graphs

Graham Dynis

2024-09-18

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.3

## Warning: package 'ggplot2' was built under R version 4.2.3

## Warning: package 'tibble' was built under R version 4.2.3

## Warning: package 'tidyr' was built under R version 4.2.3

## Warning: package 'readr' was built under R version 4.2.3

## Warning: package 'purrr' was built under R version 4.2.3

## Warning: package 'dplyr' was built under R version 4.2.3

## Warning: package 'stringr' was built under R version 4.2.3

## Warning: package 'forcats' was built under R version 4.2.3

## Warning: package 'lubridate' was built under R version 4.2.3

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(stats)
library(ggplot2)
library(readr)
library(dplyr)
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.2.3

##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```r
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.2.3

##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

```r
df_train = read.csv('df_train.csv')
cat("Training Data Types:\n")
```

```
## Training Data Types:
```

```r
str(df_train)
```

```
## 'data.frame':    58327370 obs. of  21 variables:
##  $ id          : chr  "FOODS_1_001_CA_1" "FOODS_1_001_CA_1" "FOODS_1_001_CA_1" "FOODS_1_001_CA_1" ..
##  $ wm_yr_wk    : int  11101 11101 11101 11101 11101 11101 11101 11102 11102 11102 ...
##  $ date        : chr  "2011-02-03" "2011-02-04" "2011-01-31" "2011-01-29" ...
##  $ item_id     : chr  "FOODS_1_001" "FOODS_1_001" "FOODS_1_001" "FOODS_1_001" ...
##  $ dept_id     : chr  "FOODS_1" "FOODS_1" "FOODS_1" "FOODS_1" ...
##  $ cat_id      : chr  "FOODS" "FOODS" "FOODS" "FOODS" ...
##  $ store_id    : chr  "CA_1" "CA_1" "CA_1" "CA_1" ...
##  $ state_id    : chr  "CA" "CA" "CA" "CA" ...
##  $ sales       : int  2 0 0 3 0 4 1 0 0 2 ...
##  $ weekday     : chr  "Thursday" "Friday" "Monday" "Saturday" ...
##  $ wday        : int  6 7 3 1 2 5 4 4 2 1 ...
##  $ month       : int  2 2 1 1 1 2 2 2 2 2 ...
##  $ year        : int  2011 2011 2011 2011 2011 2011 2011 2011 2011 2011 ...
##  $ event_name_1: chr  "None" "None" "None" "None" ...
##  $ event_type_1: chr  "None" "None" "None" "None" ...
##  $ event_name_2: chr  "None" "None" "None" "None" ...
##  $ event_type_2: chr  "None" "None" "None" "None" ...
##  $ snap_CA     : int  1 1 0 0 0 1 1 1 1 1 ...
##  $ snap_TX     : int  1 0 0 0 0 0 1 0 1 1 ...
##  $ snap_WI     : int  1 0 0 0 0 1 0 1 1 1 ...
##  $ sell_price  : num  2 2 2 2 2 2 2 2 2 2 ...
```

```r
df_train <- df_train %>%
  mutate(month_year_index = (year - 2011) * 12 + month)
df_train <- df_train %>%
  mutate(month_year_index = month_year_index - min(month_year_index) + 1)
head(df_train[, c("year", "month", "month_year_index")])
```

```
##   year month month_year_index
## 1 2011     2                2
## 2 2011     2                2
## 3 2011     1                1
## 4 2011     1                1
## 5 2011     1                1
## 6 2011     2                2
```

```r
df_summary <- df_train %>%
  group_by(month_year_index, state_id) %>%
  summarise(total_sales = sum(sales, na.rm = TRUE))
```

```
## ‘summarise()‘ has grouped output by ’month_year_index’. You can override using
## the ‘.groups‘ argument.
```

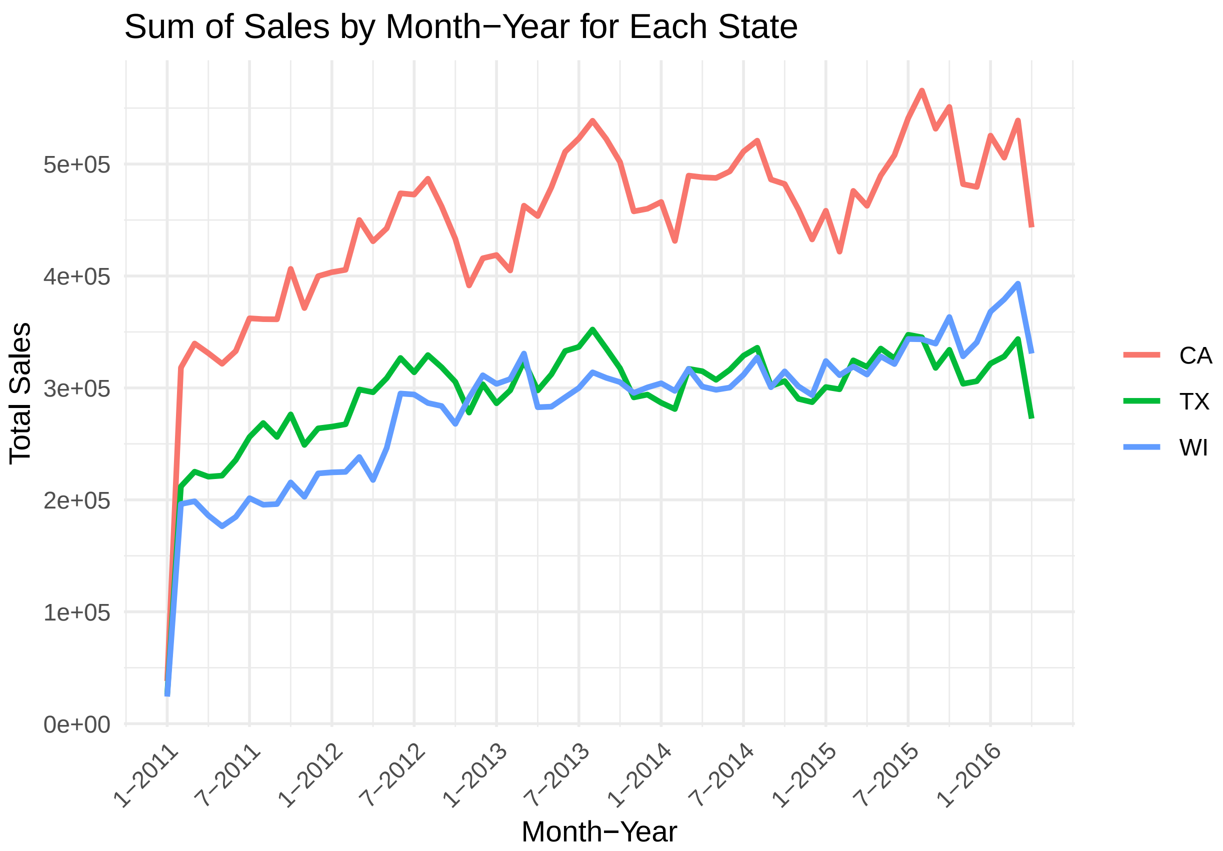
```r
df_train <- df_train %>%
  mutate(month_year_label = paste(month, year, sep = "-"))
```

```r
df_labels <- df_train %>%
  select(month_year_index, month_year_label) %>%
  distinct()
```

```r
breaks_6_months <- seq(1, max(df_summary$month_year_index), by = 6)
break_labels <- df_labels %>%
  filter(month_year_index %in% breaks_6_months)
```

```r
ggplot(df_summary, aes(x = month_year_index, y = total_sales, color = state_id, group = state_id)) +
  geom_line(size = 1) +
  labs(title = "Sum of Sales by Month-Year for Each State",
       x = "Month-Year",
       y = "Total Sales") +
  theme_minimal() +
  theme(legend.title = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_continuous(
    breaks = breaks_6_months,
    labels = break_labels$month_year_label
  )
```

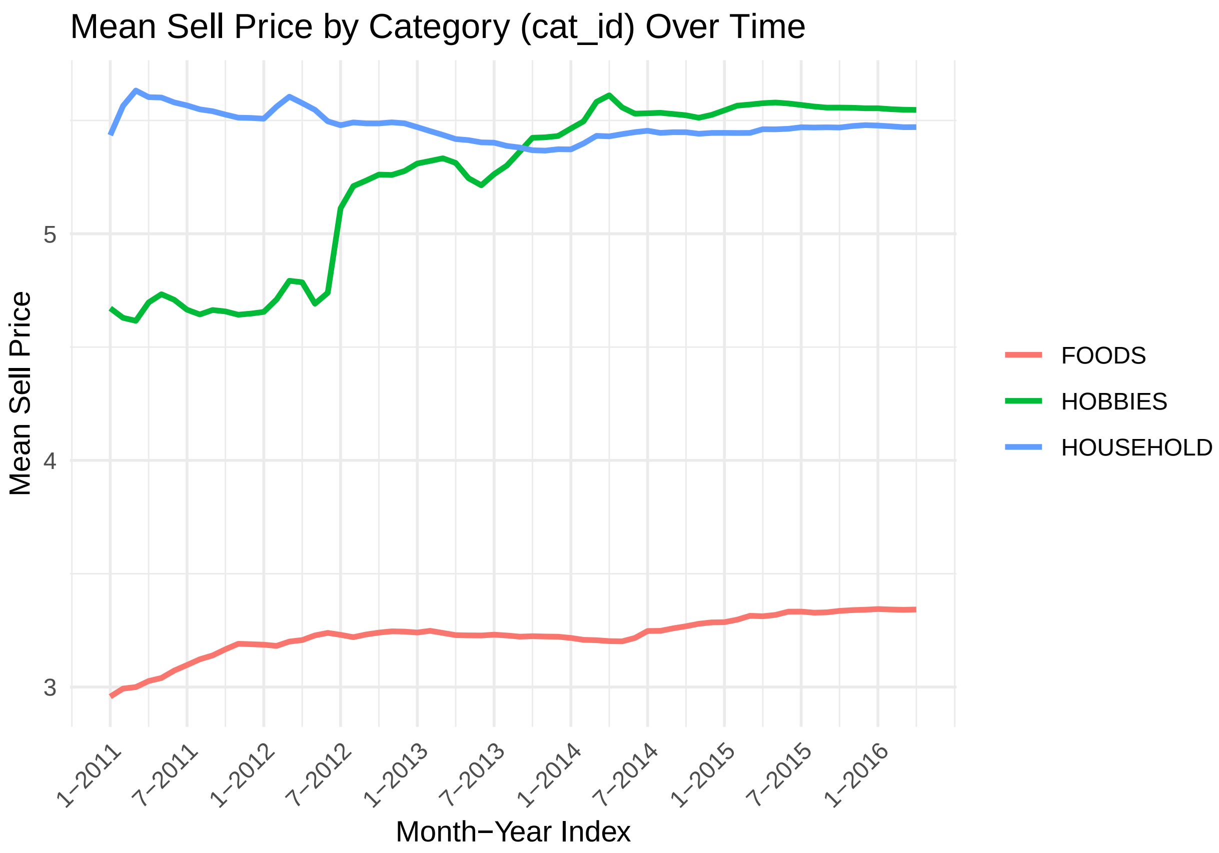
```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```



```
df_summary2 <- df_train %>%
  group_by(month_year_index, cat_id) %>%
  summarise(mean_sell_price = mean(sell_price, na.rm = TRUE))
```

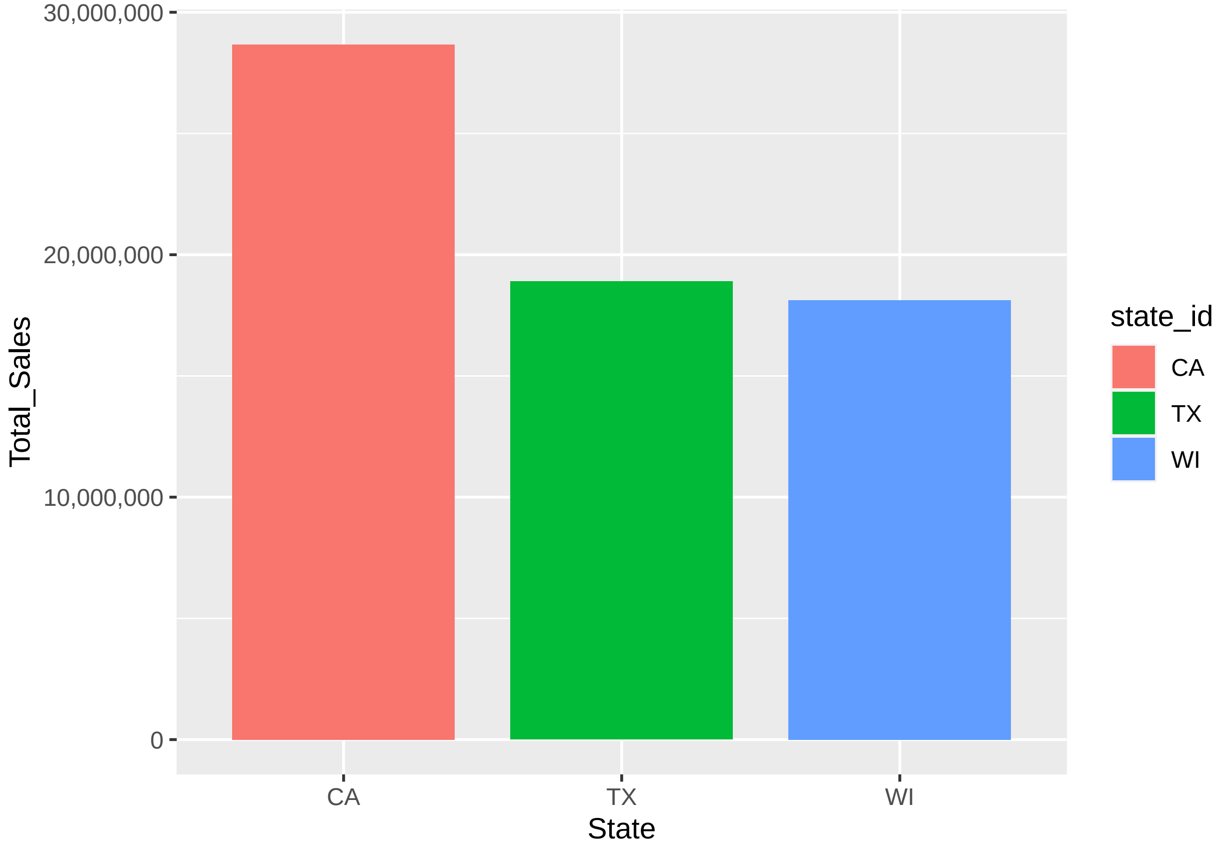
```
## `summarise()` has grouped output by 'month_year_index'. You can override using
## the `.groups` argument.
```

```
ggplot(df_summary2, aes(x = month_year_index, y = mean_sell_price, color = cat_id, group = cat_id)) +
  geom_line(size = 1) +
  labs(title = "Mean Sell Price by Category (cat_id) Over Time",
       x = "Month-Year Index",
       y = "Mean Sell Price") +
  theme_minimal() +
  theme(legend.title = element_blank(),
        axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_continuous(breaks = breaks_6_months,
                     labels = break_labels$month_year_label)
```

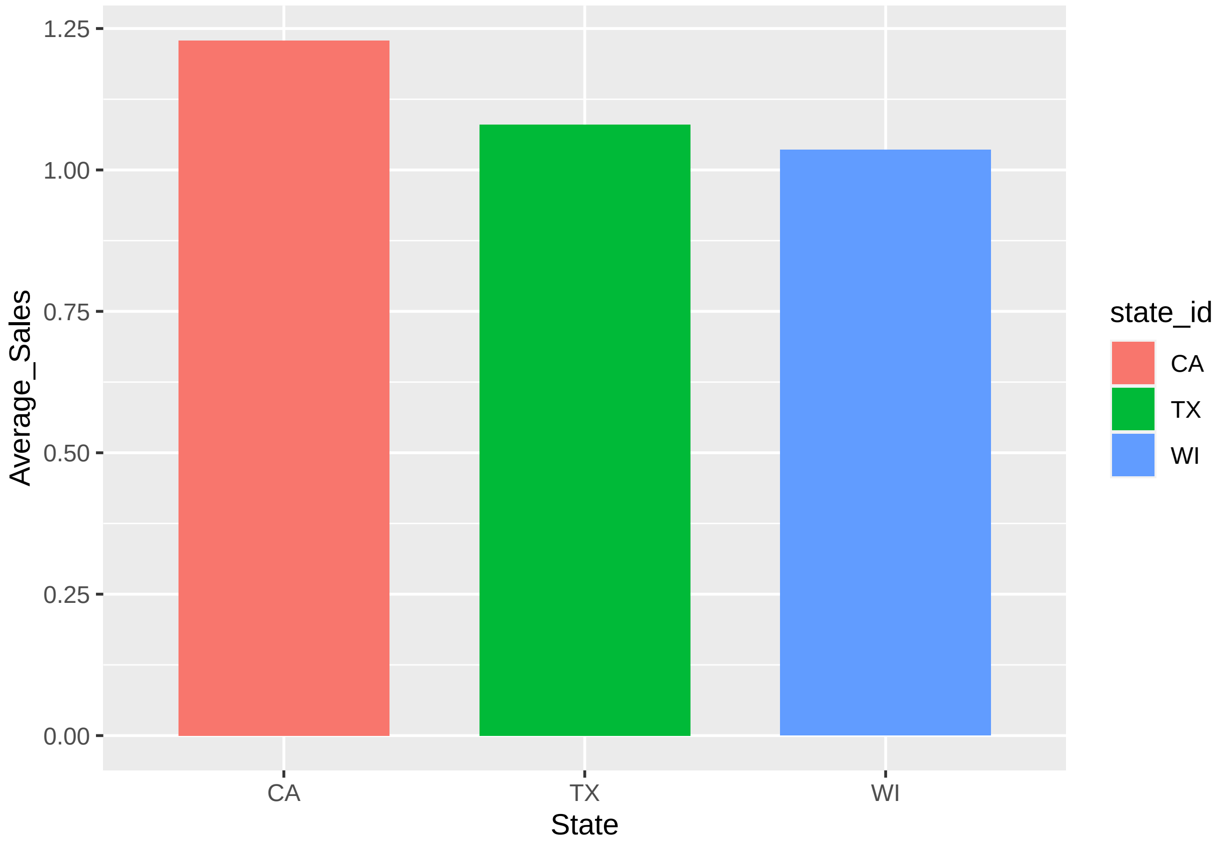


```r
df_state = df_train %>% group_by(state_id) %>% summarise(total_sales = sum(sales))
df_state_ave = df_train %>% group_by(state_id) %>% summarise(avg_sales = mean(sales))
```

```r
require(scales)
#Graph
ggplot(data = df_state, aes(x=state_id, y=total_sales, fill=state_id)) +
  geom_col(width = 0.8) + scale_y_continuous(labels = label_comma()) +
  labs(
         x = "State",
         y = "Total_Sales"
        )
```



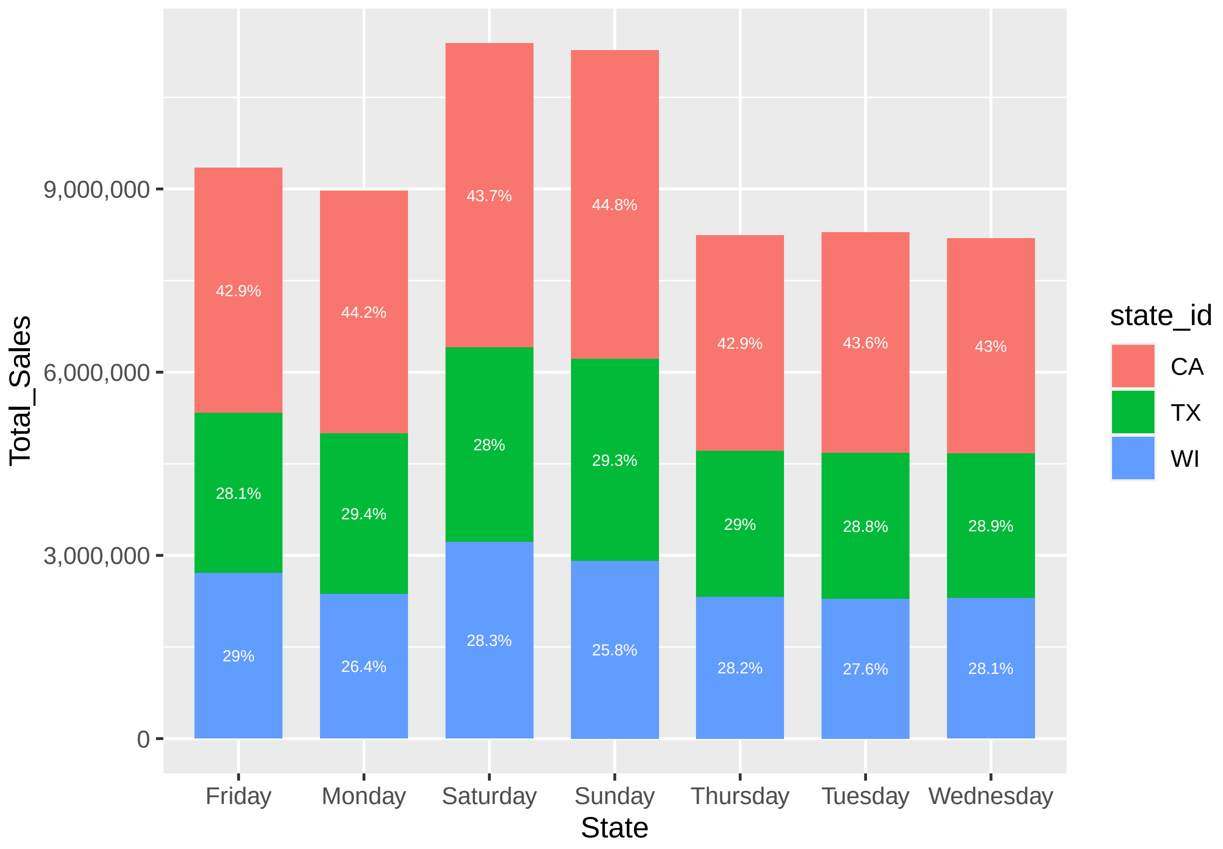
```
ggplot(df_state_ave, aes(x=state_id, y=avg_sales, fill=state_id)) +
  geom_col(width = 0.7)+ labs(
         x = "State",
         y = "Average_Sales"
        )
```



```
df_weekday <- df_train %>% group_by(state_id, weekday) %>% summarise(total_sales = sum(sales))
```

```
## ‘summarise()‘ has grouped output by ’state_id’. You can override using the
## ‘.groups‘ argument.
```

```
df_weekday <- df_weekday %>%
  group_by(weekday) %>%
  mutate(pct_sales = total_sales / sum(total_sales) * 100)
ggplot(df_weekday, aes(x=weekday, y=total_sales, fill=state_id, label = paste0(round(pct_sales, 1), "%"
  geom_col(width = 0.7)+ labs(
         x = "State",
         y = "Total_Sales"
        ) + scale_y_continuous(labels = label_comma()) +
  geom_text(
  size = 2, position = position_stack(vjust = 0.5),colour = "white")
```



Total_Sales
9,000,000
6,000,000
3,000,000
0
42.9%
28.1%
29%
44.2%
29.4%
26.4%
43.7%
28%
28.3%
44.8%
29.3%
25.8%
42.9%
29%
28.2%
43.6%
28.8%
27.6%
43%
28.9%
28.1%
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
State
state_id
CA
TX
WI