graphs

Graham Dynis

2024-09-18

```
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
```

```
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
```

```
library(data.table)
```

```
## Warning: package 'data.table' was built under R version 4.2.3
```

```
##
## Attaching package: 'data.table'
##
## The following objects are masked from 'package:reshape2':
##
##     dcast, melt
##
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
##
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
##
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
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
```

```
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
```

```
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.
```

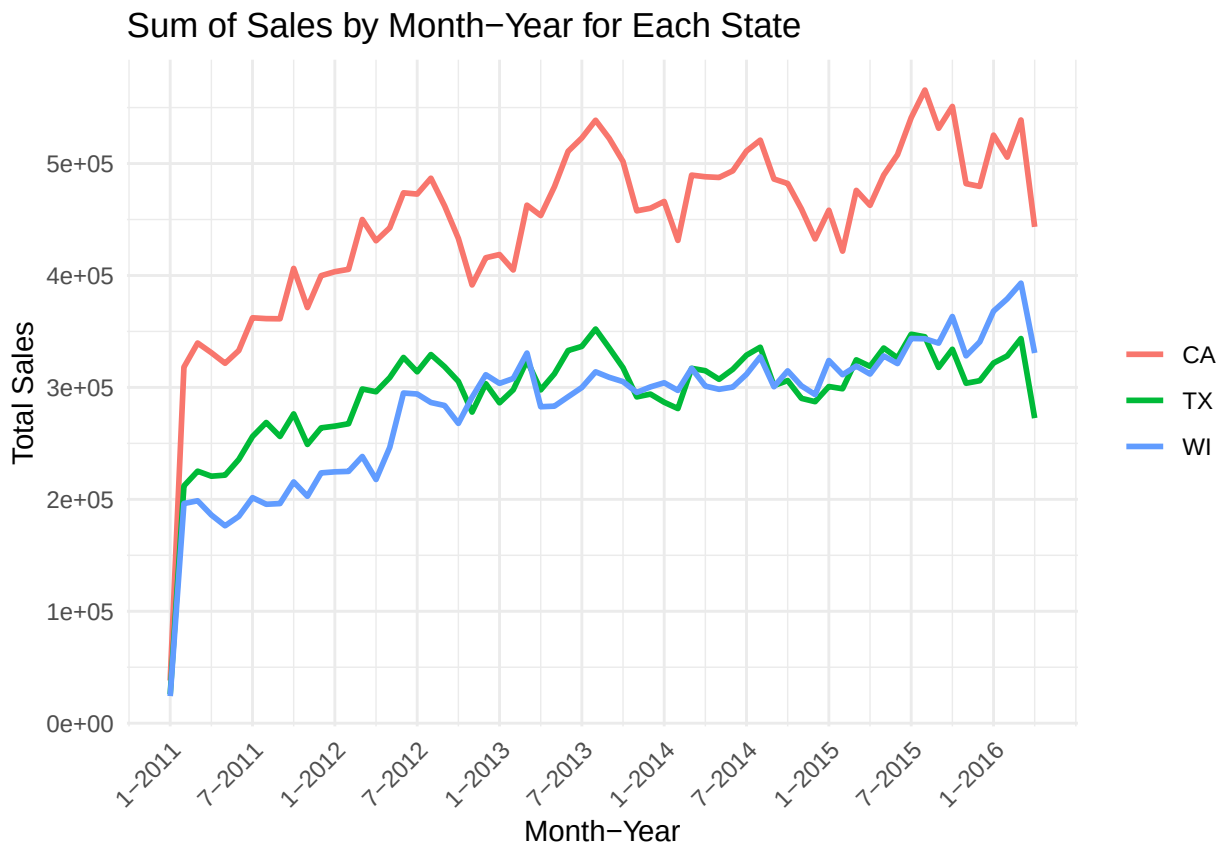
```
df_train <- df_train %>%
  mutate(month_year_label = paste(month, year, sep = "-"))
```

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

```
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)
```

```
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.
```

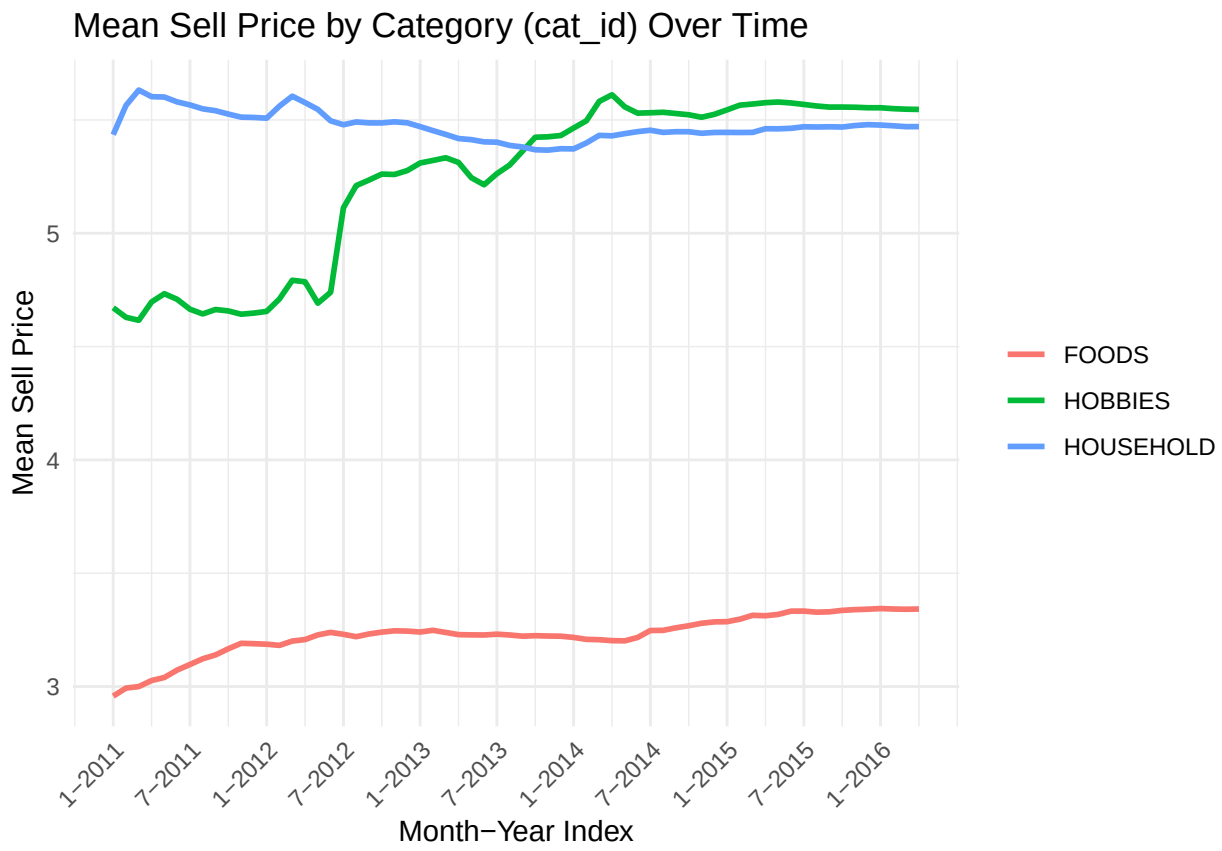


Here we have a graph that shows the total number of sales made per month by state. California's numbers clear those of Texas and Wisconsin mostly due to the fact that our dataset pulls data from four stores in California and just 3 each in Texas and Wisconsin. If you normalize California's numbers to match the number of stores in TX and WI, then the graphs match quite well. There is obvious seasonality as each line peaks in August every year, which we have correlated with back to school shopping being the primary driver. There are also smaller peaks that occur yearly in December for holiday shopping.

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

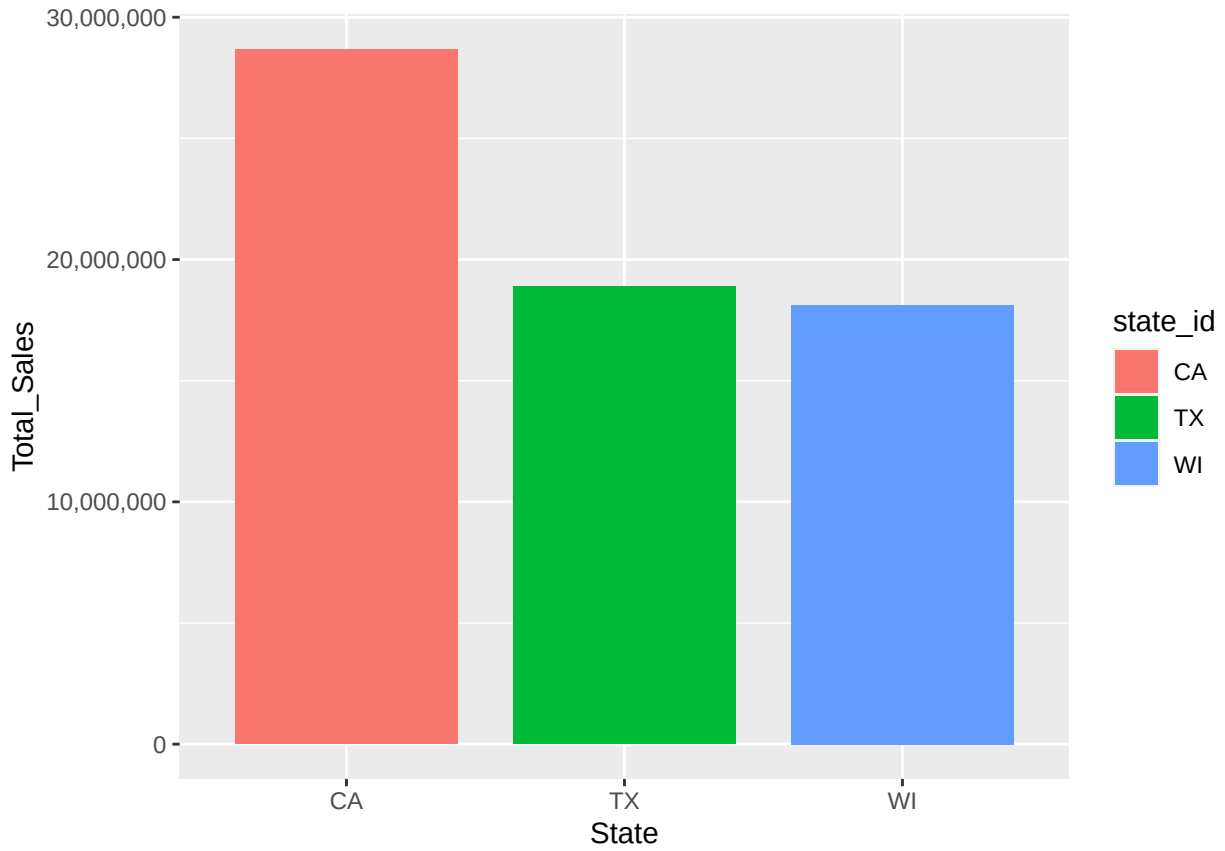
```
## `summarise()` has grouped output by 'month_year_index'. You can override using
## the `.groups` argument.
```

```
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)
```



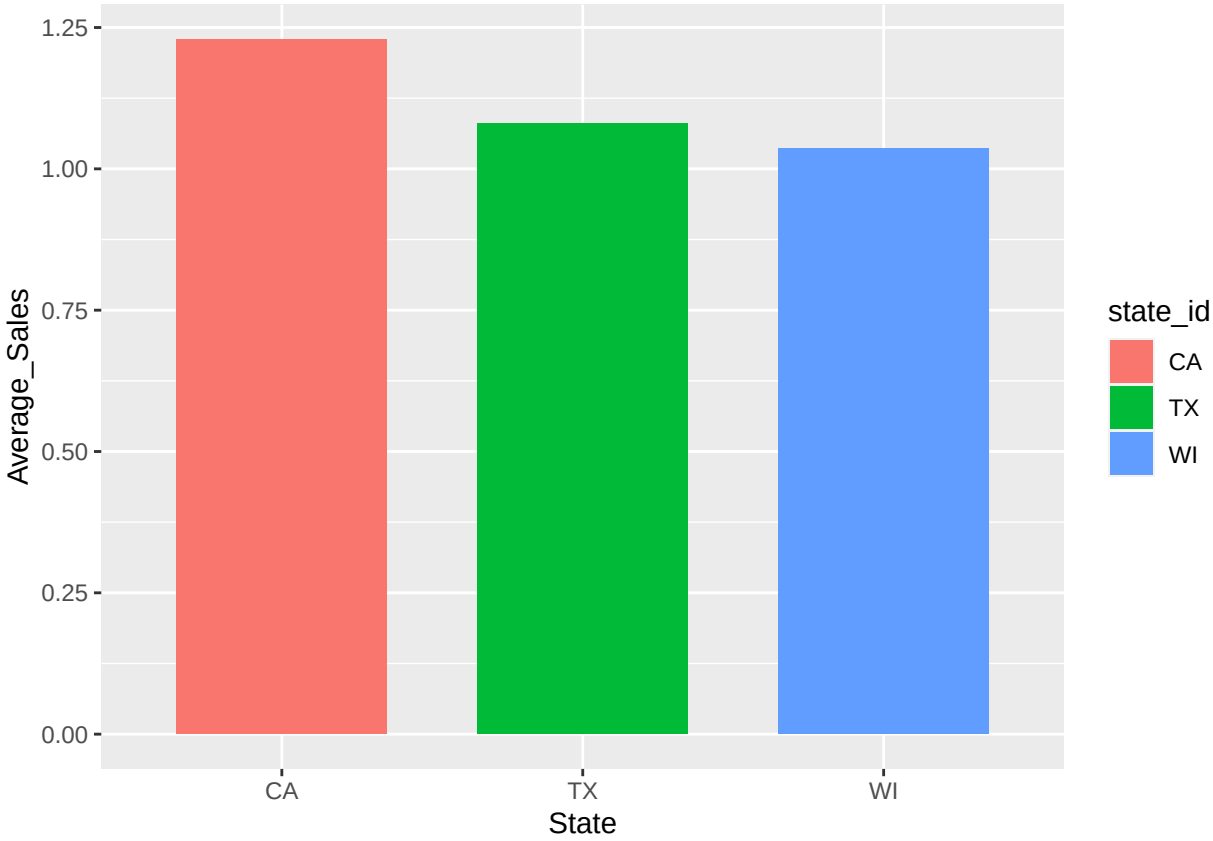
This graph demonstrates the average sell price of each category sold at Walmart over time. We can see that during this time frame, the average sell price of food products stays relatively constant, with minimal rise over time. Hobbies and household products consistently have a higher mean sell price than food products, with the mean sell price of hobbies overtaking household around the fourth quarter of 2013. Hobbies continues to have a higher mean sell price going into 2016.

```
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"
        )
```



We made a graph of total sales by state to identify which state is selling the most. As seen above, we find that California is selling the most. One reason could be that Californians have a higher income and are able to purchase more product. But a closer look at the dataset reveals that the dataset has 4 stores for California and 3 for Texas and Wisconsin. This is also verified in the next graph.

```
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"
        )
```

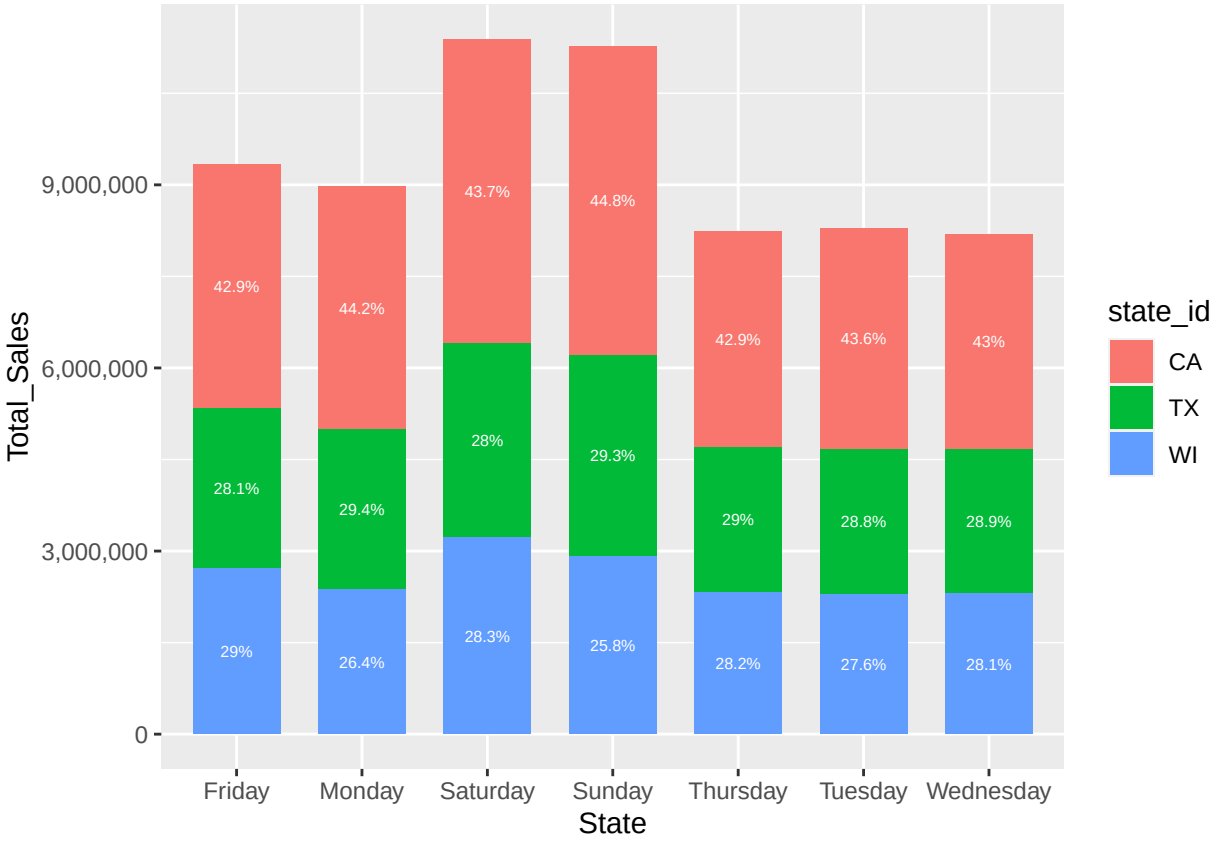


Here, we see that the average sale per product per day is very similar for all states. This shows that on average, the Walmart stores sell the same. And the discrepancy in the total_sales graph is due to the extra store that the dataset records for California.

```
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")
```



We decided to visualize the total sales on each day. We find that the most sales happen on Saturday and Sunday which is expected as they are the weekend. And we also find the percentage of sales for each state is constant and California contributes more to the total sales / day due to the extra store for California.