Data Science Capstone Final Report - Yelp Dataset Analysis

Kelvin Tan

November 22, 2015

Introduction

The Yelp dataset consists of information on businesses, reviews, and users. Business attributes include location, categories, average rating, and review count. Review attributes include rating, text of the review, and votes for useful, funny, and cool. User attributes include review count, average rating, and votes received. Running queries on this information to provide a visual overview of search results as well as an overview of the neighborhoods, which will aid the making a well-informed decision.

Business

1. What businesses have the highest rating as a function of average star rating and total review count?
2. Which businesses are closed? Which neighborhoods have the highest percentage of closed businesses?
3. What are the most popular neighborhoods for bars? Coffee shops?

Users

1. Who are the funniest reviewers? Most useful reviewers? 'Coolest' reviewers?
2. Which users have written the most reviews in the greater Phoenix area?

Reviews

1. What are the top funny reviews in each area/category? Useful? Cool?
2. Which categories have the greatest total count of reviews? Highest average?

Methods and Data

Getting Data

Data is from the Yelp dataset challenge. User GetData.R to download and prepare the data into data frame.

The data was originally in JSON form, in a total of 5 datasets connected by identifiers. The data contained reviews from businesses in 10 cities around the world. The 6 cities in the USA were selected to be analyzed to remove reviews in other languages, and other dialects of English. We will also limit the analysis to food service businesses, as the features of the reviews would not be similar for different kinds of businesses. Using the function 'grep' the variable 'categories of business' was searched for restaurants and business serving food.

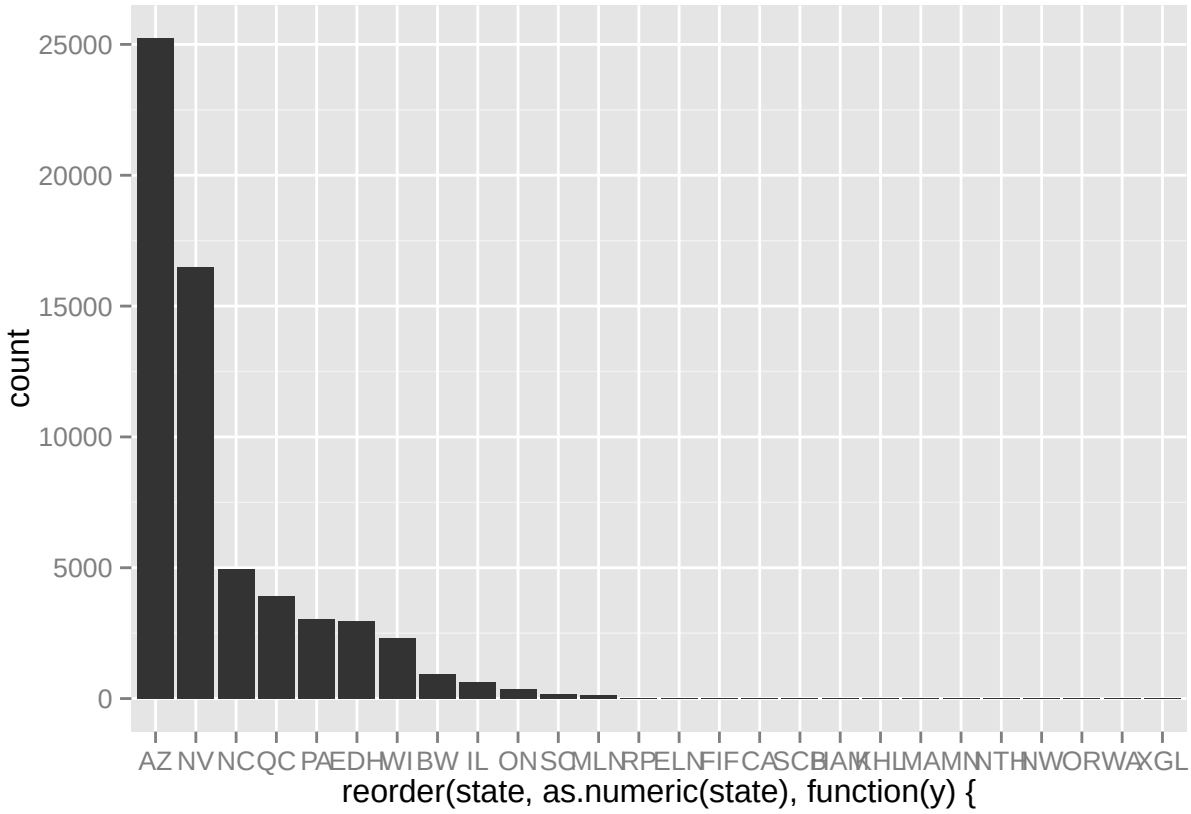
```
# votes data
review_votes <- rbind(reviews$votes)
review_votes$review_id <- reviews$review_id
reviews$date <- ymd(reviews$date)
```

```r
reviews <- reviews[!names(reviews) %in% c("votes", "type")]

# checkin data
checkin_info <- rbind(checkins$checkin_info)
checkin_info$business_id <- checkins$business_id
checkins <- checkin_info
rm(checkin_info)

# business data
business_attributes <- rbind(business$attributes)
business_attributes$business_id <- business$business_id

#attributes$`Accepts Credit Cards` <- unlist(attributes$`Accepts Credit Cards`)
business <- select(business, -c(type))

#bus_attributes <- business$attributes
business <- business[, names(business) != "attributes"]

# all glory and honor to @hrbrmstr
kludge <- function(i) {
              data.frame(id=business[i,]$business_id,
              add_rownames(do.call(rbind.data.frame, business[i,]$hours), "day"),
              stringsAsFactors=FALSE)}

business_hours <- bind_rows(pblapply(1:nrow(business), kludge))
business <- business[, names(business) != "hours"]

# tips data
tips$date <- ymd(tips$date)
tips <- tips[!names(tips) %in% c("type")]

# users data
users$yelping_since <- ymd(paste0(users$yelping_since, "-01"))
compliments <- users$compliments
users <- users[,names(users) != "compliments"]
compliments$user_id <- users$user_id

#clean up names in our compliments data_table
names(compliments) <- gsub("\\.", "_", names(compliments))
```

Results

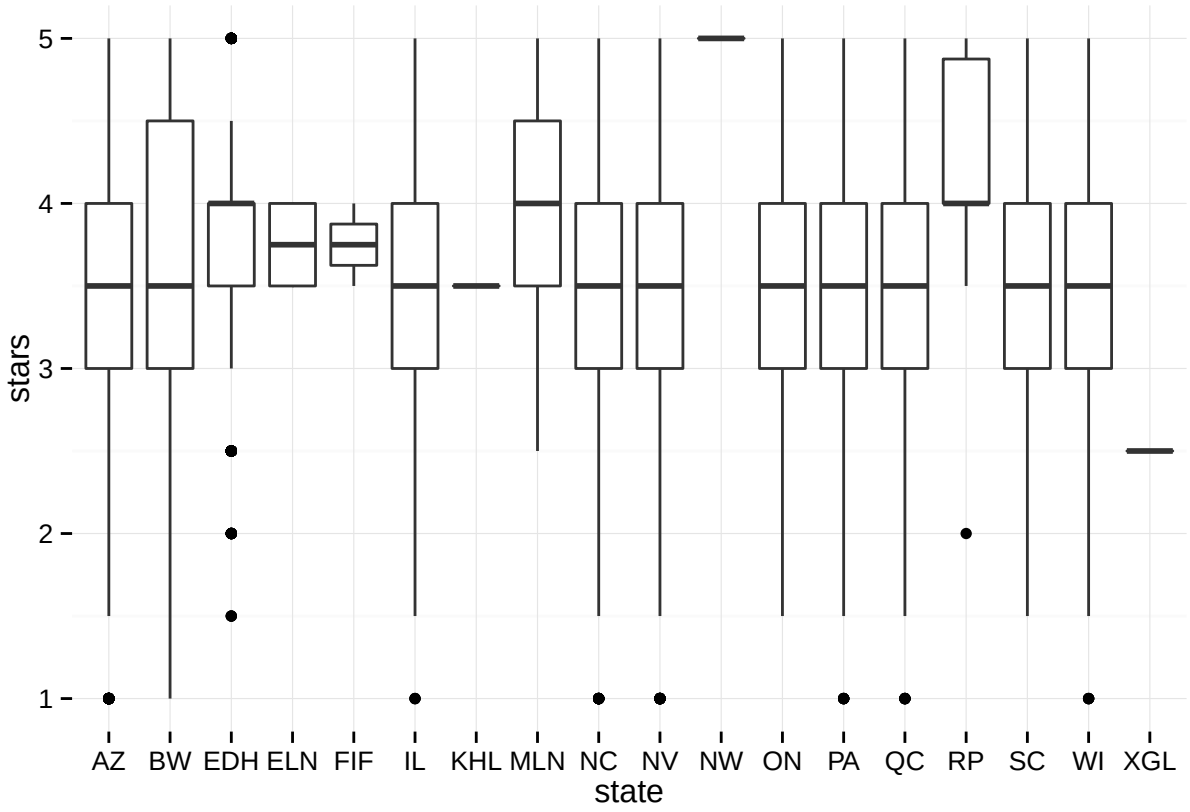
Business

Lots of data for a few states. AZ is particularly heavy.



Review

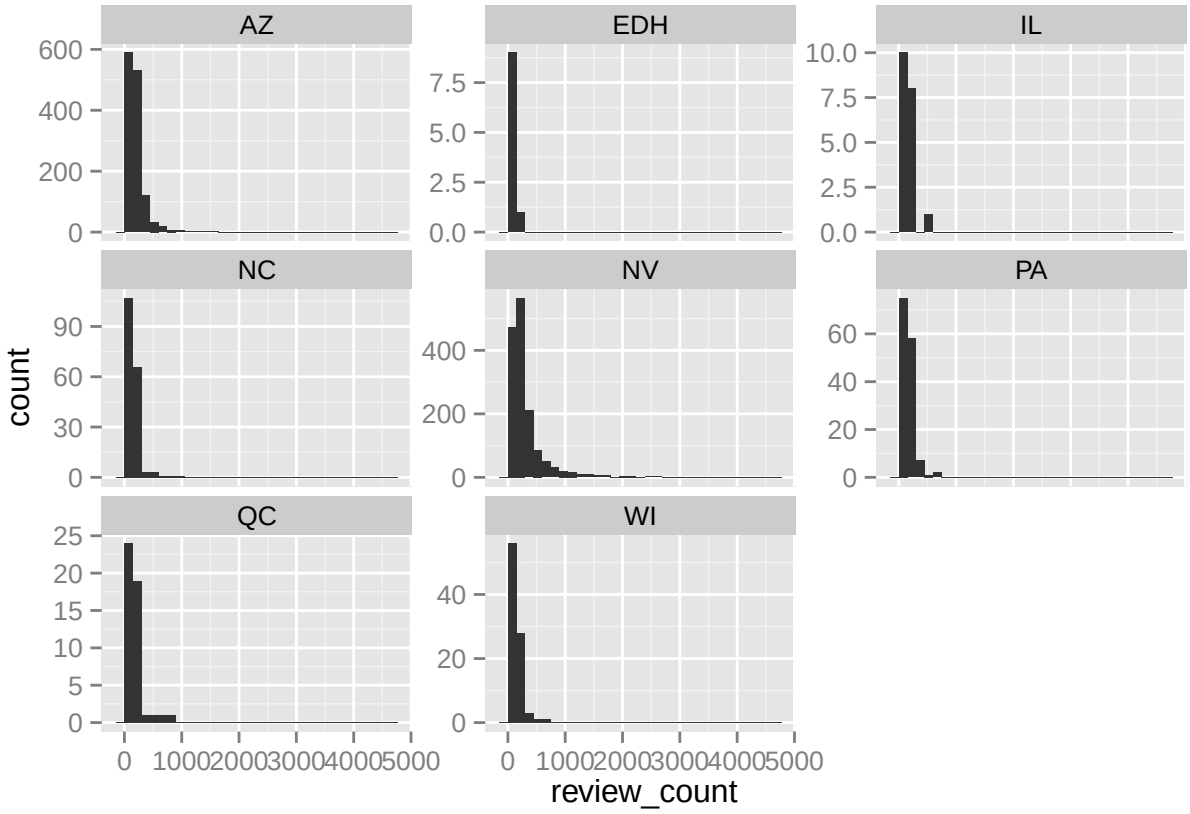
Rating of restaurant.



User

What are the most rating location associated with restaurants?

```
business[business$review_count > 100,] %>% ggplot(aes(x=review_count)) + geom_histogram() + facet_wrap(
```



Discussion

Given the subjectivity and oversimplification of the data, the model is quite good as a rough predictor. A much more robust model can potentially be created using all aspects of the raw dataset

Appendix

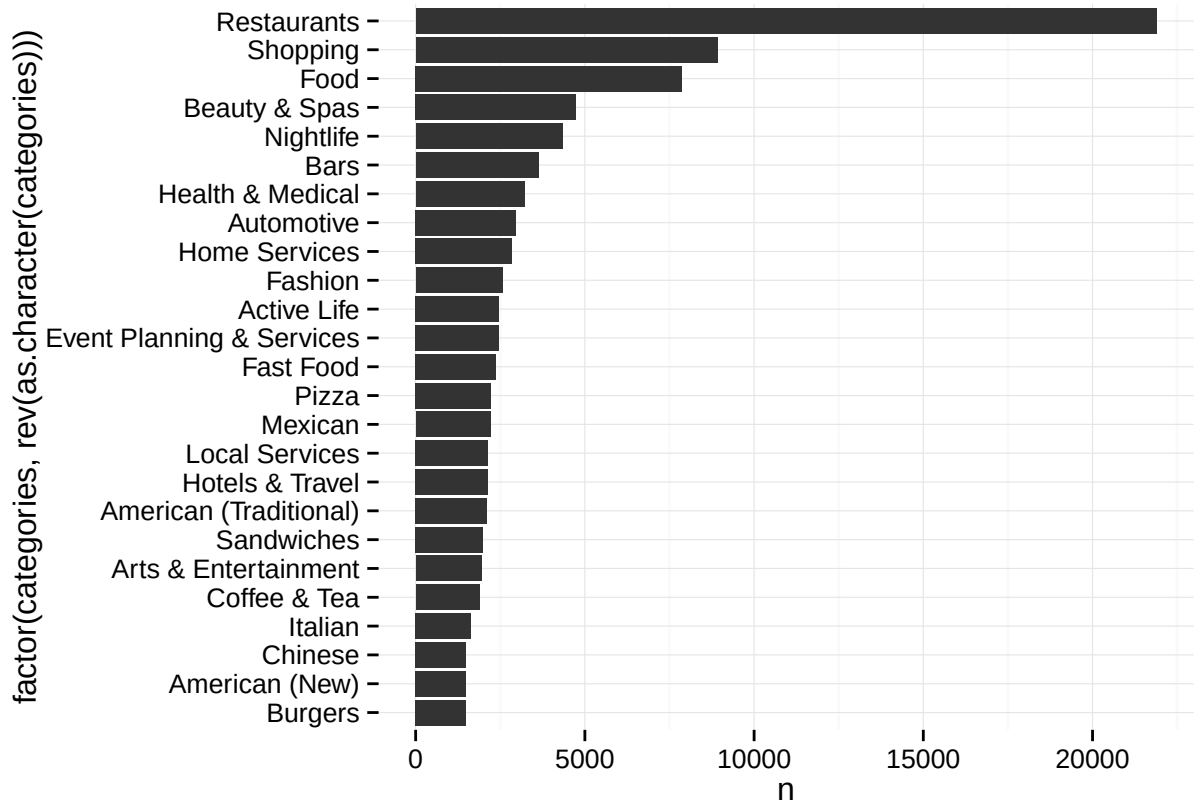
What are the most common business categories?

```
data.frame(categories = unlist(business$categories)) %>%
  group_by(categories)  %>% tally()  %>%
  arrange(desc(n)) %>% top_n(10)
```

```
## Source: local data frame [10 x 2]
##
##          categories     n
##              (fctr) (int)
## 1       Restaurants 21892
## 2          Shopping  8919
## 3              Food  7862
## 4     Beauty & Spas  4738
## 5         Nightlife  4340
```

```
## 6              Bars  3628
## 7  Health & Medical  3213
## 8        Automotive  2965
## 9     Home Services  2853
## 10          Fashion  2566
```

```
data.frame(categories = unlist(business$categories)) %>%
  group_by(categories)  %>% tally()  %>%
  arrange(desc(n)) %>% top_n(25) %>% ggplot() -> gg
gg + geom_bar(aes(x=factor(categories, rev(as.character(categories))), y=n),
              stat="identity") + coord_flip() + theme_minimal()
```



What are the most rating categories associated with restaurants?

```
dat <- business[business$restaurants==TRUE,]
data.frame(categories = unlist(dat$categories)) %>%
  filter(categories != "Restaurants") %>%
  group_by(categories)  %>% tally()  %>%
  arrange(desc(n)) %>% top_n(20)
```

```
## Source: local data frame [20 x 2]
##
##                categories     n
##                    (fctr) (int)
## 1               Fast Food  2383
```

```
## 2                   Pizza  2223
## 3                 Mexican  2208
## 4  American (Traditional)  2113
## 5               Nightlife  2045
## 6              Sandwiches  1981
## 7                    Bars  1934
## 8                    Food  1807
## 9                 Italian  1633
## 10                Chinese  1496
## 11         American (New)  1494
## 12                Burgers  1481
## 13     Breakfast & Brunch  1116
## 14                  Cafes   776
## 15               Japanese   746
## 16             Sushi Bars   671
## 17                  Delis   649
## 18                Seafood   554
## 19            Steakhouses   554
## 20          Chicken Wings   516
```

```
table(business[business$longitude>-10,c("state")])
```

```
##
##   BW  EDH  ELN  FIF  HAM  KHL  MLN  NTH   NW   RP  SCB  XGL
##  934 2971   10    4    1    1  123    1    1   13    3    1
```

```
review_votes %>% mutate(votes=(funny + useful+cool)) %>% arrange(desc(votes)) %>% head()
```

```
##   funny useful cool              review_id votes
## 1   141    166  137 5w_yWEGVXVEYYbnD0Roi6g   444
## 2     0    126  126 NzyNSNBjT9gJ2Df2xBk00w   252
## 3    62     96   89 Q1FB2G18Y_5YSJseCC2AUg   247
## 4    80     70   72 hJjaVNRY_g_srMqwhlVfAA   222
## 5     1    107  107 3gjdy4FVgAWpHX6uvu4MHg   215
## 6     0    102  100 xM5miTAPNL4XiNdl_yedpQ   202
```

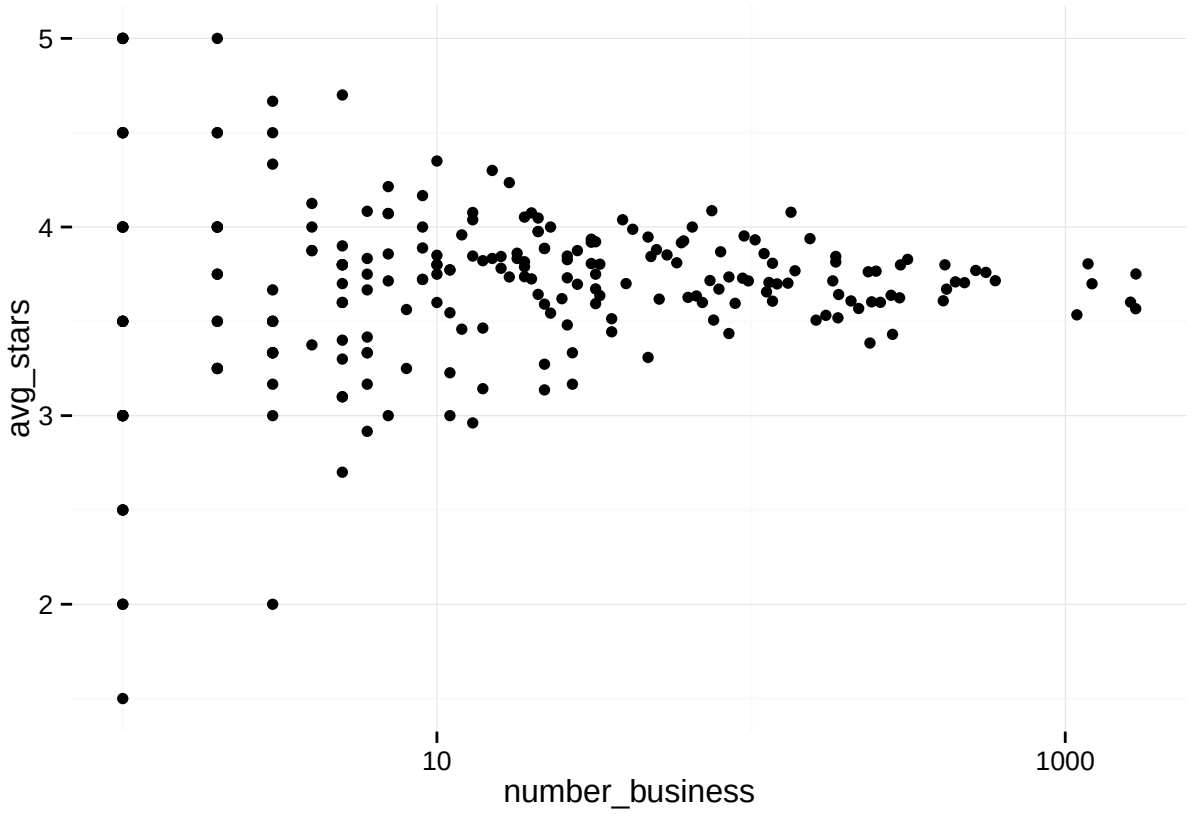
```
reviews[reviews$review_id=="5w_yWEGVXVEYYbnD0Roi6g",]$text
```

```
## [1] "I'm the first real person to review this place, let all other fake spammers be gone!  Yelp shou
```

```
business[,c("neighborhoods", "stars")] %>% as_data_frame %>% mutate(neighborhoods=as.character(neighbor
```

```
## Source: local data frame [1 x 2]
##
##   neighborhoods avg_stars
##           (chr)     (dbl)
## 1     The Strip  3.566826
```

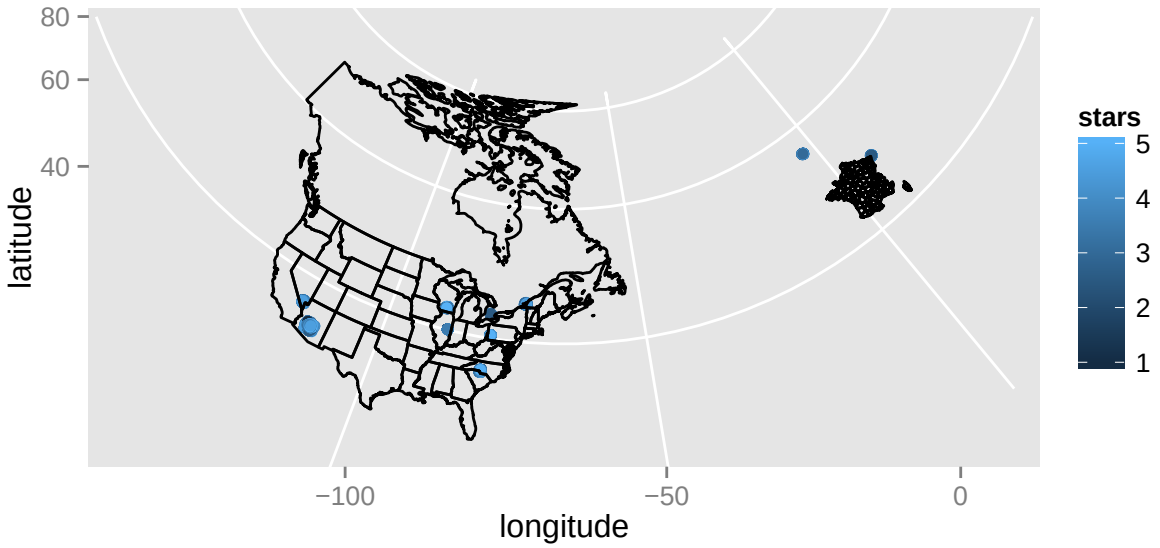
```
business[,c("neighborhoods", "stars")] %>% as_data_frame %>% mutate(neighborhoods=as.character(neighbor

dat %>% filter(neighborhoods!="character(0)") ->dat2
ggplot(dat2, aes(x=number_business, y=avg_stars)) + geom_point() + theme_minimal() + scale_x_log10()
```



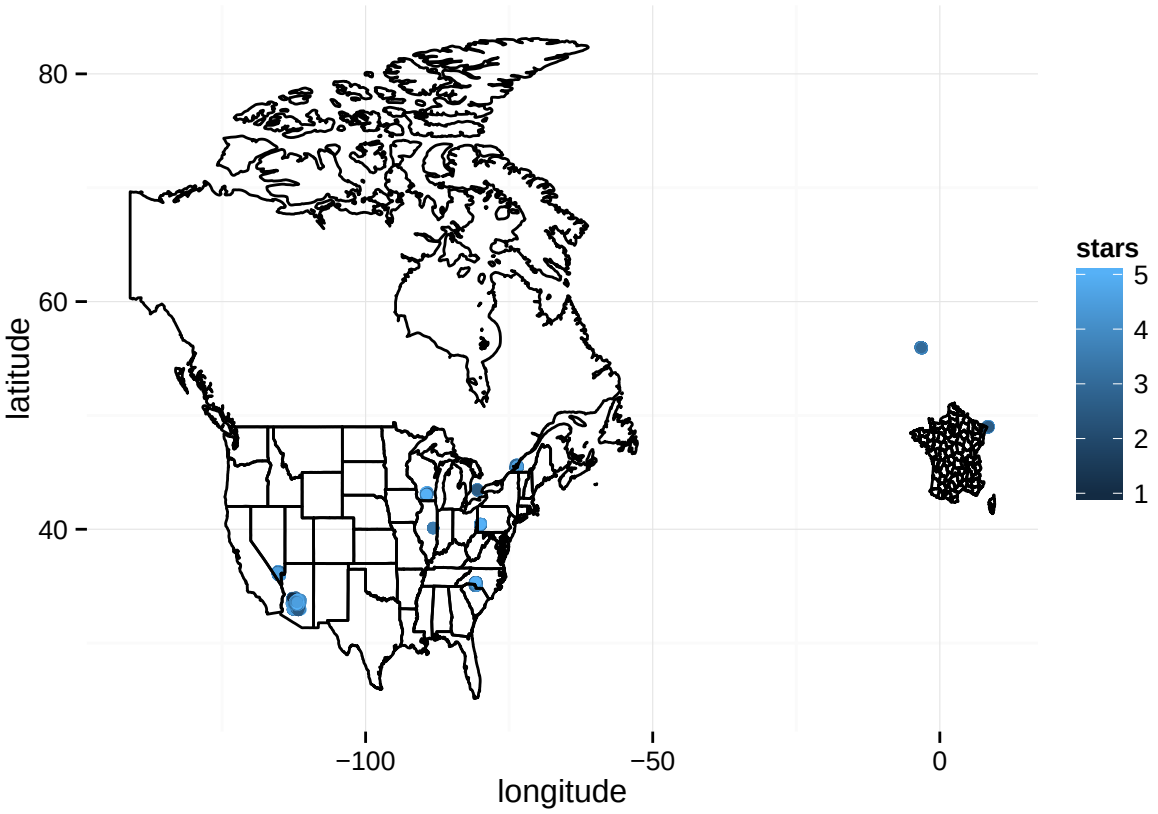

```
library(tidyr)
gather(checkins, timeframe, occurances, -business_id) -> tall_checkins
```

User check-in location.

```
library(maps)
library(mapproj)
us <- map_data("state")
ca <- map_data("world", "Canada")
fr <- map_data("france")

xlim = c(-110,-100)
ylim = c(40,60)
dat_grid = expand.grid(x = xlim[1]:xlim[2], y = ylim[1]:ylim[2])
dat_grid$z = runif(nrow(dat_grid))

gg <- ggplot()
gg <- gg + geom_point(data=business, aes(x=longitude, y=latitude, color=stars))
gg <- gg + geom_tile() +
  geom_polygon(data=us, aes(x=long, y=lat, group=group), colour="black", fill="white", alpha=0) +
  geom_polygon(data=ca, aes(x=long, y=lat, group=group), colour="black", fill="white", alpha=0) +
  geom_polygon(data=fr, aes(x=long, y=lat, group=group), colour="black", fill="white", alpha=0)
gg + coord_map("albers", at0= 45.5, lat1=29.5)
```




```
gg <- gg + theme_minimal()
gg
```


80
60
40
latitude
−100
−50
0
longitude
stars
5
4
3
2
1