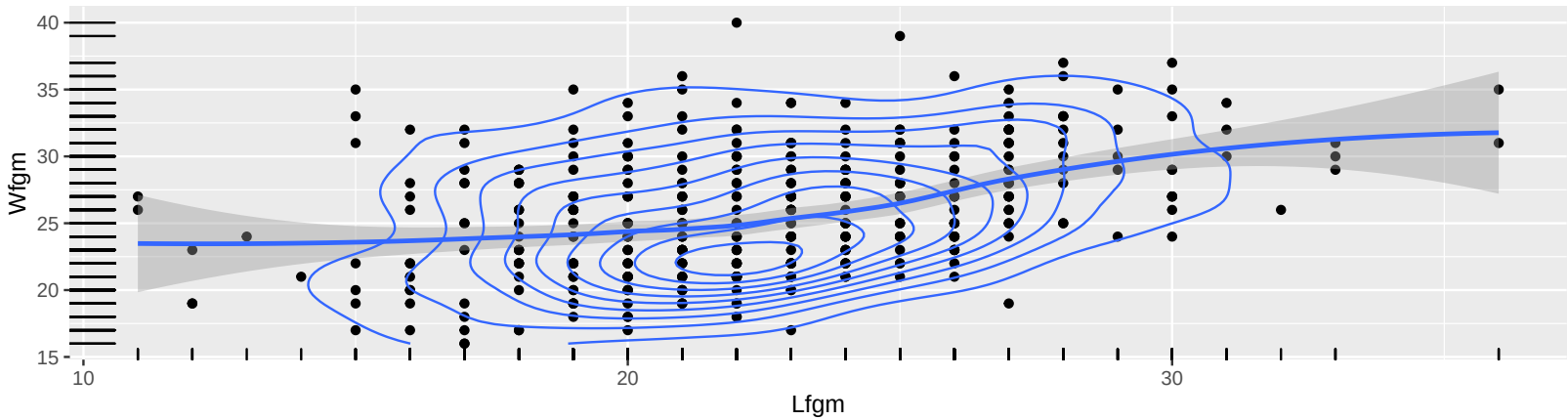
Adding geoms: geom_rug()


```
graph.a + geom_point() +
  geom_smooth(method = 'loess', formula = 'y ~ x') +
  geom_rug()
```



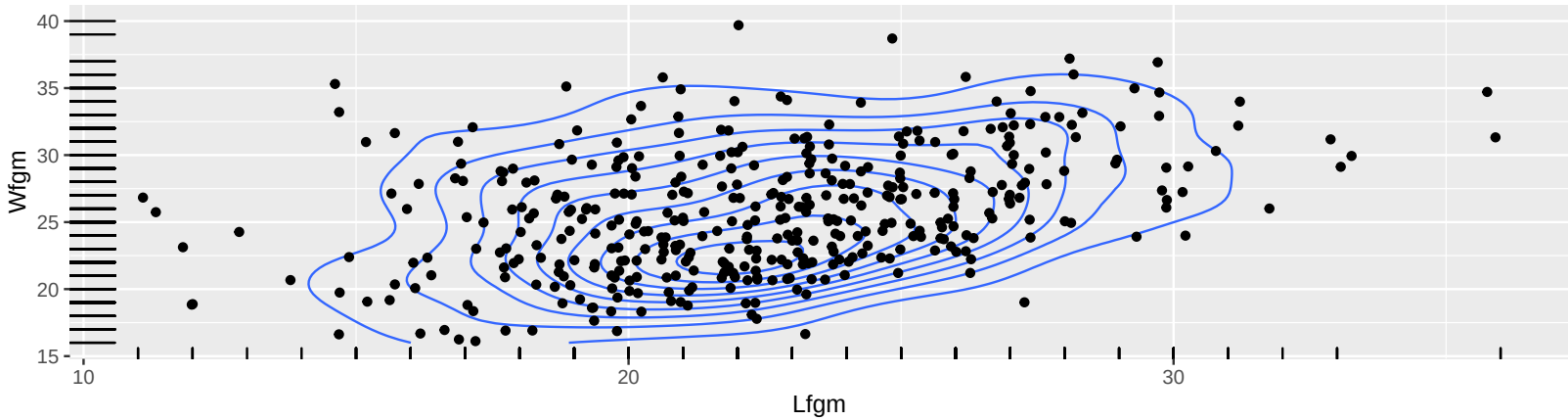
Adding geoms: geom_density2d()


```
graph.a + geom_point() +
  geom_smooth(method = 'loess', formula = 'y ~ x') +
  geom_rug() + geom_density2d()
```



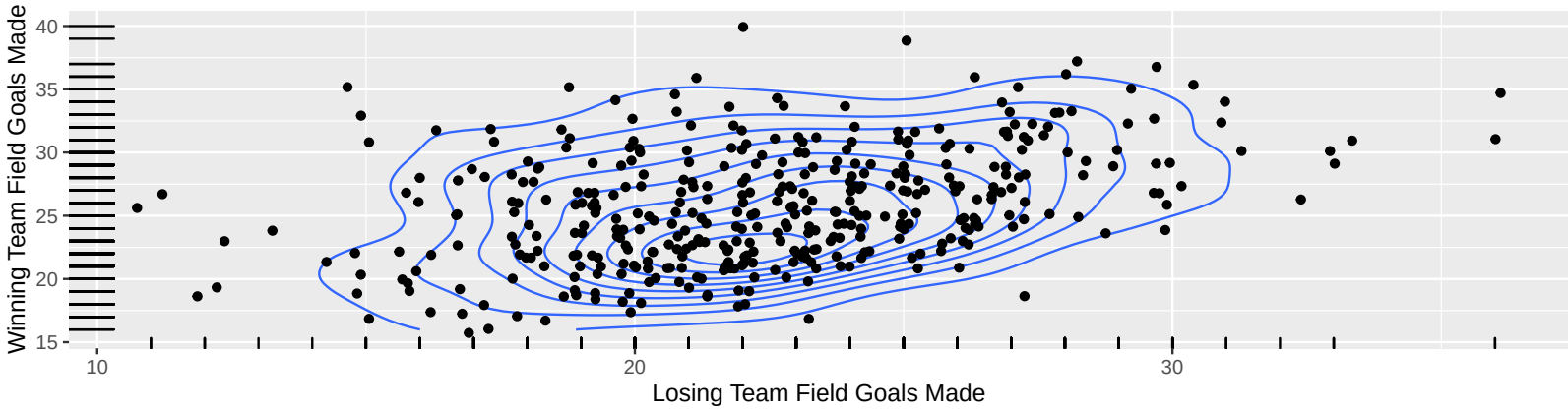
Adding geoms: geom_jitter()


```
graph.a + geom_rug() + geom_density2d() + geom_jitter
```



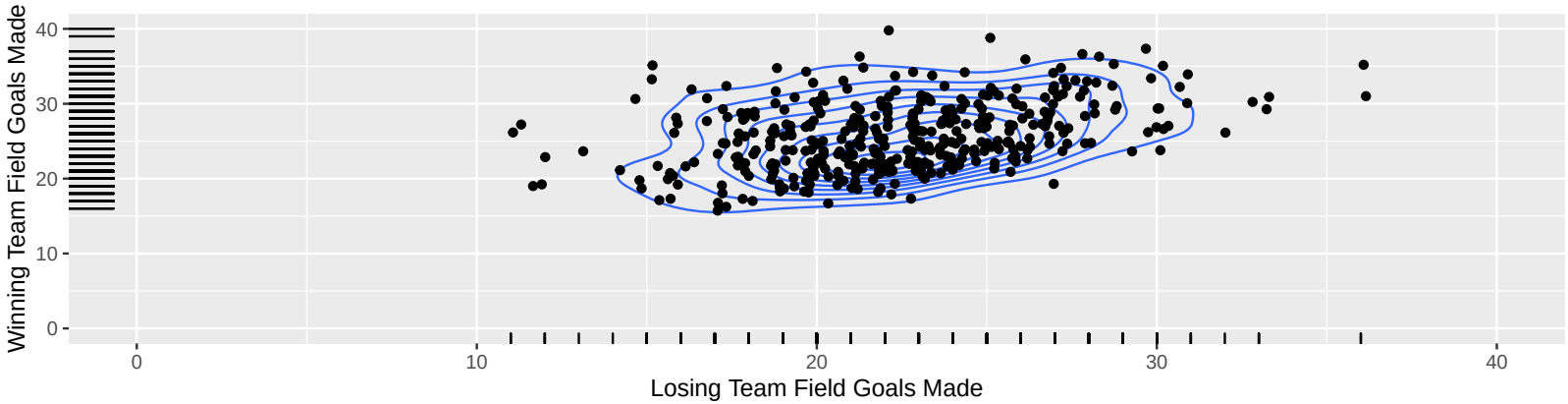
Adding geoms: `labs()`


```
graph.a  + geom_rug() + geom_density2d() +
 geom_jitter() +
  labs(x='Losing Team Field Goals Made',
       y = 'Winning Team Field Goals Made')
```



Scales: xlim() and ylim()

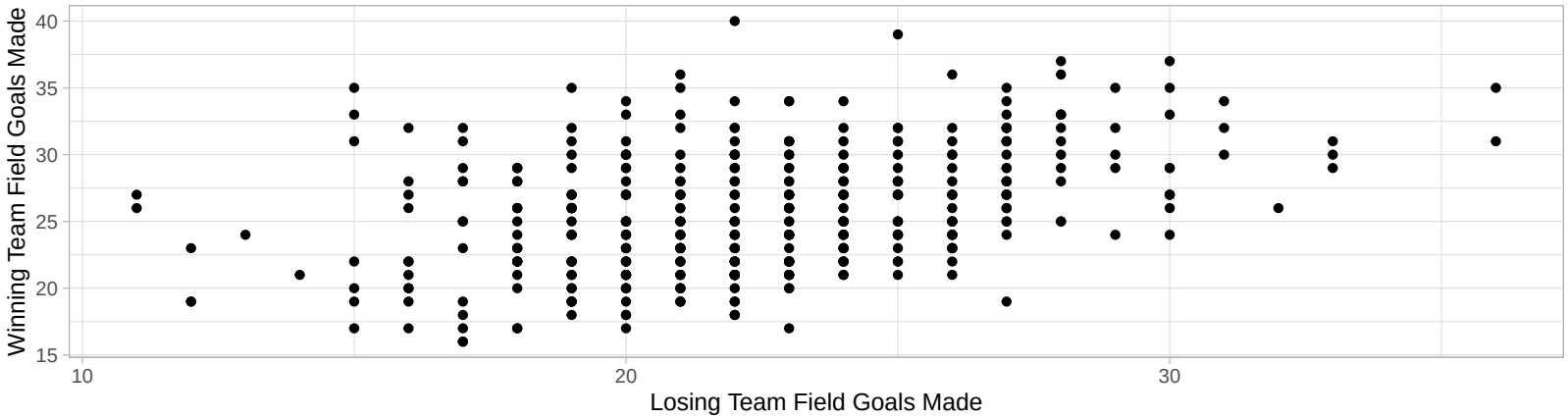

```
graph.a + geom_rug() + geom_density2d() +
 geom_jitter() +
  labs(x='Losing Team Field Goals Made',
       y = 'Winning Team Field Goals Made') +
  xlim(c(0,max(hoops_2011$Wfgm))) + ylim(c(0,max(hoop
```



Themes

Themes

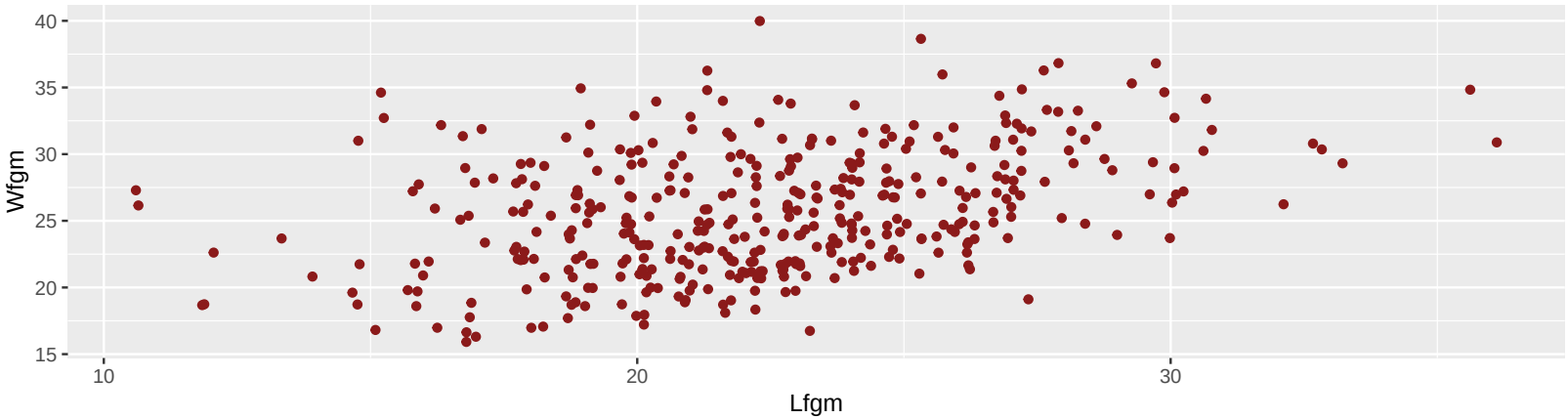
There are a wide range of themes available in ggplot: theme overview



More on aes

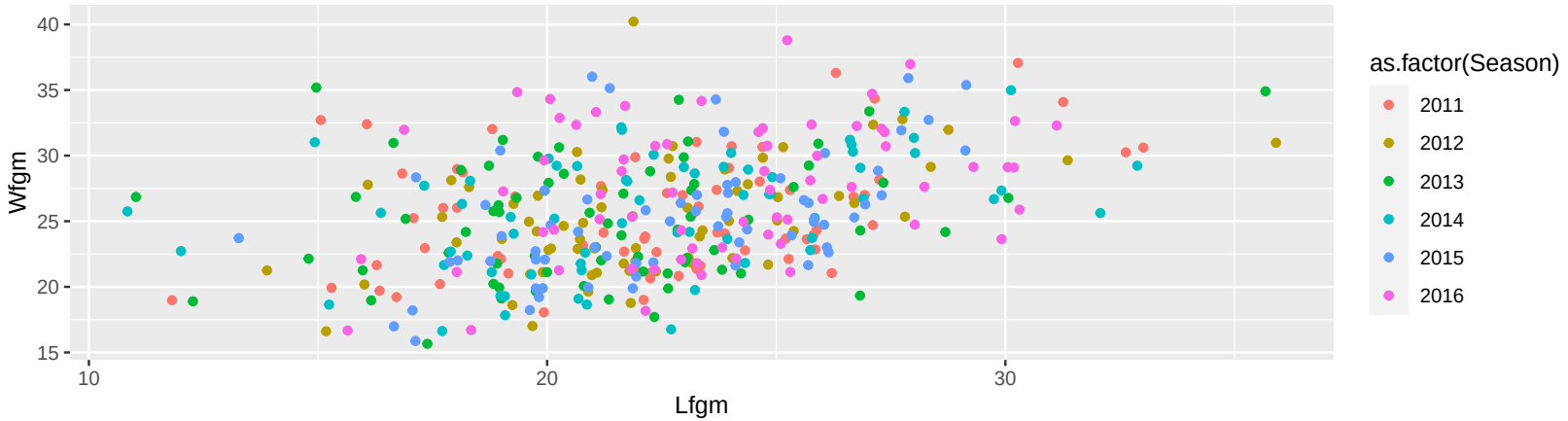
More about aes


```
graph.a + geom_jitter(col = 'firebrick4')
```



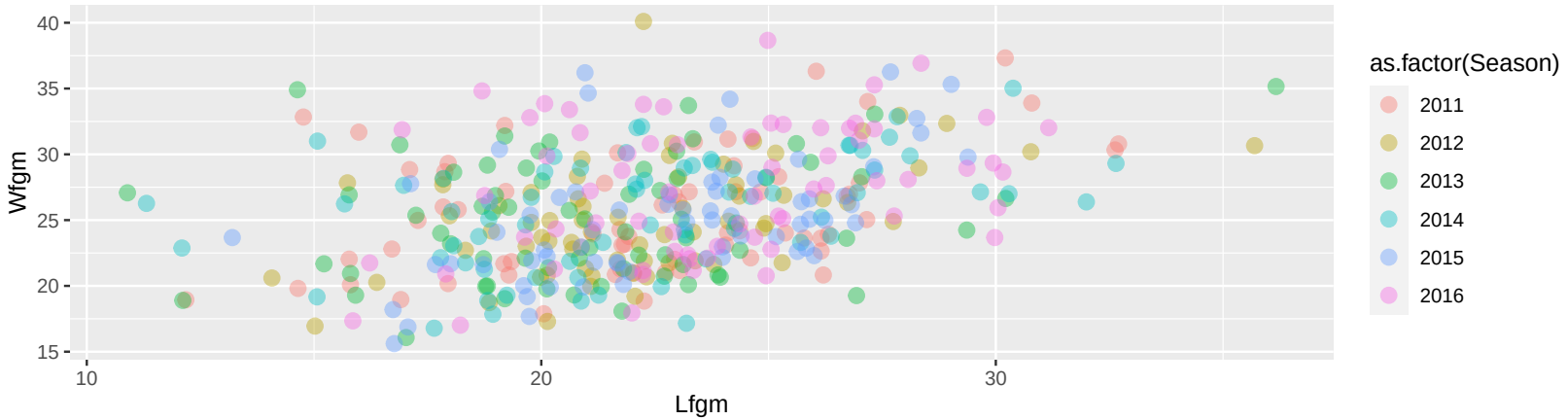
More about aes


```
graph.a + geom_jitter(aes(col = as.factor(Season)))
```

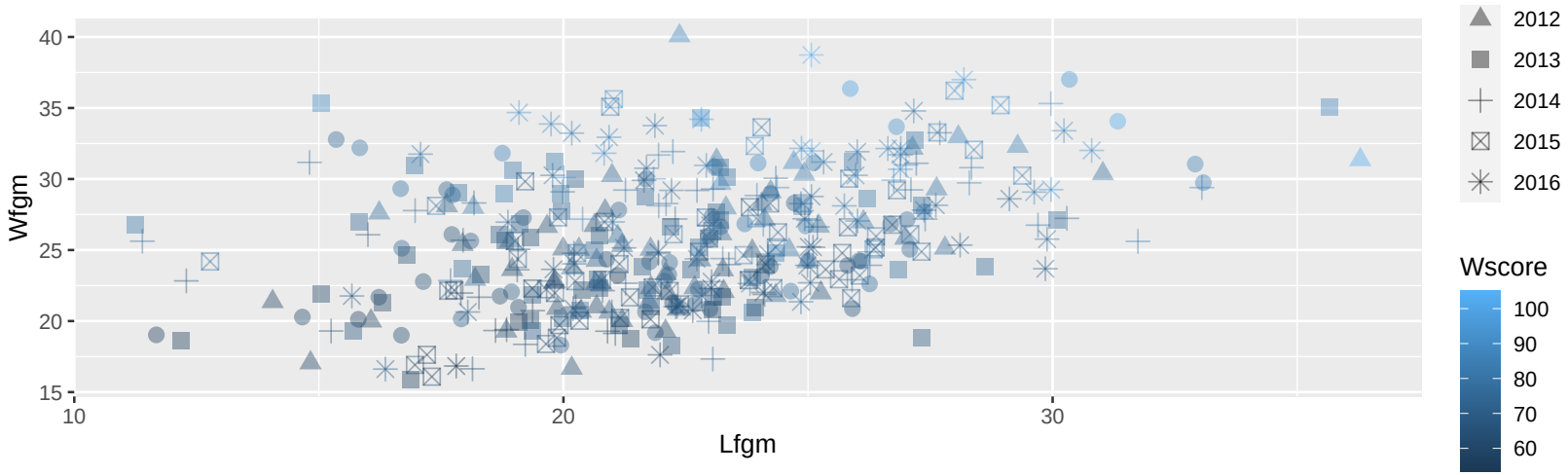


More about aes


```
graph.a + geom_jitter(aes(col = as.factor(Season)),
                      size=3, alpha=.4)
```



More about aes

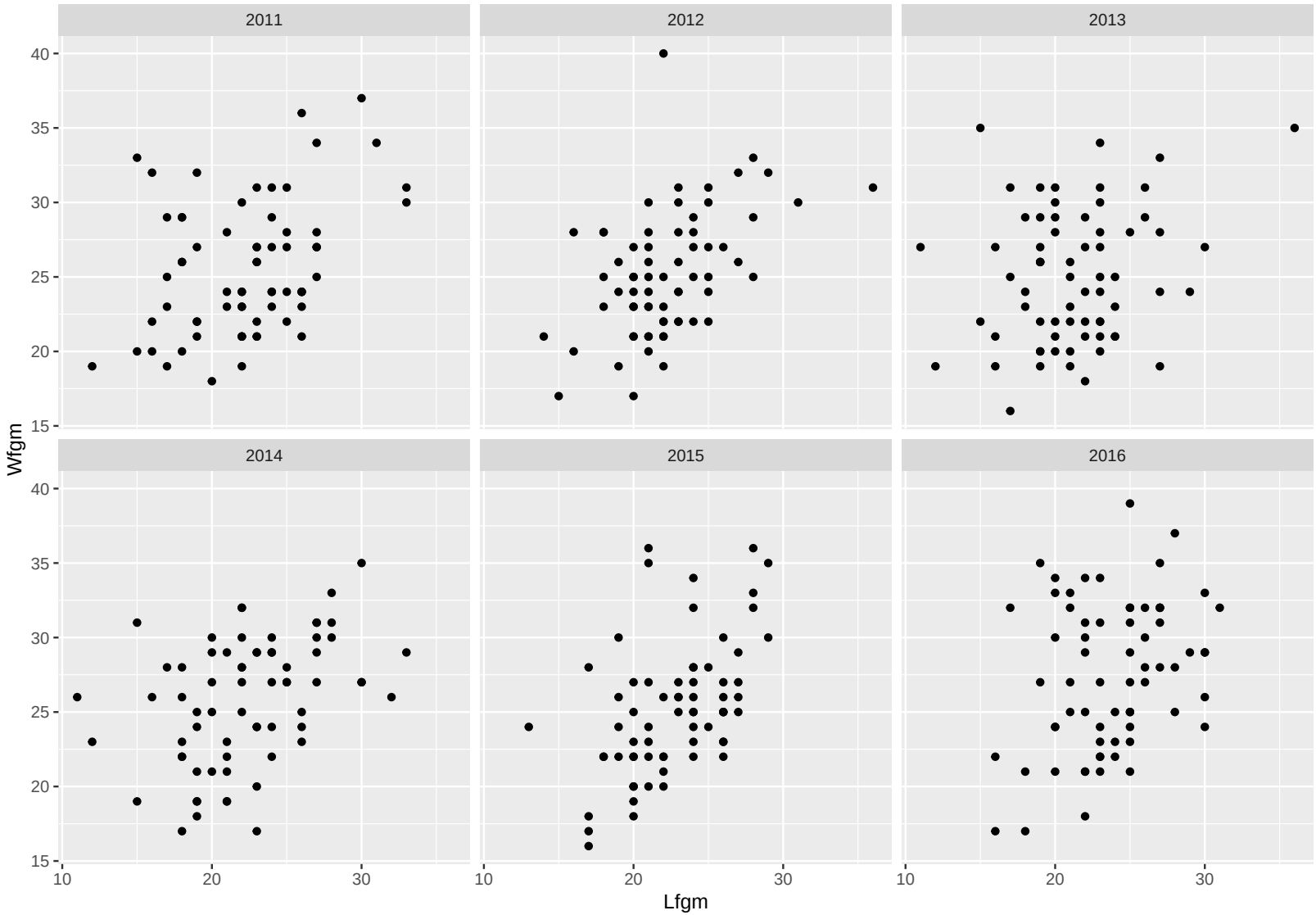


More about aes: code


```
graph.a +
  geom_jitter(aes(shape = as.factor(Season), col=Wsco
                  size=3, alpha=.4)
```

Faceting

Faceting

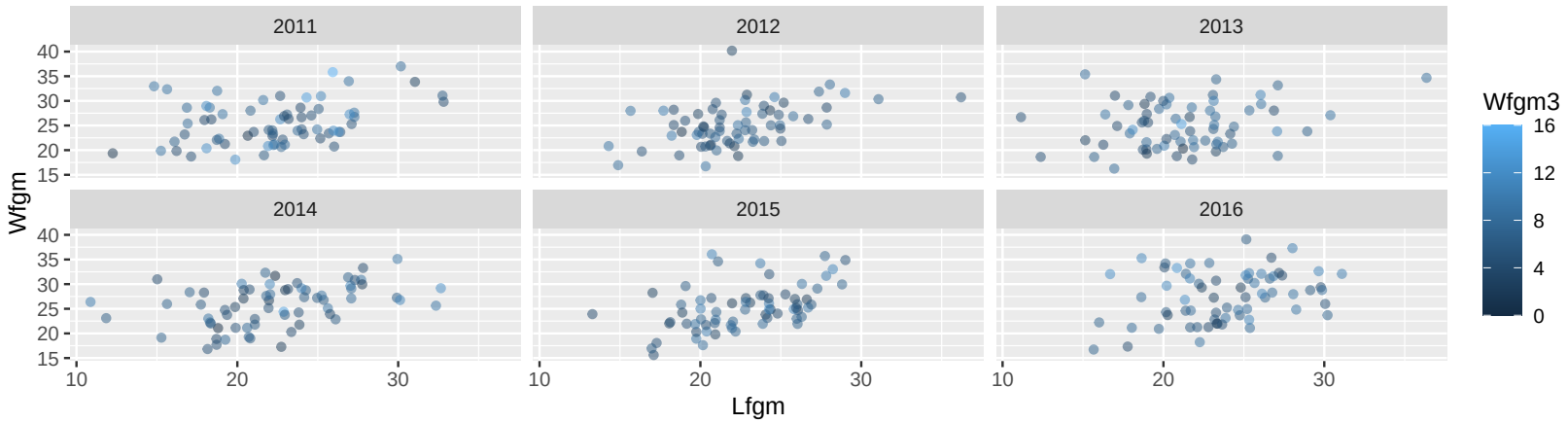


Faceting: code


```
graph.a + geom_point() + facet_wrap(~Season)
```

Faceting


```
graph.a + facet_wrap(~Season) +
  geom_jitter(alpha=.5, aes(color=Wfgm3))
```



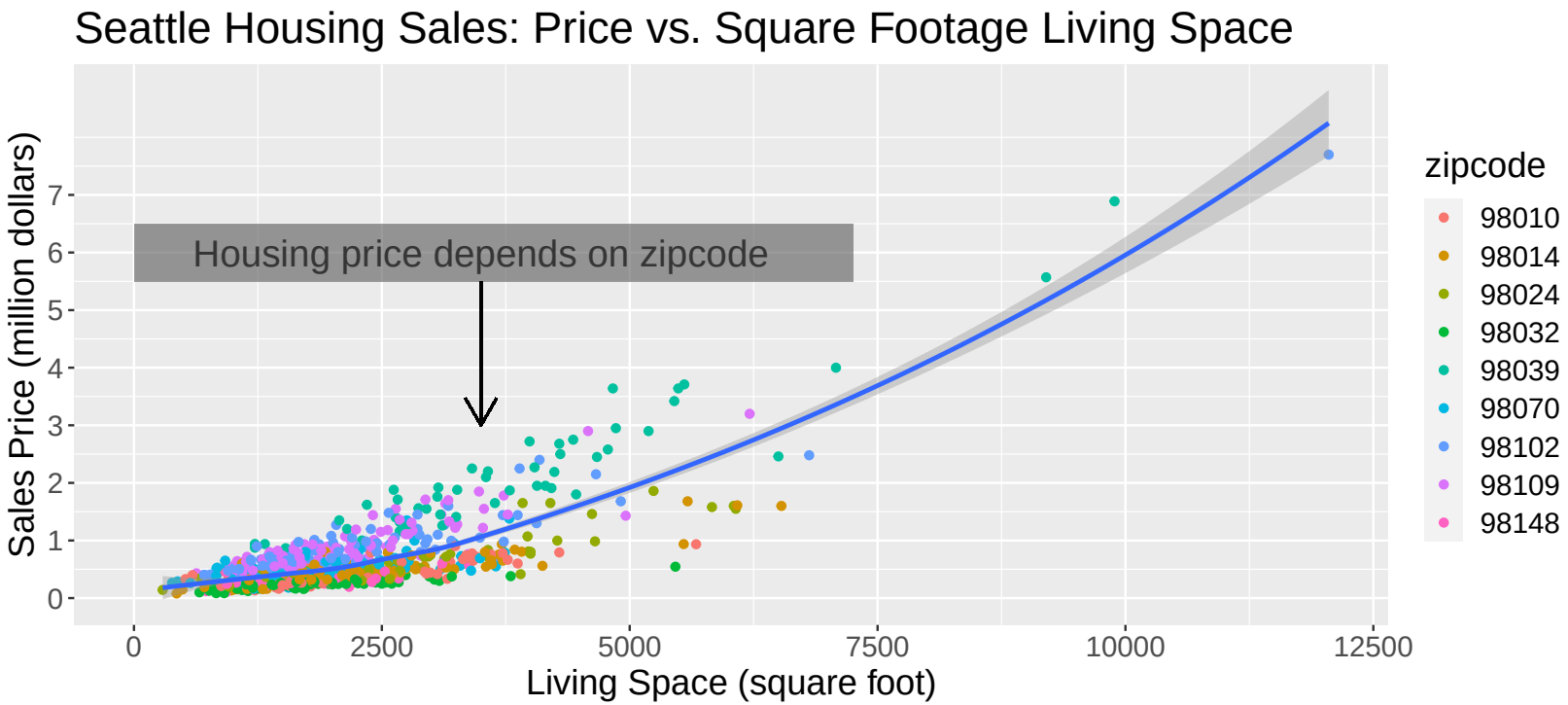
Your Turn: Seattle Housing Data

Seattle Housing Data Set

Use the Seattle Housing Data Set https://math.montana.edu/shancock/data/SeattleHousing.csv to create an interesting graphic, include informative titles, labels, and add an annotation.

```
seattle_in <-
read_csv('https://math.montana.edu/shancock/data/SeattleHousing.csv')
```

Example solution



Example solution


```
seattle_in$zipcode <- as.factor(seattle_in$zipcode)
graph.a <- ggplot(data = seattle_in, aes(sqft_living,price))
graph.a + geom_jitter(aes(col = zipcode))+
  theme(plot.title = element_text(size=20),
        text = element_text(size=16)) + geom_smooth(method='loess')+
  ggtitle('Seattle Housing Sales:
          Price vs. Square Footage Living Space') +
  ylab('Sales Price (million dollars)') +
  xlab('Living Space (square foot)')+
  scale_y_continuous(breaks=c(seq(0,7000000,by=1000000)),
                     labels=as.character(0:7)) +
  annotate('text',3500,6000000,
           label = 'Housing price depends on zipcode', size=6) +
  annotate("rect", xmin = 0, xmax = 7250,
           ymin = 5500000, ymax = 6500000, alpha = .6) +
  geom_segment(aes(x=3500, xend=3500, y=5500000, yend=3000000),
               arrow = arrow(length = unit(0.5, "cm")))
```