Distribution of Farmers Markets

Martin Oberg

04/03/2021

Introduction

This document reports on trends of Farmers Markets accessibility throughout the United States using data collected from https://www.kaggle.com/madeleineferguson/farmers-markets-in-the-united-states. This is a data set that requires date cleaning and the main aim is project to provide some code for how to do so. I will then compare market accessibility as measured by number of days that markets are open; having many markets is beneficial to consumers, but if they are only open for a few weeks then it is not reasonable to think that consumers could buy a significant portion of their yearly food at Farmers Markets.

I will also compute the number of markets per capita and compare results.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
data_path = "D:/data/farmers-markets-in-the-united-states/farmers_markets_from_usda.csv"
county_path = "D:/data/farmers-markets-in-the-united-states/wiki_county_info.csv"

data_cols = cols(
  .default = col_character(),
  FMID = col_integer(),
```

```
  Season4Date = col_character(),
  Season4Time = col_character(),
  x = col_double(),
  y = col_double()
)

data_full = read_csv(data_path, col_types = data_cols)
cleaned_data = FALSE
data_county = read_csv(county_path)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   number = col_character(),
##   county = col_character(),
##   State = col_character(),
##   `per capita income` = col_character(),
##   `median household income` = col_character(),
##   `median family income` = col_character(),
##   population = col_number(),
##   `number of households` = col_number()
## )
```

Data Cleaning

There must have been a lot of manual data entry for this data set because there are a lot of non-standard date entries. On first glance of the column names one would suspect that Season1-4 corresponds to seasons throughout the year and the various dates and times of market hours. This might be the case for some markets, however, Season1Date is filled in for most markets and date often extend for more than a year. I will make the assumption that market dates that extend beyond a year are simply year round markets without intermittent closures as there is no evidence to the contrary. I will also be ignoring columns Season2-4 as the purpose of those columns is unclear.

Another analysis could look at how many markets offer which kinds of products. That will be left for a different report.

This is a look at the kind of data we have.

```
## # A tibble: 5 x 4
##      FMID County    State          Season1Date
##     <int> <chr>     <chr>          <chr>
## 1 1018261 Caledonia Vermont        06/14/2017 to 08/30/2017
## 2 1018318 Cuyahoga  Ohio           06/24/2017 to 09/30/2017
## 3 1009364 Pickens   South Carolina <NA>
## 4 1010691 Barton    Missouri       04/02/2014 to 11/30/2014
## 5 1002454 New York  New York       July to November
```

The following code fixes problems in data entry.

```
# There are quite a few different formats that dates are recorded in.  Some misspellings, some extra te
# This will fix them for the date parser.
if (!cleaned_data) {
```

```
  data_full[data_full$FMID == 1011963, "Season1Date"] = str_replace(data_full[data_full$FMID == 1011963
  data_full[data_full$FMID == 1008935, "Season1Date"] %>%
    map_chr(~ str_replace(.x, "Start Date ","")) %>%
    map_chr(~ str_replace(.x, "End Date ","")) -> data_full[data_full$FMID == 1008935, "Season1Date"]
  data_full[data_full$FMID == 1010153, "Season1Date"] %>%
    map_chr(~ str_replace(.x, "Octobsr", "October")) -> data_full[data_full$FMID == 1010153, "Season1Da
  data_full[data_full$FMID == 1001139, "Season1Date"] %>%
    map_chr(~ str_replace(.x, "Sept", "September")) -> data_full[data_full$FMID == 1001139, "Season1Dat
  data_full$Season1Date %>%
    map_chr(~ str_replace(.x, "09/31", "09/30")) -> data_full$Season1Date
  data_full$Season1Date %>%
    map_chr(~ str_replace(.x, "11/31", "11/30")) -> data_full$Season1Date
  data_full$Season1Date %>%
    map_chr(~ str_replace(.x, "04/31", "04/30")) -> data_full$Season1Date
  data_full[data_full$FMID == 1011963, "Season1Date"] = "June 23, 2012 to September 8, 2012"
  data_full[data_full$FMID == 1008935, "Season1Date"] = "01/01/2013 to 12/31/2013"
  data_full[data_full$FMID == 1004852, "Season1Date"] = "June 30, 2011 to October 13, 2011"
}
# There are som string replace functions that are called that should only be run once!
cleaned_data = TRUE
```

And now we can calculate how long each market is open.

```
data_good_dates = data_full %>%
  select(FMID, State, County, Season1Date) %>%
  separate(Season1Date, into = c("SeasonBeg", "SeasonEnd"), sep = " to", fill = "right") %>%
  filter(!is.na(SeasonBeg)) %>%
  filter(!is.na(SeasonEnd)) %>%
  filter(nchar(SeasonEnd) > 1 )

data_good_dates %>%
  mutate(begDate = case_when( !str_detect(SeasonBeg, regex("[:alpha:]")) ~ parse_date_time2(SeasonBeg,
                              !str_detect(SeasonBeg, regex("[:digit:]")) ~ parse_date_time(SeasonBeg, o
                              TRUE ~ parse_date_time(SeasonBeg, orders = c("b %d, %Y", "b %d, %Y", "b %
                              )) -> markets_by_days
```

```
## Warning: Problem with `mutate()` input `begDate`.
## i  4830 failed to parse.
## i Input `begDate` is `case_when(...)`.

## Warning: 4830 failed to parse.

## Warning: Problem with `mutate()` input `begDate`.
## i  733 failed to parse.
## i Input `begDate` is `case_when(...)`.

## Warning: 733 failed to parse.
```

```
markets_by_days %>%
  mutate(endDate = case_when( !str_detect(SeasonEnd, regex("[:alpha:]")) ~ parse_date_time2(SeasonEnd,
                              !str_detect(SeasonEnd, regex("[:digit:]")) ~ parse_date_time(SeasonEnd, o
                              TRUE ~ parse_date_time(SeasonEnd, orders = c("b %d, %Y", "b %d, %Y", "b %
                              )) -> markets_by_days
```

```
## Warning: Problem with `mutate()` input `endDate`.
## i  4827 failed to parse.
## i Input `endDate` is `case_when(...)`.

## Warning: 4827 failed to parse.

## Warning: Problem with `mutate()` input `endDate`.
## i  736 failed to parse.
## i Input `endDate` is `case_when(...)`.

## Warning: 736 failed to parse.
```

```
# Fix missing years in begDate
markets_by_days %>%
  mutate(begDate = if_else(year(begDate) == 0, update(begDate, year = year(endDate)), begDate)) %>%
  mutate(DaysOpen = map2_int(begDate, endDate, ~ as.integer(.y-.x))) -> markets_by_days
# Fix missing years in endDate
markets_by_days %>%
  mutate(endDate = if_else(year(endDate) == 0, update(endDate, year = year(begDate)), endDate)) %>%
  mutate(DaysOpen = map2_int(begDate, endDate, ~ as.integer(.y-.x))) -> markets_by_days
  #filter(abs(nDays) > 366)

# Some markets have mdy for SeasonBeg, but only month for SeasonEnd.  This
markets_by_days %>%
  #filter(nDays < 0)
  mutate(endDate = if_else(month(begDate) > month(endDate),
                           update(endDate, year = year(begDate) + 1),
                           endDate)) %>%
  mutate(DaysOpen = map2_int(begDate, endDate, ~ as.integer(.y-.x))) -> markets_by_days

# only outliers now are DoysOpen > 365.
markets_by_days %>%
  filter(DaysOpen <0)
```

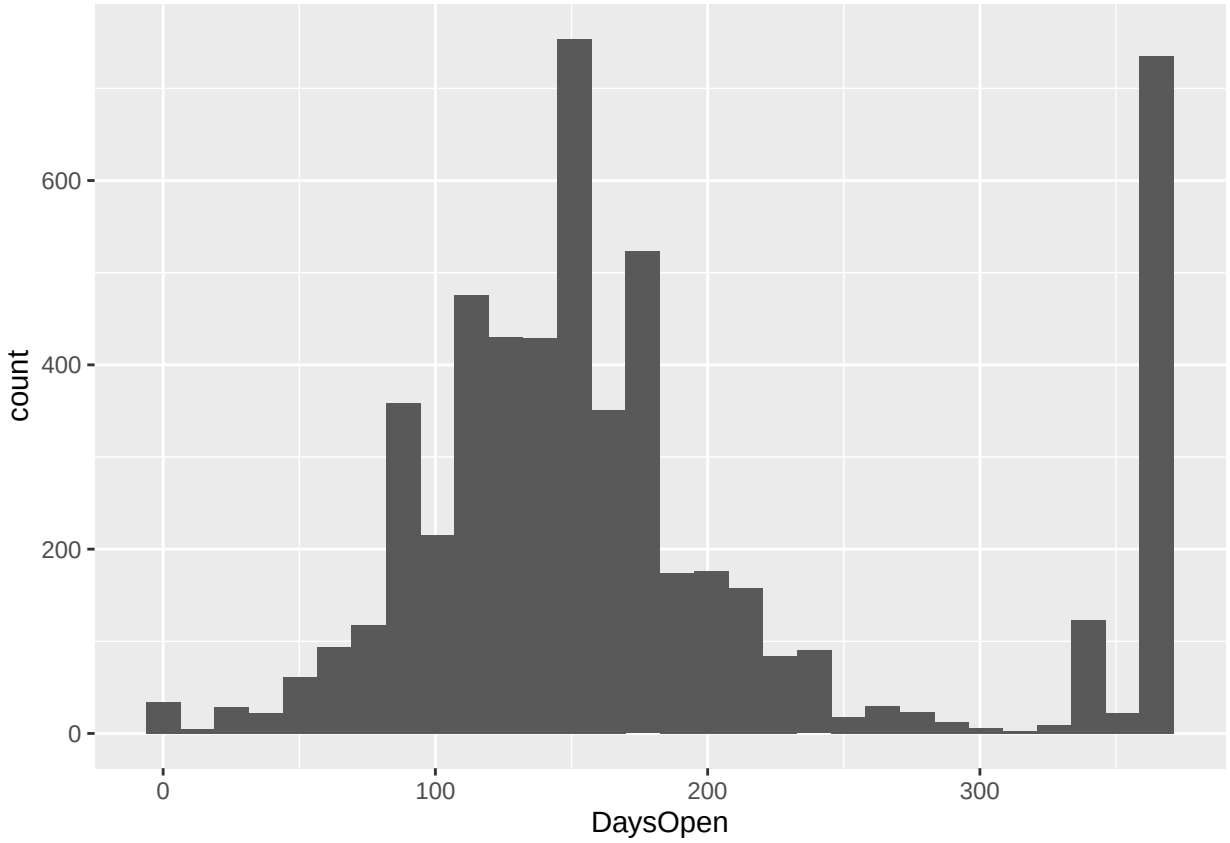
```
## # A tibble: 0 x 8
## # ... with 8 variables: FMID <int>, State <chr>, County <chr>, SeasonBeg <chr>,
## #   SeasonEnd <chr>, begDate <dttm>, endDate <dttm>, DaysOpen <int>
```

```
#markets open for more than a year will be considered to be open for a full year
markets_by_days %>%
  mutate(DaysOpen = if_else(DaysOpen > 365, as.integer(365), DaysOpen)) -> markets_by_days

head(markets_by_days)
```

```
## # A tibble: 6 x 8
##     FMID State County SeasonBeg SeasonEnd begDate
##    <int> <chr> <chr>  <chr>     <chr>     <dttm>
## 1 1.02e6 Verm~ Caled~ 06/14/20~ " 08/30/~ 2017-06-14 00:00:00
## 2 1.02e6 Ohio  Cuyah~ 06/24/20~ " 09/30/~ 2017-06-24 00:00:00
## 3 1.01e6 Miss~ Barton 04/02/20~ " 11/30/~ 2014-04-02 00:00:00
## 4 1.00e6 New ~ New Y~ July      " Novemb~ 0000-07-01 00:00:00
## 5 1.01e6 Tenn~ David~ 05/05/20~ " 10/27/~ 2015-05-05 00:00:00
## 6 1.01e6 New ~ New Y~ 06/10/20~ " 11/25/~ 2014-06-10 00:00:00
## # ... with 2 more variables: endDate <dttm>, DaysOpen <int>
```

```
markets_by_days = markets_by_days %>%
  select(FMID, State, County, DaysOpen)

markets_by_days %>%
  ggplot(aes(x=DaysOpen)) +
  geom_histogram()
```

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




```
markets_by_days %>% nrow()
```

```
## [1] 5563
```

```
markets_by_days %>% filter(DaysOpen >= 350) %>% nrow() / 5563
```

```
## [1] 0.1357181
```

```
markets_by_days %>% filter(DaysOpen < 350) %>% select(DaysOpen) %>% summary()
```

```
##     DaysOpen
##  Min.   :  0
##  1st Qu.:119
```

```
##  Median :147
##  Mean   :149
##  3rd Qu.:175
##  Max.   :349
```

Roughly 13% of markets are open year-round, or nearly year round. The rest are normally distributed around an average of 148 days, about 5 months.

```
library(tmap)
library(spData)
library(spDataLarge)
data(us_states)

state_population =
  data_county %>%
  group_by(State) %>%
  summarise(pop = sum(population))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
norm_days = markets_by_days %>%
  group_by(State) %>%
  summarise(TotalDays = sum(DaysOpen %% 366, na.rm=T)) %>%
  right_join(state_population) %>%
  mutate(normDays = TotalDays/pop*1000)
```

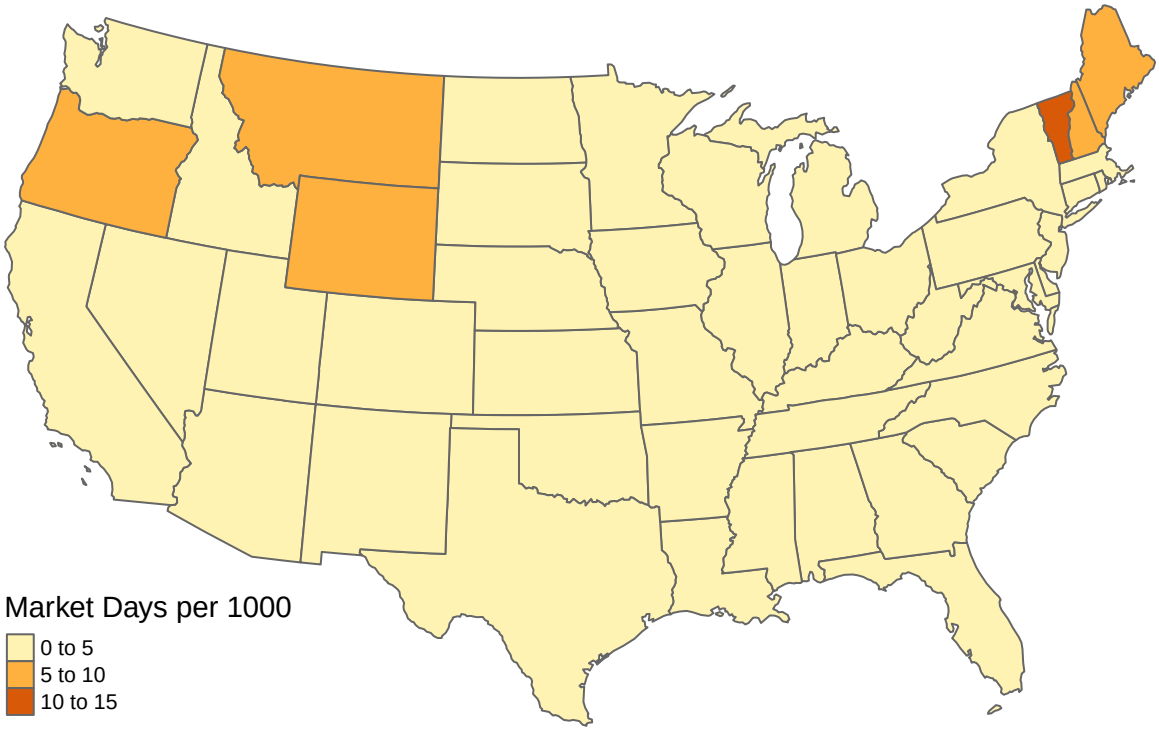
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## Joining, by = "State"
```

```
norm_days_states =
  us_states %>%
  right_join(norm_days, by=c("NAME" = "State"))
```

```
tm_shape(norm_days_states, projection = 2163) +
  tm_polygons("normDays" , title = "Market Days per 1000") +
  tm_layout(frame = FALSE) +
  tm_layout(main.title = "Number of Market days in the US", title.size = 1.5,main.title.position="cente
```

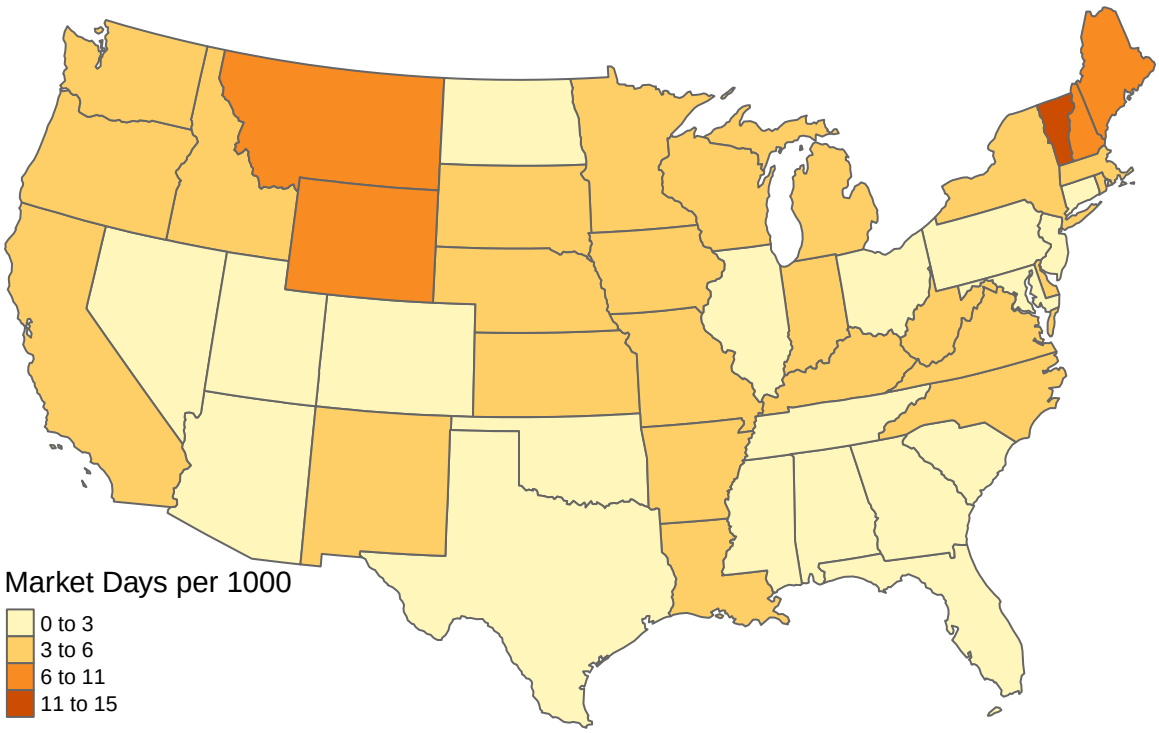
```
## Warning: The shape norm_days_states contains empty units.
```



With categories breaks of 0, 5, 10, and 15 market days open per 1,000 population, the state of Farmers Markets across the country looks rather grim.

```
tm_shape(norm_days_states, projection = 2163) +
  tm_polygons("normDays" , title = "Market Days per 1000", breaks=c(0,3,6,11,15)) +
  tm_layout(frame = FALSE) +
  tm_layout(main.title = "Number of Market days in the US", title.size = 1.5,main.title.position="cente
```

```
## Warning: The shape norm_days_states contains empty units.
```



Looking at the number of days that Farmers Markets are open shows how often throughout the year consumers are able to shop at markets. Ignoring population means that markets may be busy or perhaps in a neighboring county. I was surprised to find an almost inverse effect of latitude.

Number of Markets by State

If we simply look at the number of markets in each state we can avoid all of the date cleaning that was required in the previous section.

```
data_full %>%
  group_by(State) %>%
  summarise(N = n()) %>%
  right_join(state_population) %>%
  mutate(NperCapita = N/pop*100000) -> markets_by_states
```

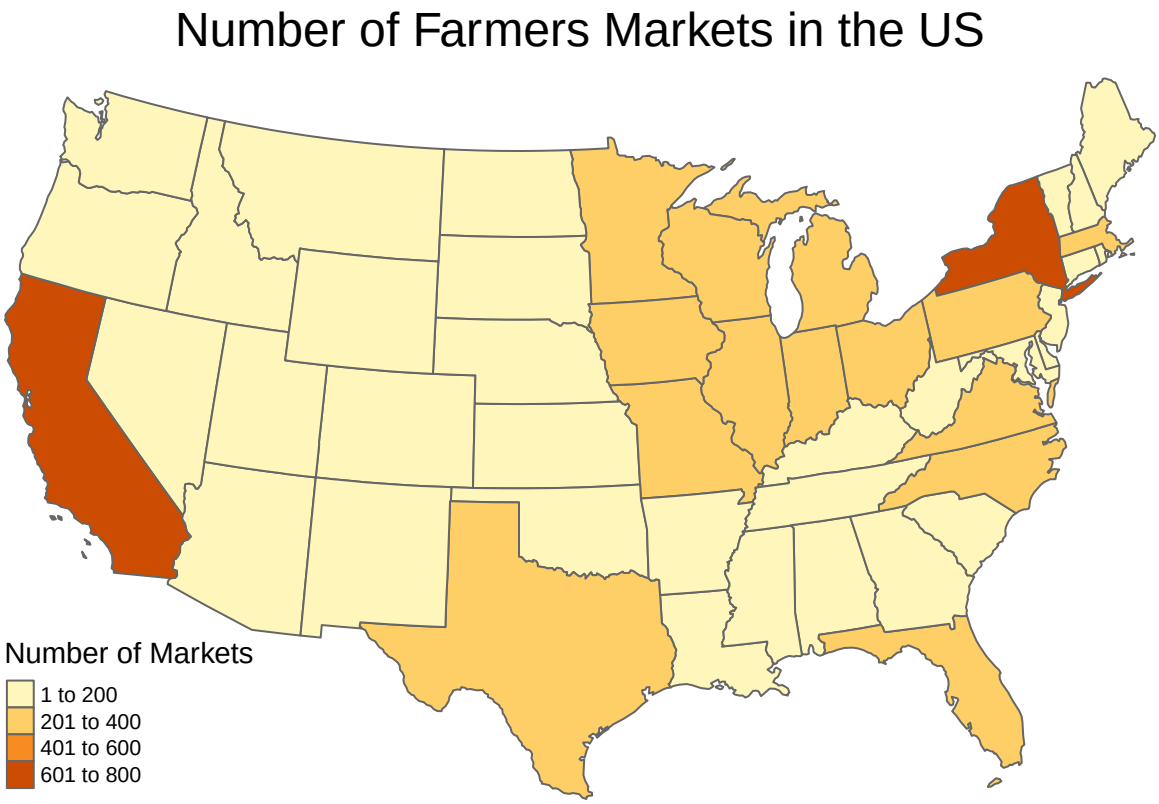
```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## Joining, by = "State"
```

```
state_market_counts =
  us_states %>%
  right_join(markets_by_states, by=c("NAME" = "State"))

tm_shape(state_market_counts, projection = 2163) +
```

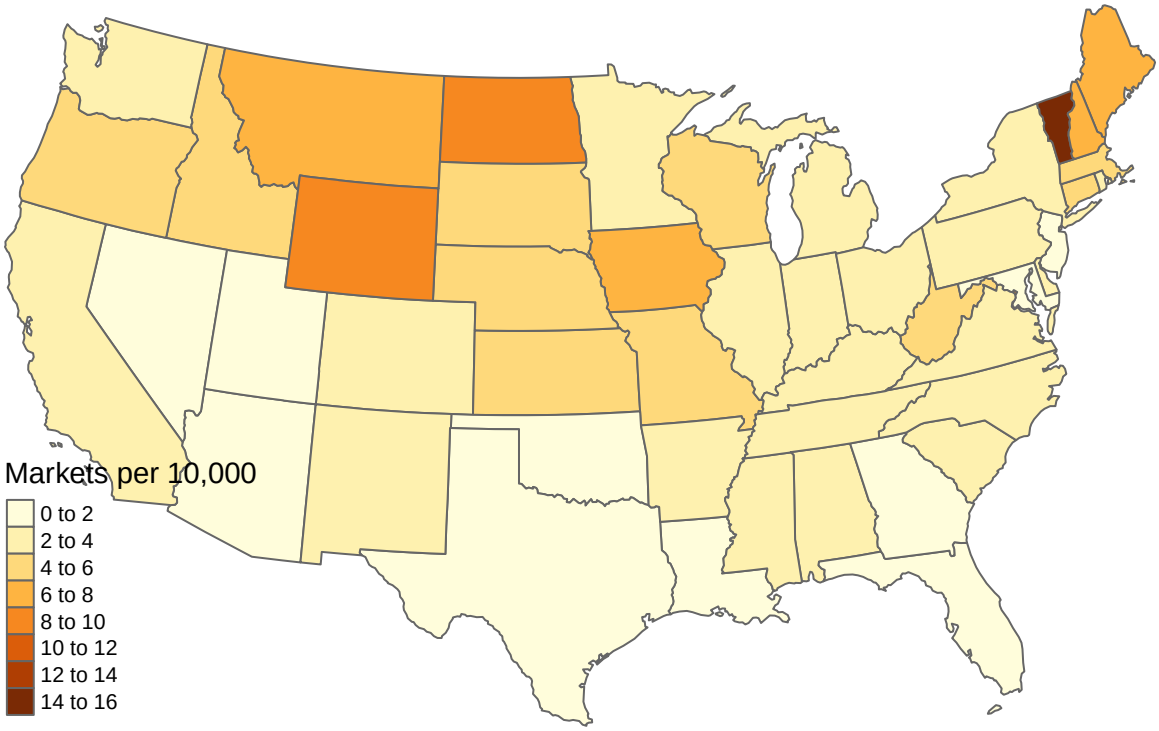
```
  tm_polygons("N" , title="Number of Markets") +
  tm_layout(frame = FALSE) +
  tm_layout(main.title = "Number of Farmers Markets in the US", title.size = 1.5,main.title.position="c
```

```
## Warning: The shape state_market_counts contains empty units.
```




```
tm_shape(state_market_counts, projection = 2163) +
  tm_polygons("NperCapita", title="Markets per 10,000") +
  tm_layout(frame = FALSE) +
  tm_layout(main.title = "Number of Farmers Markets in the US per Capita", title.size = 1.5,main.title.
```

```
## Warning: The shape state_market_counts contains empty units.
```



Looking at the total number of markets in each state show that New York and California clearly have the most, however, per capita they are well below average. Looking at these states at the county level could be useful. Vermont stands out in the leader of markets per capita and in number of open market days. Oregon, Montana, Wyoming, New Hampshire, and Maine are leaders in open market days and also stand out in number of markets per capita.