Data Science Internship Assessment 2021/22

Kabilan Kathiravel

20/08/2021

Task One: Load the data into an environment of your choice. Have a look at the structure of the data. What information are you provided?

From the instructions provided in the assessment, the sales data is in "train.csv". Hence it would be worthwhile to look through (and understand) the data here first, before considering the supplementary data provided in "stores.csv".

```
library(tidyverse)
library(ggplot2)
sales_data = read_csv("ds_test_data/train.csv")
head(sales_data)
```

```
## # A tibble: 6 x 5
##   Store  Dept Date       Weekly_Sales IsHoliday
##   <dbl> <dbl> <date>            <dbl> <lgl>
## 1     1     1 2010-02-05       24924. FALSE
## 2     1     1 2010-02-12       46039. TRUE
## 3     1     1 2010-02-19       41596. FALSE
## 4     1     1 2010-02-26       19404. FALSE
## 5     1     1 2010-03-05       21828. FALSE
## 6     1     1 2010-03-12       21043. FALSE
```

Our sales data here is rectangular. Our response variable in terms of this tibble is Weekly_Sales (we want to predict the number of sales), which is a double. From looking at the tibble in RStudio's environment we can see that this tibble has 421570 rows (each row corresponding to a distinct observation). From the instructions provided for the assessment, we are told that this data corresponds to weekly sales data for a major retailer by store and department, so what this means is our rows (if arranged correctly) should be grouped by week, store and department. We have four possible explanatory variables:

- Store - A double that pertains to a specific store from which sales (and other data) has been collected for a specific week
- Dept - A double that pertains to a specific department from a store from which sales (and other data) has been collected for a specific week
- Date - A date object that pertains to the day when the weekly data represented in that row was collected for that specific combination of store and department (so essentially marking a specific week for that store and department combination). Hence each date for a store and department should be separated by any adjacent dates by seven days
- IsHoliday - A logical vector which pertains to whether a particular date is a holiday or not

Our response, Weekly_Sales is a double.

Each of these explanatory variables seem to be potentially useful in making predictions for our response (and will be needed for Task Two anyway), so I don't think we should remove any of them at this stage.

There doesn't seem to be much information about where this data specifically came from. I can't determine what type of retailer this is, what any of the products they sell are, or even the location of the retailer. With this in mind, I will go through this whole exercise without making hasty assumptions that rely on knowledge of these factors, as they could potentially end up leading to incorrect conclusions.

However, it's worth checking if our data is clean.

```
sales_data %>% str()
```

```
## spec_tbl_df [421,570 x 5] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Store       : num [1:421570] 1 1 1 1 1 1 1 1 1 1 ...
##  $ Dept        : num [1:421570] 1 1 1 1 1 1 1 1 1 1 ...
##  $ Date        : Date[1:421570], format: "2010-02-05" "2010-02-12" ...
##  $ Weekly_Sales: num [1:421570] 24925 46039 41596 19404 21828 ...
##  $ IsHoliday   : logi [1:421570] FALSE TRUE FALSE FALSE FALSE FALSE ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Store = col_double(),
##   ..   Dept = col_double(),
##   ..   Date = col_date(format = ""),
##   ..   Weekly_Sales = col_double(),
##   ..   IsHoliday = col_logical()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
sales_data %>% summary()
```

```
##      Store           Dept            Date              Weekly_Sales
##  Min.   : 1.0   Min.   : 1.00   Min.   :2010-02-05   Min.   : -4989
##  1st Qu.:11.0   1st Qu.:18.00   1st Qu.:2010-10-08   1st Qu.:  2080
##  Median :22.0   Median :37.00   Median :2011-06-17   Median :  7612
##  Mean   :22.2   Mean   :44.26   Mean   :2011-06-18   Mean   : 15981
##  3rd Qu.:33.0   3rd Qu.:74.00   3rd Qu.:2012-02-24   3rd Qu.: 20206
##  Max.   :45.0   Max.   :99.00   Max.   :2012-10-26   Max.   :693099
##  IsHoliday
##  Mode :logical
##  FALSE:391909
##  TRUE :29661
##
##
##
```

From this summary, Store, Dept and Date look fine (nothing that clearly seems out of place given what we were told in the Assessment Brief). However, it appears Weekly_sales is unclean. If we had a lot of data (say maybe ten million rows), and it wasn't grouped, then removing all the rows corresponding to this would be a simple way to sort it out. However since that is not the case (and this is time series data), it's worth investigating these then figuring out what the most sensible thing to do might be (as negative values for Weekly_Sales are invalid, some kind of replacement with a non-negative value at worst can't be any worse than keeping them the same!).

With this in mind, we should first look at all the affected rows (my assumption is that only Weekly_sales of less than 0 are a definite concern).

```
sales_data %>% filter(Weekly_Sales < 0)
```

```
## # A tibble: 1,285 x 5
##    Store  Dept Date       Weekly_Sales IsHoliday
##    <dbl> <dbl> <date>            <dbl> <lgl>
##  1     1     6 2012-08-10      -140.   FALSE
##  2     1    18 2012-05-04        -1.27 FALSE
##  3     1    47 2010-02-19      -863    FALSE
##  4     1    47 2010-03-12      -698    FALSE
##  5     1    47 2010-10-08       -58    FALSE
##  6     1    47 2011-04-08      -298    FALSE
##  7     1    47 2011-07-08      -198    FALSE
##  8     1    47 2011-10-14      -498    FALSE
##  9     1    47 2011-12-23      -498    FALSE
## 10     1    47 2012-02-17      -198    FALSE
## # ... with 1,275 more rows
```

Unfortunately, we have 1285 rows that are affected in this regard. While this is only about 0.3% of the data, I think it is a reasonably significant number considering the data we are working with (at least compared to say just 10 rows), so unless we do some intense analysis, getting an 'ideal' change could be tricky.

We also should check for any missing values.

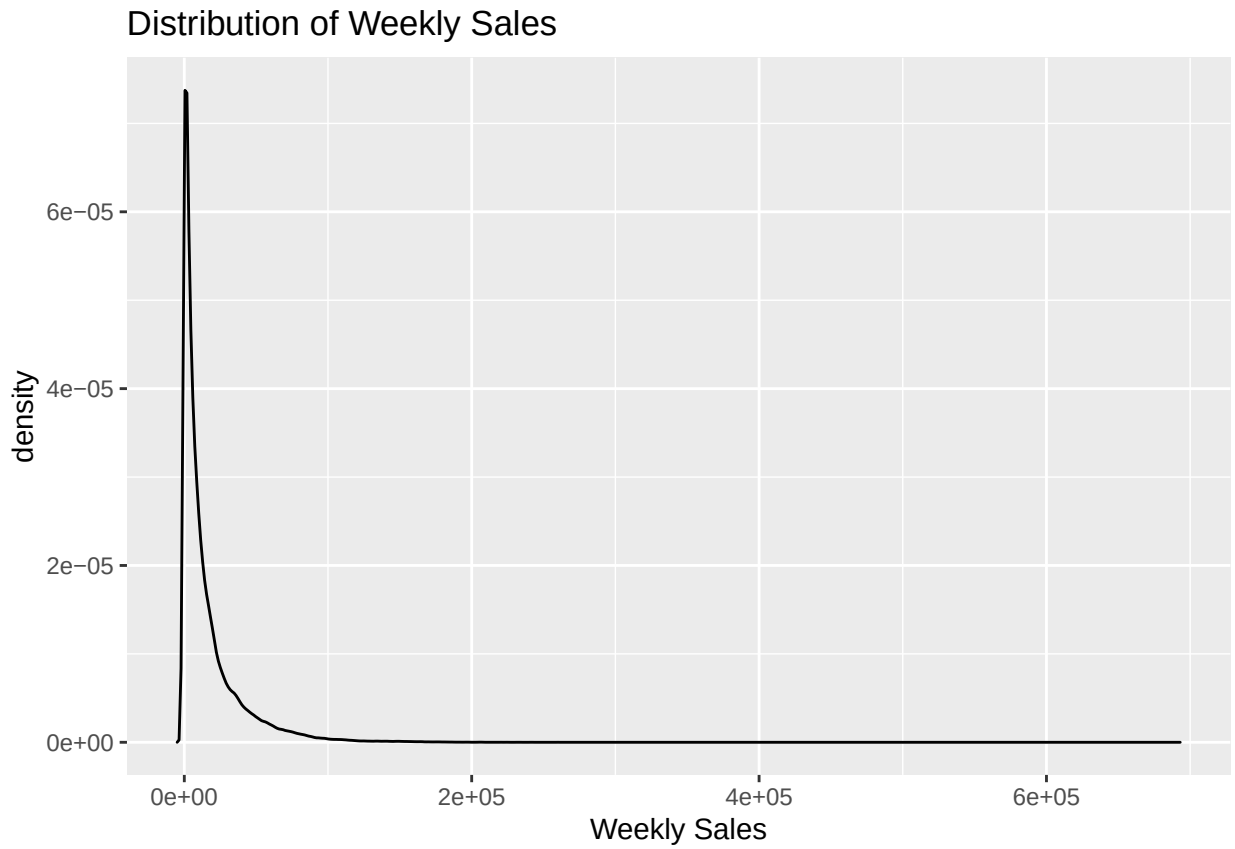
```
sales_data %>% summarise(na = sum(is.na(Weekly_Sales), is.na(Date), is.na(IsHoliday), is.na(Dept), is.na
```

```
## # A tibble: 1 x 1
##      na
##   <int>
## 1     0
```

There are no concerns with missing variables, so we just need to consider dealing with our negative values of sales before doing anything further.

To get an indication of the sales we are getting on different dates, we will graph this, so we can understand how 'unusual' our negative values are in relation to the other values. It seems like a density plot would be the best way to achieve this.

```
ggplot(sales_data, aes(x=Weekly_Sales)) +
  geom_density() +
  ggtitle("Distribution of Weekly Sales") +
  labs(x = "Weekly Sales")
```



It is hard to see this clearly (as the scale of 1000 or e+04 for -4989 is small compared to 100000 or e+05), but what is evident is that negative values of weekly sales do exist. They are (fortunately) not particularly common in comparison to positive values, but some do exist, as can be seen with some x values below 0 being on this plot. Also that the max of 693099 is extraordinary and may be anomalous. However for now, we will avoid making any changes to higher extreme values as the plots we make later on should be more revealing. However, the negative values are a problem that should be dealt with before making these plots.

If we had a lot of time, I would consider doing some serious inspection of where the negative values occur in particular (in terms of Date, Store and Dept and trends), which might give some solid indication as to how we could precisely deal with this data. However, due to the complex grouping of Store, Dept and Date, this may take too long given the limited time nature of this assessment.

With this in mind, we should use some method to replace the negative values of Weekly_sales with non-negative values that isn't too complicated. At worst, it can't make our predictions any worse, and while about 1300 rows are affected, this only is 0.3% of the data (assuming of course that the other 99.7% has been entered correctly!). Replacing them with the mean, or median value of weekly_sales would be simple, but probably not the best approach as this ignores seasonality concerns (which are likely to come up when we investigate the trends in this data later).

Something I think would be even better than replacing the negative values with the mean or median would be to replace each relevant value with 'N/A', then take advantage of Tidyverse's 'fill' function to replace each affected value with the value above it (which will usually be the entry for the same store and department for the week before). I ensure that the data is ordered as I intended it to be (By Store and Department, then by Date) by using "arrange".

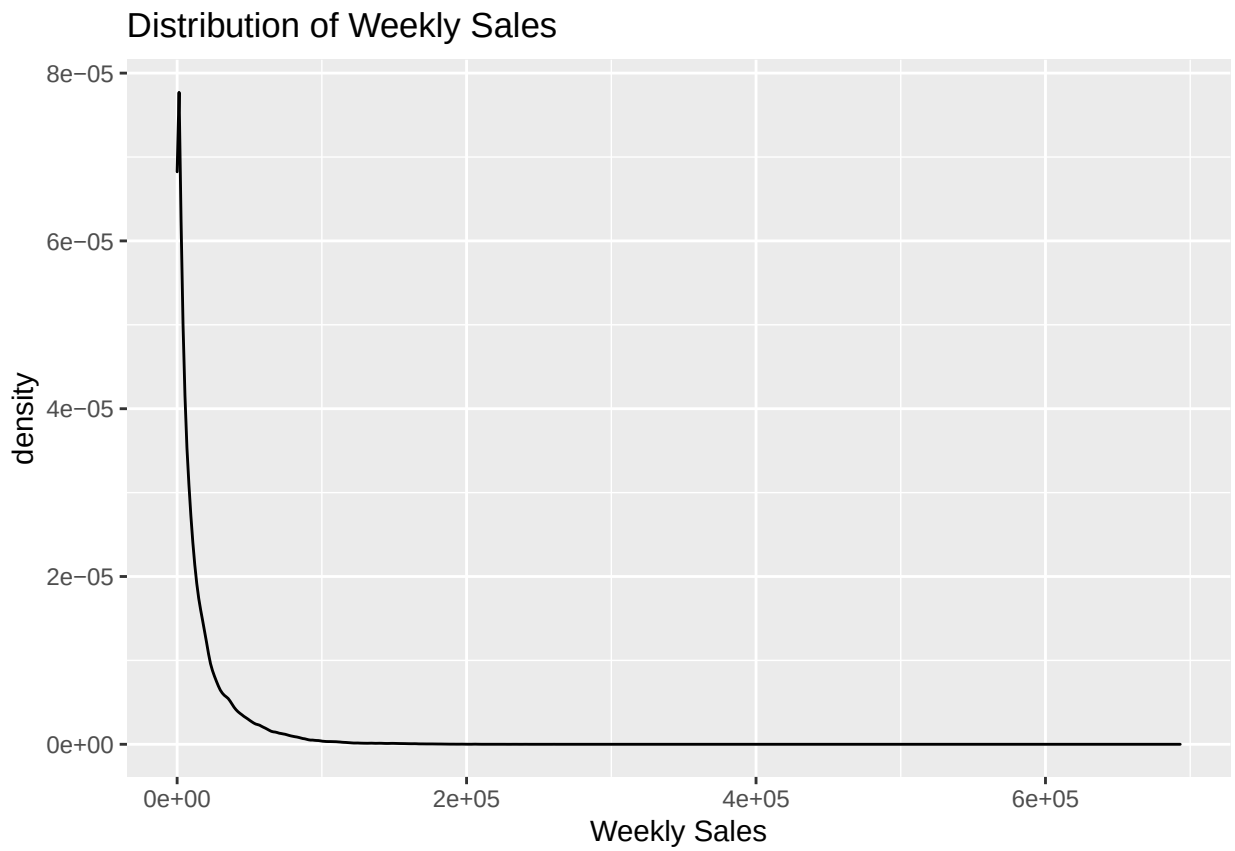
```
clean_sales_data <- sales_data  %>% mutate(Weekly_Sales = replace(Weekly_Sales, which(Weekly_Sales < 0)
 clean_sales_data <- clean_sales_data %>%
   arrange(Store, Dept, Date)
```

```
clean_sales_data <- clean_sales_data %>%
  fill(Weekly_Sales, .direction = "down")
clean_sales_data %>% summary()
```

```
##      Store           Dept            Date              Weekly_Sales
##  Min.   : 1.0   Min.   : 1.00   Min.   :2010-02-05   Min.   :     0
##  1st Qu.:11.0   1st Qu.:18.00   1st Qu.:2010-10-08   1st Qu.:  2082
##  Median :22.0   Median :37.00   Median :2011-06-17   Median :  7617
##  Mean   :22.2   Mean   :44.26   Mean   :2011-06-18   Mean   : 15985
##  3rd Qu.:33.0   3rd Qu.:74.00   3rd Qu.:2012-02-24   3rd Qu.: 20210
##  Max.   :45.0   Max.   :99.00   Max.   :2012-10-26   Max.   :693099
##  IsHoliday
##  Mode :logical
##  FALSE:391909
##  TRUE :29661
##
##
##
```

It looks like we have successfully cleaned all negative values from the data. Let's check our plot again.

```
ggplot(clean_sales_data, aes(x=Weekly_Sales)) +
  geom_density() +
  ggtitle("Distribution of Weekly Sales") +
  labs(x = "Weekly Sales")
```



Nothing is left of 0 in this plot. Consequently, we no longer need to worry about the negative values anymore, which is likely to lead to improvements to our analysis and predictions later on.

However, another thing that is immediately striking about this plot is that it shows that 0 seems to be close to the most common amount of sales in a week. Let's check the data where this is the case (as we should be asking if there was some data inputting error for these values too).

```
sales_data %>% filter(Weekly_Sales == 0)
```

```
## # A tibble: 73 x 5
##    Store  Dept Date       Weekly_Sales IsHoliday
##    <dbl> <dbl> <date>            <dbl> <lgl>
##  1     1    47 2011-03-11            0 FALSE
##  2     1    47 2011-08-12            0 FALSE
##  3     1    47 2011-08-19            0 FALSE
##  4     2    47 2012-04-13            0 FALSE
##  5     2    60 2010-03-19            0 FALSE
##  6     3    36 2012-08-17            0 FALSE
##  7     6    78 2010-02-26            0 FALSE
##  8     7    49 2011-06-17            0 FALSE
##  9     7    54 2011-01-21            0 FALSE
## 10     8    78 2011-08-26            0 FALSE
## # ... with 63 more rows
```

73 rows is not too significant in the grand scheme of our dataset having a few hundred thousand rows, and from quick inspection of the tibble, there doesn't appear to be any store or department which has 0 sales **every week** (this would at least be slightly alarming). Consequently, I don't think it is too big an assumption to assume these values are fine; the 0 values perhaps do represent weeks of no sales. This would be worth questioning more if we had some context as to what type of retailer the data or even department was from (for example a clothing store) as in some contexts, you would expect no weekly sales to be a less regular occurrence than others.

Otherwise, to make things easier later on, we can separate Date out into week, month and year.

```
library(lubridate)
clean_sales_data = clean_sales_data %>%
  mutate(week = week(Date),
         month = month(Date),
         year = year(Date))

head(clean_sales_data)
```

```
## # A tibble: 6 x 8
##   Store  Dept Date       Weekly_Sales IsHoliday  week month  year
##   <dbl> <dbl> <date>            <dbl> <lgl>     <dbl> <dbl> <dbl>
## 1     1     1 2010-02-05       24924. FALSE         6     2  2010
## 2     1     1 2010-02-12       46039. TRUE          7     2  2010
## 3     1     1 2010-02-19       41596. FALSE         8     2  2010
## 4     1     1 2010-02-26       19404. FALSE         9     2  2010
## 5     1     1 2010-03-05       21828. FALSE        10     3  2010
## 6     1     1 2010-03-12       21043. FALSE        11     3  2010
```

We need to read in our stores data, as we must join this to our sales data (for type and size comparisons later on).

```
stores_data <- read_csv("ds_test_data/stores.csv")
head(stores_data)
```

```
## # A tibble: 6 x 3
##   Store Type    Size
##   <dbl> <chr>  <dbl>
## 1     1 A     151315
## 2     2 A     202307
## 3     3 B      37392
## 4     4 A     205863
## 5     5 B      34875
## 6     6 A     202505
```

```
stores_data %>% summary()
```

```
##      Store        Type                Size
##  Min.   : 1   Length:45          Min.   : 34875
##  1st Qu.:12   Class :character   1st Qu.: 70713
##  Median :23   Mode  :character   Median :126512
##  Mean   :23                      Mean   :130288
##  3rd Qu.:34                      3rd Qu.:202307
##  Max.   :45                      Max.   :219622
```

There is not as much to comment on here. Each store has a type, however different stores can be of the same type. However, what is of note is that each store has a specific size. Perhaps an interesting thing to check would be if the number of distinct sizes is the same as the number of distinct stores, as if this is not the case, then this probably means that some stores have the same size. First though, we again will check for any missing values.

```
stores_data %>% summarise(na = sum(is.na(Store), is.na(Type), is.na(Size)))
```

```
## # A tibble: 1 x 1
##      na
##   <int>
## 1     0
```

Fortunately there are none. We will check for the distinct sizes.

```
size = stores_data %>%
  group_by(Size) %>%
  count()

head(size)
```

```
## # A tibble: 6 x 2
## # Groups:   Size [6]
##    Size     n
##   <dbl> <int>
## 1 34875     1
## 2 37392     1
## 3 39690     3
```

```
## 4 39910     3
## 5 41062     1
## 6 42988     1
```

If we inspect the 'size' tibble, we see some sizes (203819, 39690, 39910) have multiple stores associated with them. Let's see how many distinct sizes we have.

```
nrow(size)
```

```
## [1] 40
```

We have 40 distinct sizes for 45 stores. This will be interesting to consider later on. For now however, let's join the two tibbles together .

```
clean_sales_data <- clean_sales_data %>% left_join(stores_data)
```

```
## Joining, by = "Store"
```

```
head(clean_sales_data)
```

```
## # A tibble: 6 x 10
##   Store  Dept Date       Weekly_Sales IsHoliday  week month  year Type    Size
##   <dbl> <dbl> <date>            <dbl> <lgl>     <dbl> <dbl> <dbl> <chr>  <dbl>
## 1     1     1 2010-02-05       24924. FALSE         6     2  2010 A     151315
## 2     1     1 2010-02-12       46039. TRUE          7     2  2010 A     151315
## 3     1     1 2010-02-19       41596. FALSE         8     2  2010 A     151315
## 4     1     1 2010-02-26       19404. FALSE         9     2  2010 A     151315
## 5     1     1 2010-03-05       21828. FALSE        10     3  2010 A     151315
## 6     1     1 2010-03-12       21043. FALSE        11     3  2010 A     151315
```

Task Two: Prepare and comment on the following exploratory plots:

a) Total weekly and monthly sales volumes

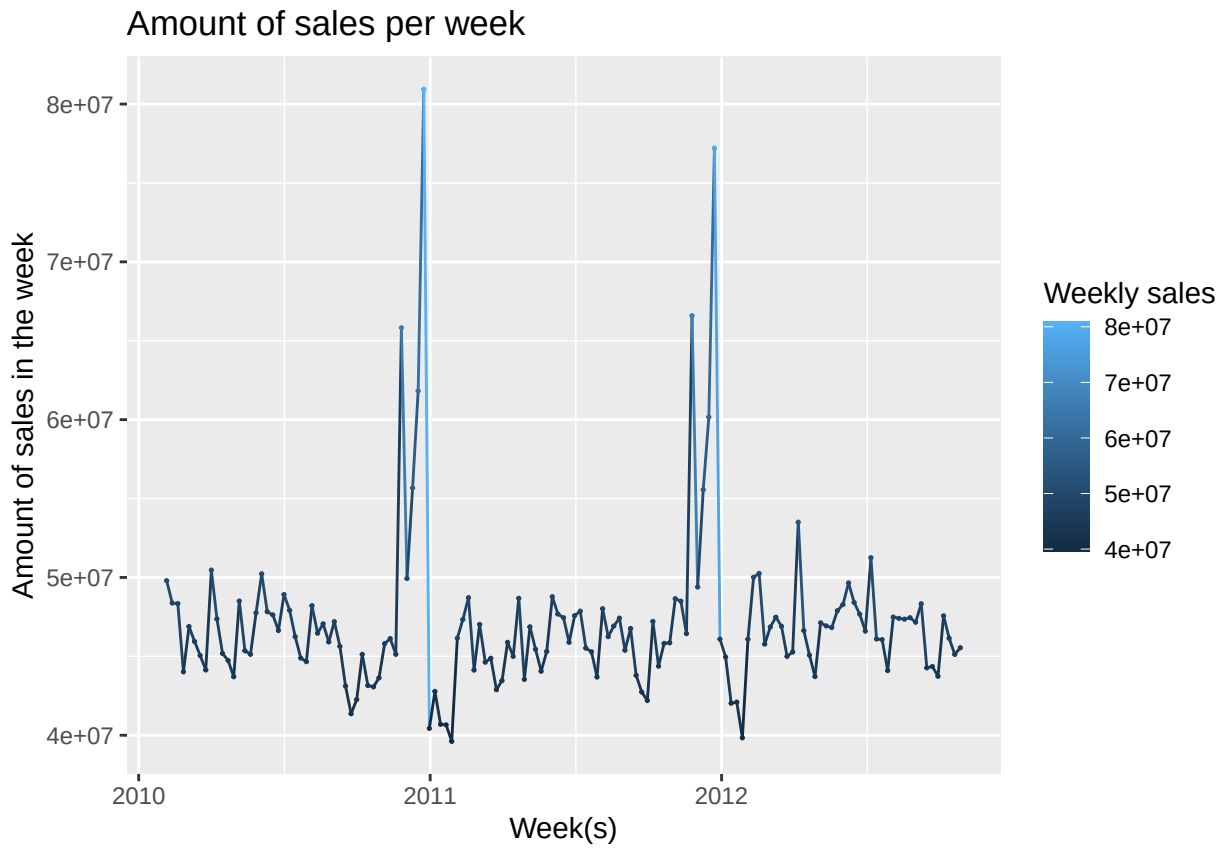
By weekly, given the prediction context I presume we want to lay each week from the start in 2010 to the end in 2012. Date is laid out for this purpose as outlined in Question One, so we group our results by Date specifically and see what happens.

```
weekly_sales_data <- clean_sales_data %>%
  group_by(Date) %>%
  summarise(total_weekly_sales = sum(Weekly_Sales)) %>%
  ungroup()

head(weekly_sales_data)
```

```
## # A tibble: 6 x 2
##   Date       total_weekly_sales
##   <date>                  <dbl>
## 1 2010-02-05          49796770.
## 2 2010-02-12          48377300.
```

```
## 3 2010-02-19          48344298.
## 4 2010-02-26          44020293.
## 5 2010-03-05          46893562.
## 6 2010-03-12          45940318.
```

```
ggplot(data = weekly_sales_data, aes(x = Date, y = total_weekly_sales, colour = total_weekly_sales)) +
  geom_line() +
  geom_point(size=0.25) +
  labs(title = "Amount of sales per week", x = "Week(s)", y = "Amount of sales in the week", col = "Weel
```

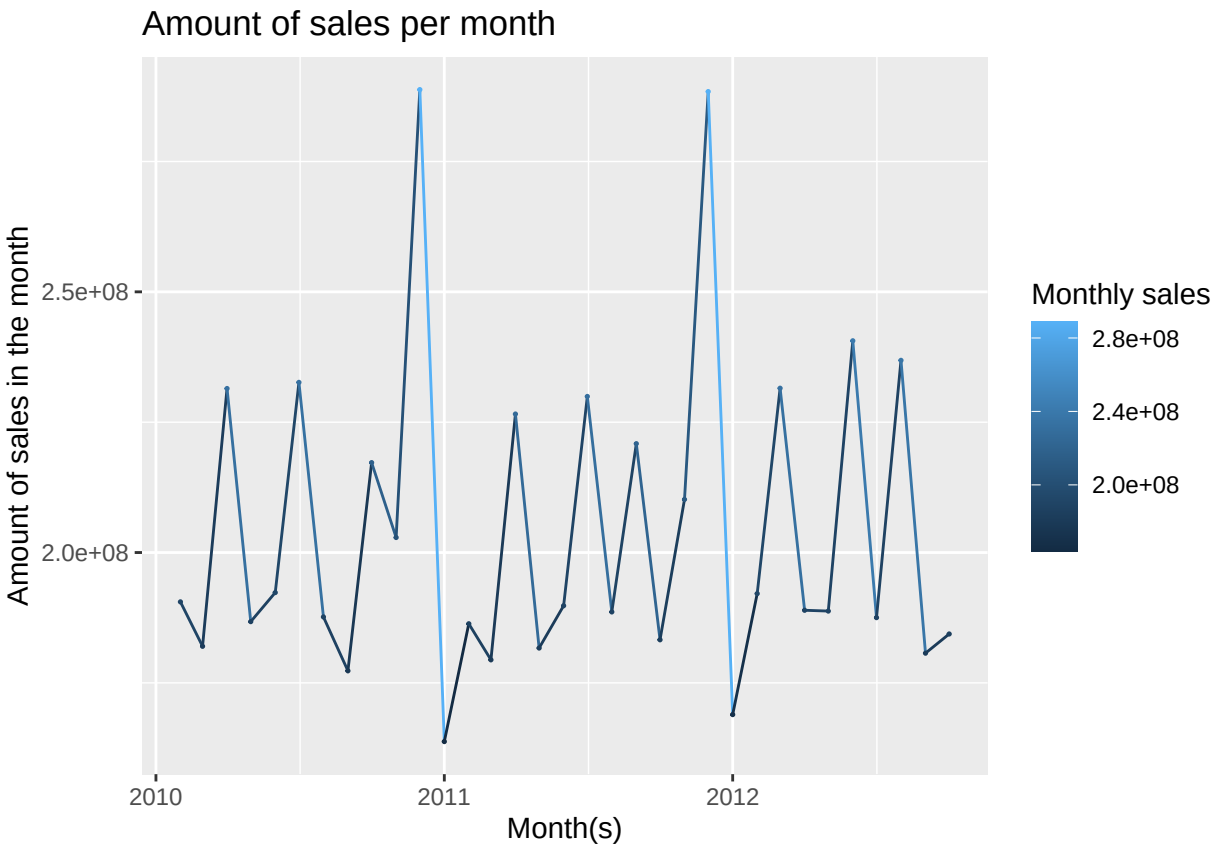



```
rm(weekly_sales_data)
monthly_sales_data <- clean_sales_data %>%
  group_by(year, month) %>%
  summarise(total_monthly_sales = sum(Weekly_Sales)) %>%
  ungroup()

monthly_sales_data <- monthly_sales_data %>%
  mutate(year_month = ym(paste(year, month)))


head(monthly_sales_data)
```

```
## # A tibble: 6 x 4
##    year month total_monthly_sales year_month
##   <dbl> <dbl>               <dbl> <date>
```

```
## 1  2010     2          190538662. 2010-02-01
## 2  2010     3          182018250. 2010-03-01
## 3  2010     4          231459357. 2010-04-01
## 4  2010     5          186747015. 2010-05-01
## 5  2010     6          192321731. 2010-06-01
## 6  2010     7          232634634. 2010-07-01
```

```
ggplot(data = monthly_sales_data, aes(x = year_month, y = total_monthly_sales, colour = total_monthly_sa
  geom_line() +
  geom_point(size=0.25) +
  labs(title = "Amount of sales per month", x = "Month(s)", y = "Amount of sales in the month", col = "
```



The trend in sales volume between weeks mostly mirrors that between months (as you would expect since weeks and months are correlated together). As such, I think it is more worthwhile to talk more about the trend in sales volume between months since this graph is cleaner (between most weeks, the total amount of sales is quite similar except in a few cases where there were sharp rises and falls).

There seems to be a consistent pattern in total sales volume for the retailer between different months in a year, at least between 2010 and 2012 as provided by our data. In both 2010 and 2011 the peak was towards the end of the year. From cross inspection with the weeks graph, I can see that this represents the Christmas and New Year period. This makes sense as a lot of retailers have specific sales in that time which tend to be very popular.

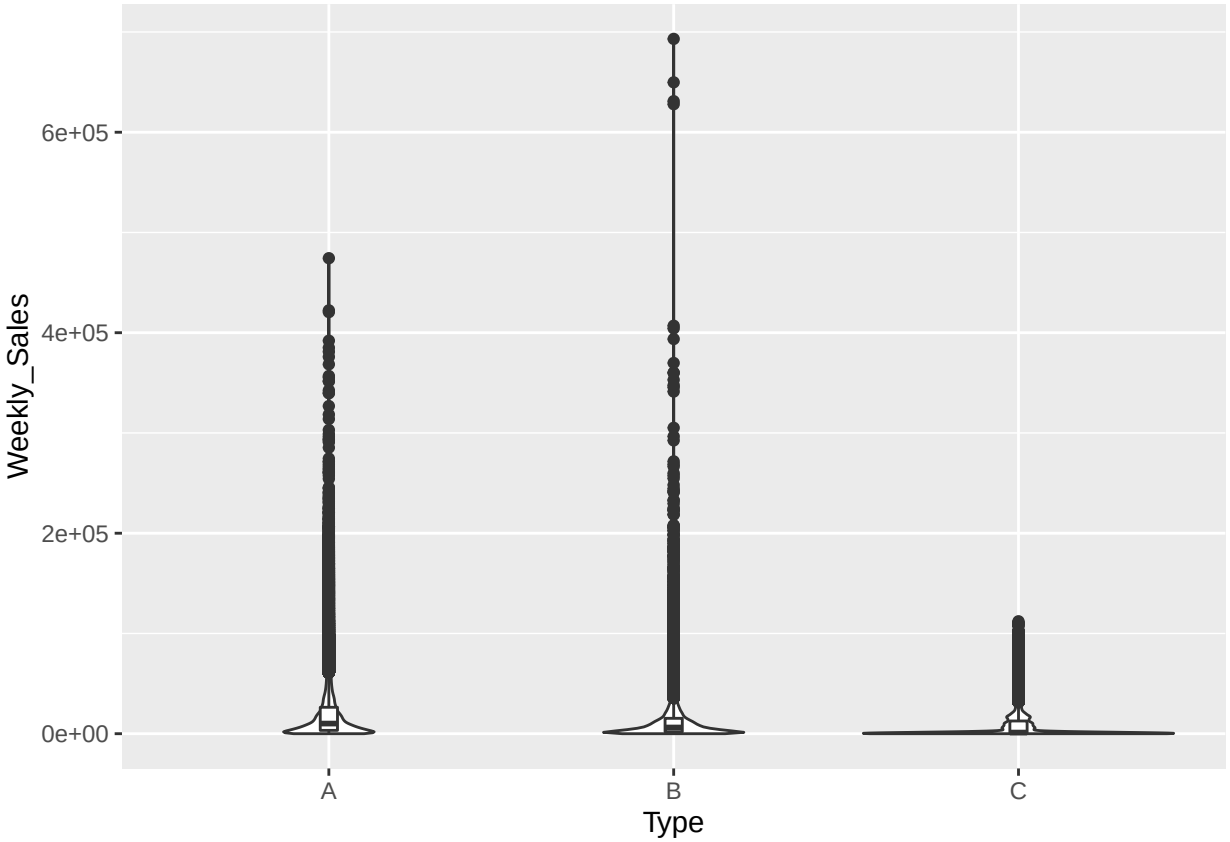
Otherwise I can see three other clear (lower) peaks in each year which seem to be reasonably evenly spaced from each other; which indicates some more seasonality. This makes sense as often when a season ends or starts, there is likely to be a surge in customers, especially if some discount or sale is at play.

It is hard to see too much of an overall trend (in terms of whether it seems that the total sales volume increases or decreases for a certain month in a year as time goes on) but I think there may be a slight overall increase in sales each year. I think this because the 'low point' at the start of 2012 is higher than that of 2011, and the peaks seem to get higher as time goes on.

b) Sales volume by store - explore how sales vary between different store types and sizes

To explain how sales vary between different store types, I used a violin plot as it shows the probability density of different sales volumes for each store type. I also superimposed each corresponding boxplot as this can also provide other useful information (like each store type's median and interquartile range).

```
ggplot(data = clean_sales_data, aes(x = Type, y = Weekly_Sales)) +
  geom_violin() +
  geom_boxplot(width = 0.05)
```



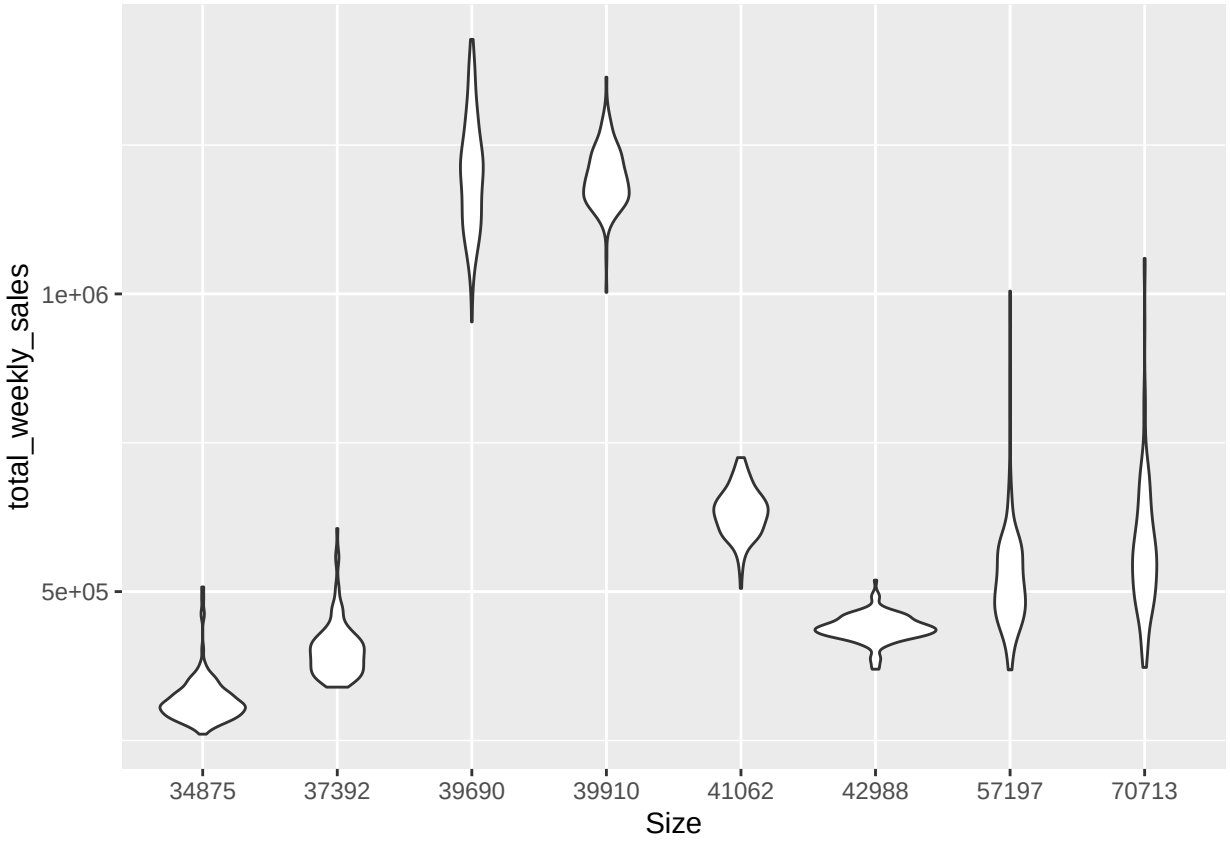
From this plot, I get the sense that shops of Type A tend to get the highest volumes of sale on average (higher median and upper quartile than B and C, less violin density at bottom or 0 sales and consequently more density at higher sales volumes). This trend continues from B to C, where B on average has a greater volume of sales than C for similar reasons as A to B. However, while B on average has smaller sales volumes than A, what is interesting is that B has the highest sales volume recorded from any of the three groups, along with the second and third highest.

Below are my plots depicting how sales vary between shops of different sizes. Please note that I split this into multiple graphs as it would be hard to nicely fit fourty different violin plots in the same graph!

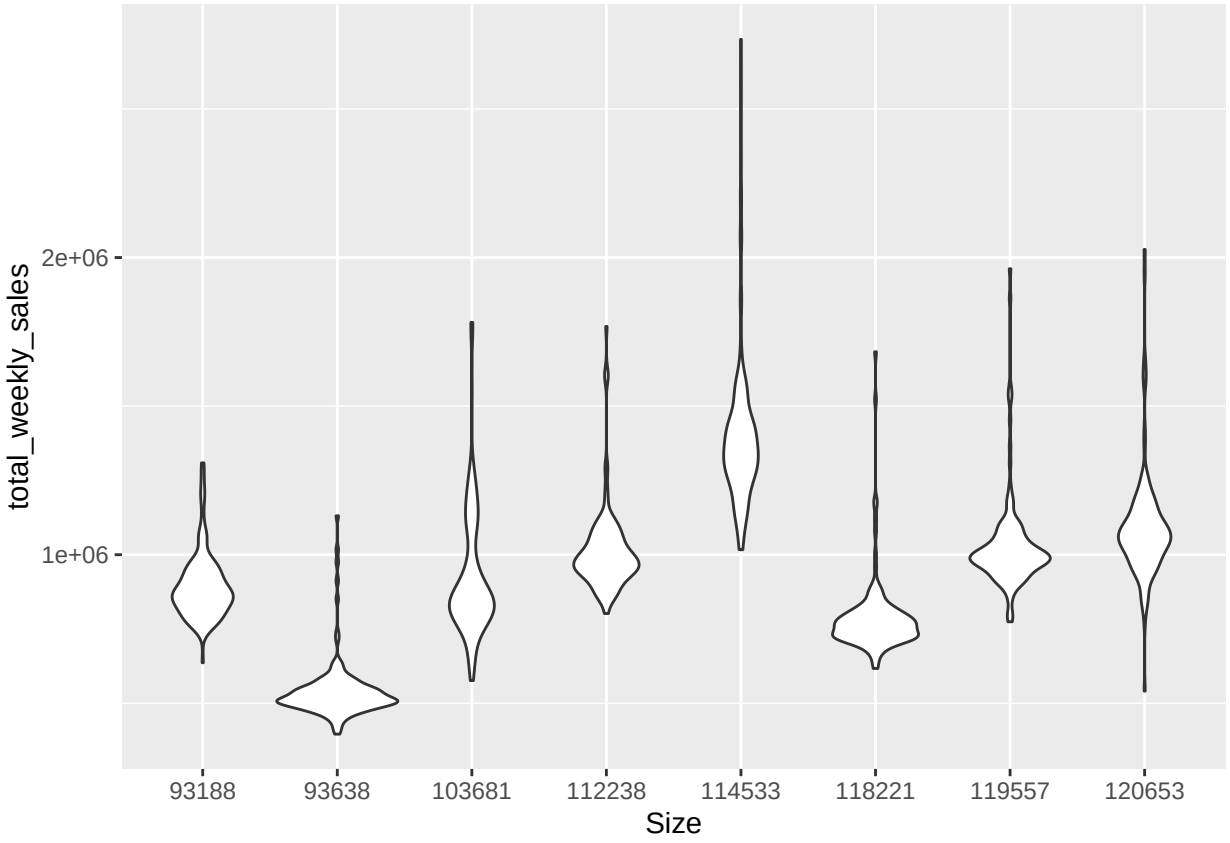
```
rm(monthly_sales_data)
weekly_store_sales_data <- clean_sales_data %>%
  group_by(Date, Size) %>%
  summarise(total_weekly_sales = sum(Weekly_Sales)) %>%
  ungroup()
smallest_group_weekly = weekly_store_sales_data %>%
  filter(Size <= 70713)
second_smallest_group_weekly = weekly_store_sales_data %>%
  filter(Size > 70713 & Size <= 120653)
third_smallest_group_weekly = weekly_store_sales_data %>%
  filter(Size > 120653 & Size <= 155078)
second_biggest_group_weekly = weekly_store_sales_data %>%
  filter(Size > 155078 & Size <= 203007)
biggest_group_weekly = weekly_store_sales_data %>%
  filter(as.integer(Size) > 203007 & as.integer(Size) <= 219622)
smallest_group_weekly$Size = as.factor(smallest_group_weekly$Size)
second_smallest_group_weekly$Size = as.factor(second_smallest_group_weekly$Size)
third_smallest_group_weekly$Size = as.factor(third_smallest_group_weekly$Size)
second_biggest_group_weekly$Size = as.factor(second_biggest_group_weekly$Size)
biggest_group_weekly$Size = as.factor(biggest_group_weekly$Size)

weekly_store_sales_data
```

```
## # A tibble: 5,720 x 3
##    Date        Size total_weekly_sales
##    <date>     <dbl>              <dbl>
##  1 2010-02-05 34875            317173.
##  2 2010-02-05 37392            461622.
##  3 2010-02-05 39690           1176474.
##  4 2010-02-05 39910           1284644.
##  5 2010-02-05 41062            647029.
##  6 2010-02-05 42988            465148.
##  7 2010-02-05 57197            477409.
##  8 2010-02-05 70713            496725.
##  9 2010-02-05 93188            789036.
## 10 2010-02-05 93638            538634.
## # ... with 5,710 more rows
```

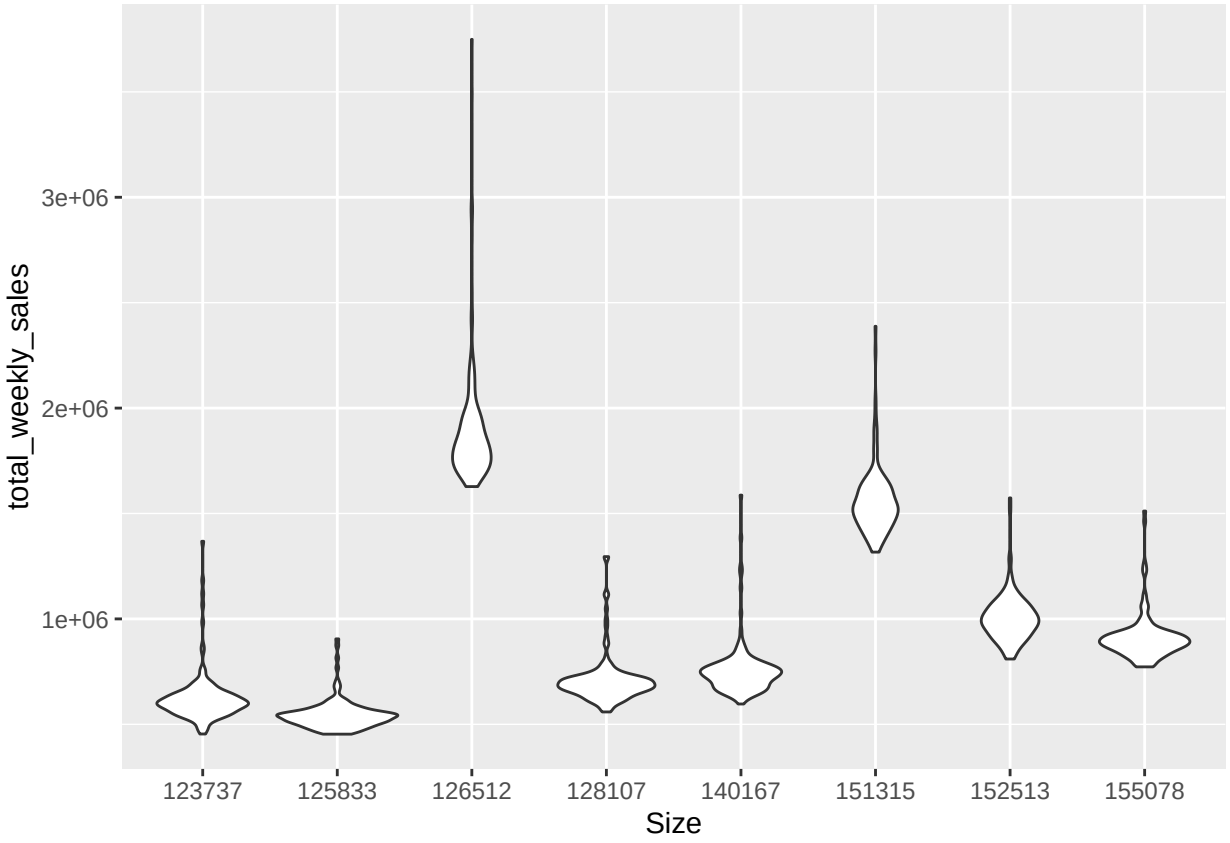
```
ggplot(data = smallest_group_weekly, aes(x = Size, y = total_weekly_sales), col = total_weekly_sales) +
  geom_violin()
```



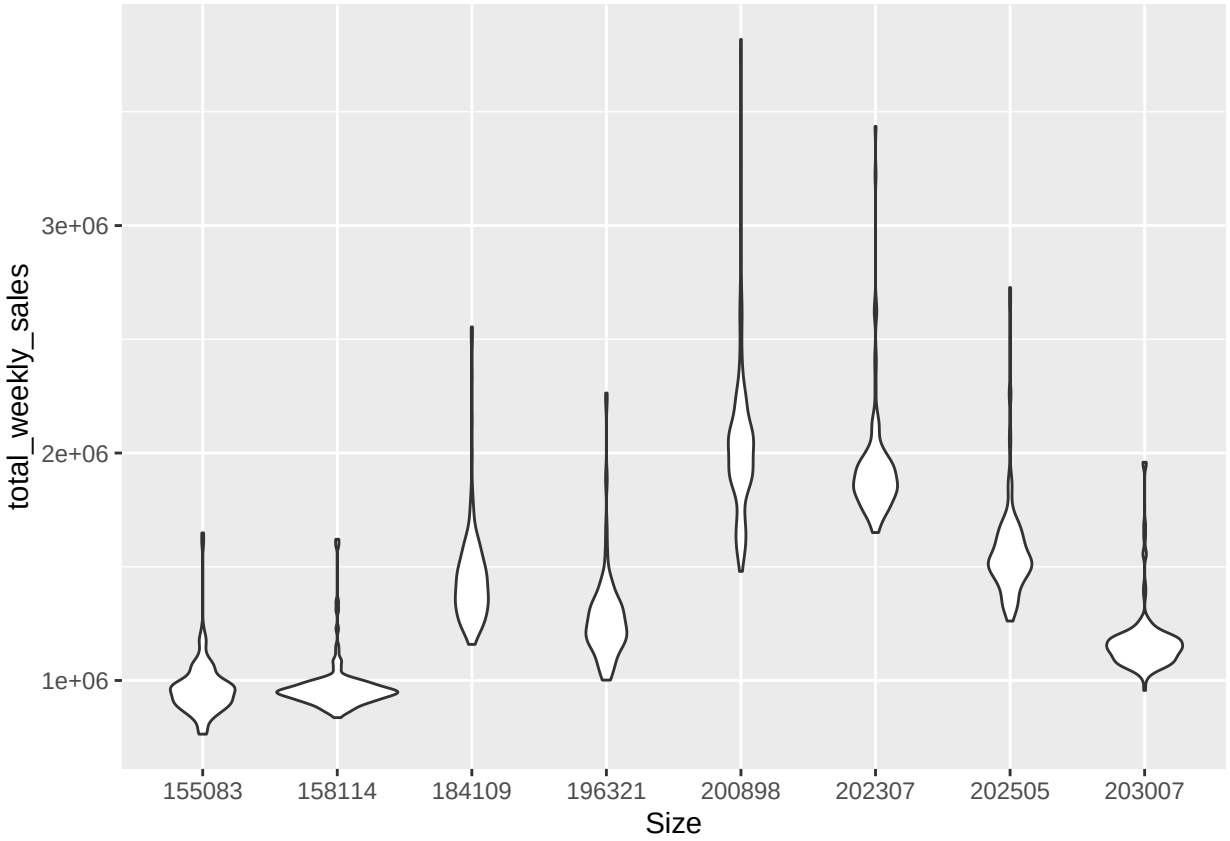

```
ggplot(data = second_smallest_group_weekly, aes(x = Size, y = total_weekly_sales), col = total_weekly_sa
  geom_violin()
```



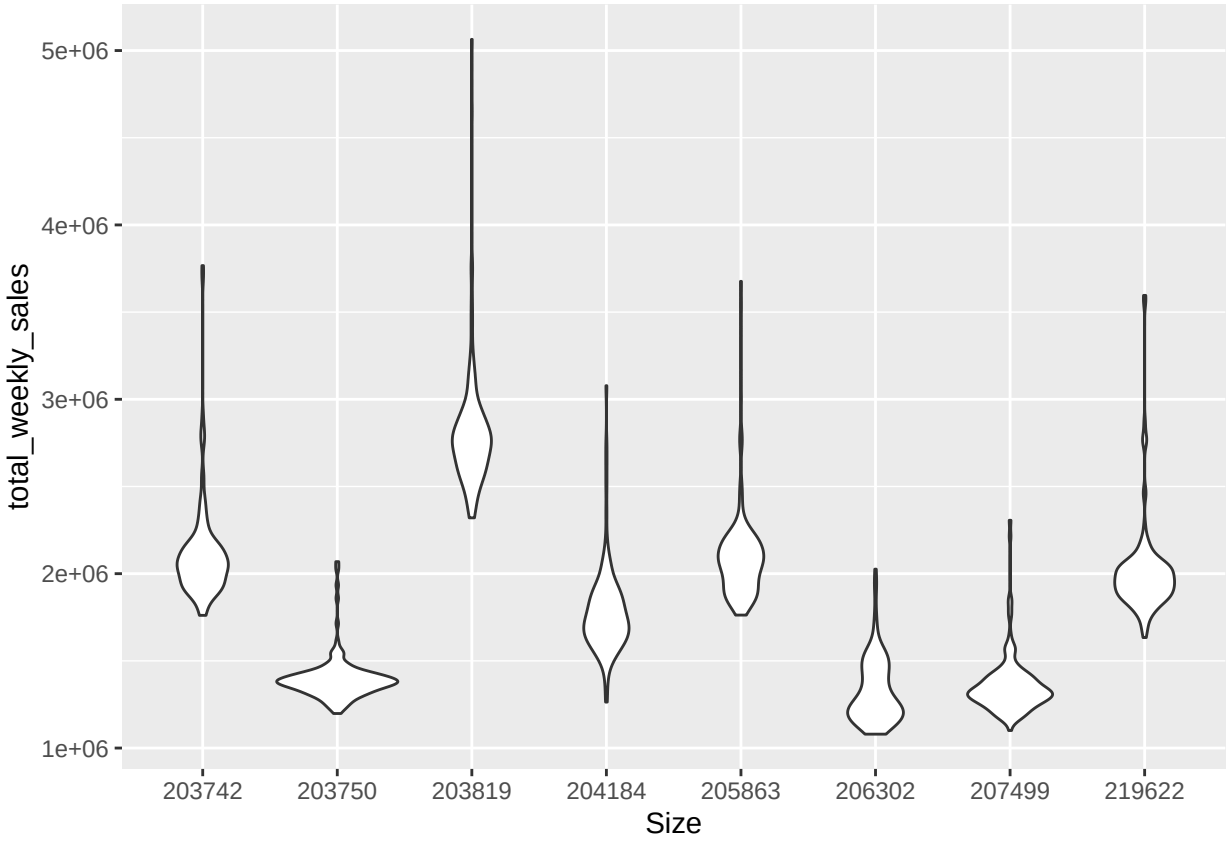

```
ggplot(data = third_smallest_group_weekly, aes(x = Size, y = total_weekly_sales), col = total_weekly_sa
  geom_violin()
```




```
ggplot(data = second_biggest_group_weekly, aes(x = Size, y = total_weekly_sales), col = total_weekly_sa
  geom_violin()
```




```
ggplot(data = biggest_group_weekly, aes(x = Size, y = total_weekly_sales), col = total_weekly_sales) +
  geom_violin()
```



It is admittedly hard to process all this information at once. I did consider methods like a histogram but I didn't want to strip out all the size information. In the interest of time, I settled for violin plot visualisations, however I would experiment with other options that might simplify the viewing experience if I had time.

I separated each size out on the x-axis, where the earlier graphs are for smaller sizes and on each graph size increases from left to right.

A general trend I noticed is that on average, sales volume tends to increase as the shop size increases which should be expected (if you keep other variables constant that is). There are a few clear exceptions to this general trend (look at shop sizes 39690 and 39910 versus 41062) however this is the exception and not the rule. For these shops which have a smaller size than many others yet clearly significantly more sales, they do have multiple shops with the same size as I found earlier (whereas the other sizes only have one). If I had more time, I would try to average this effect out to make a fairer comparison. Otherwise, maybe there is a specific appeal to some products which tends to drive a higher volume of customers than others, even when their stores have a smaller size.

I don't see any particular trend in the distributions of stores over size (bigger or smaller range, or more clustered for example), however it does appear that most shops' sales volumes follow close to a normal distribution.

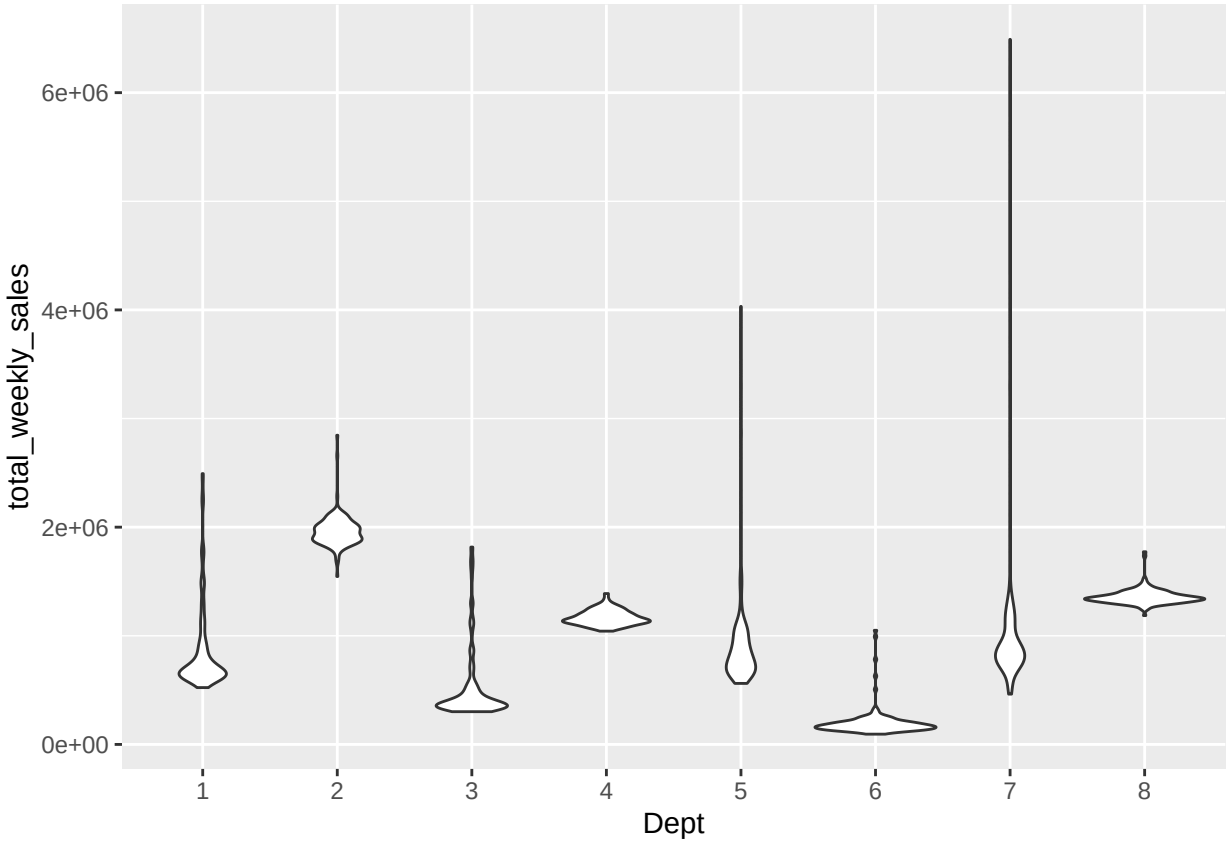
Otherwise, it appears the shop with size 203819 (one of the biggest) tended to have the biggest weekly sales volume out of all.

c) Sales volume by department

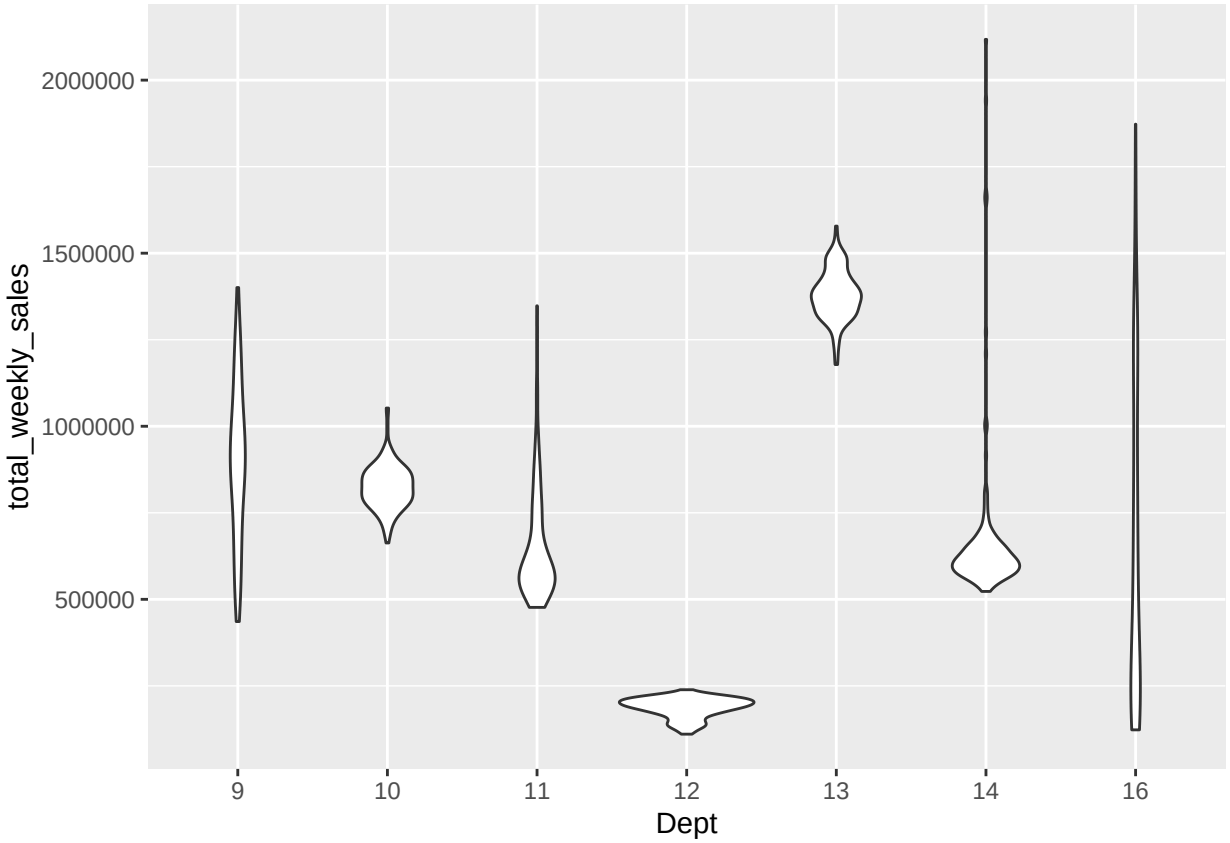
Again, it makes most sense to me to use a violin plot to depict this relationship. Much like size, I will split this up into several plots. I refrained from superimposing a box plot this time though since it often got

messy (probably due to there being more distinct departments than sizes).

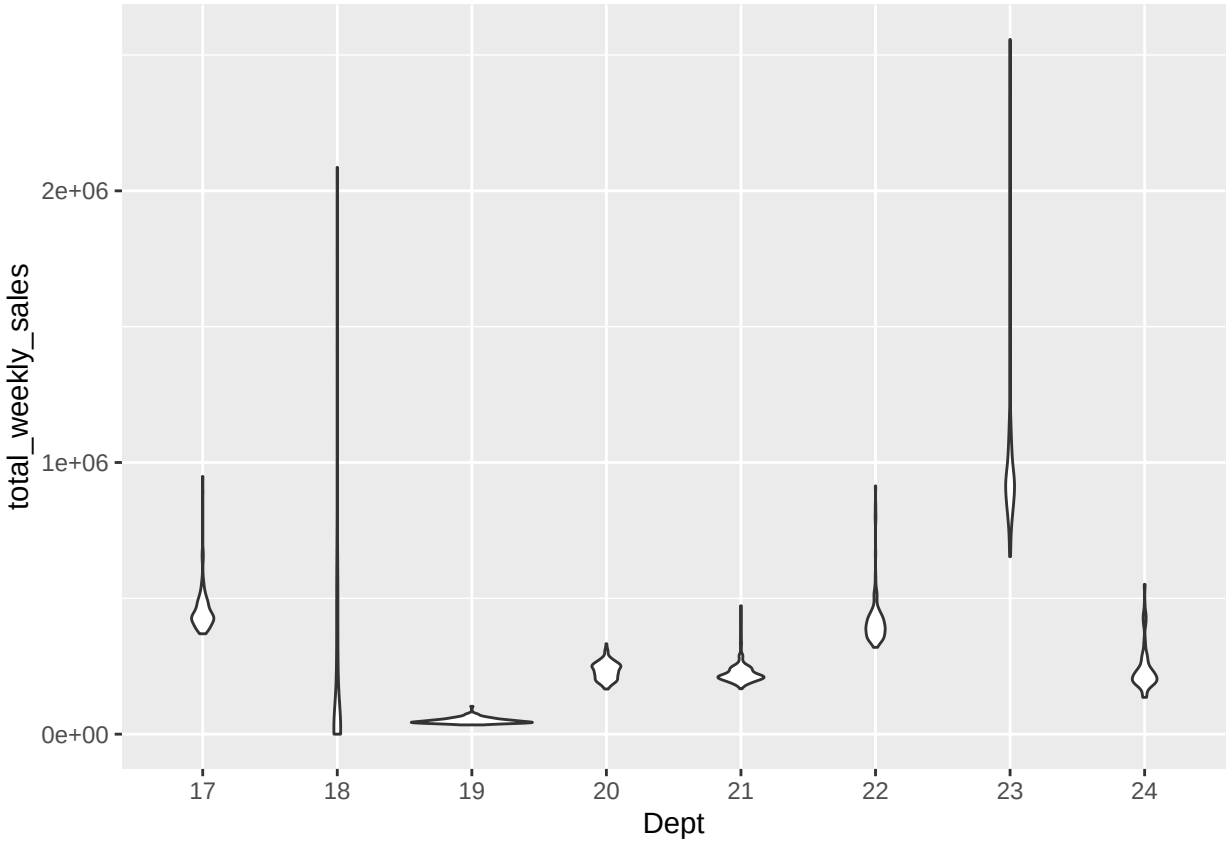
```
weekly_store_sales_data <- clean_sales_data %>%
  group_by(Date, Dept) %>%
  summarise(total_weekly_sales = sum(Weekly_Sales)) %>%
  ungroup
first_group_weekly = weekly_store_sales_data %>%
  filter(Dept <= 8)
second_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 8 & Dept <= 16)
third_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 16 & Dept <= 24)
fourth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 24 & Dept <= 32)
fifth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 32 & Dept <= 40)
sixth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 40 & Dept <= 48)
seventh_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 48 & Dept <= 56)
eighth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 56 & Dept <= 64)
ninth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 64 & Dept <= 72)
tenth_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 72 & Dept <= 90)
eleventh_group_weekly = weekly_store_sales_data %>%
  filter(Dept > 90)


first_group_weekly$Dept = as.factor(first_group_weekly$Dept)
second_group_weekly$Dept = as.factor(second_group_weekly$Dept)
third_group_weekly$Dept = as.factor(third_group_weekly$Dept)
fourth_group_weekly$Dept = as.factor(fourth_group_weekly$Dept)
fifth_group_weekly$Dept = as.factor(fifth_group_weekly$Dept)
sixth_group_weekly$Dept = as.factor(sixth_group_weekly$Dept)
seventh_group_weekly$Dept = as.factor(seventh_group_weekly$Dept)
eighth_group_weekly$Dept = as.factor(eighth_group_weekly$Dept)
ninth_group_weekly$Dept = as.factor(ninth_group_weekly$Dept)
tenth_group_weekly$Dept = as.factor(tenth_group_weekly$Dept)
eleventh_group_weekly$Dept = as.factor(eleventh_group_weekly$Dept)

ggplot(data = first_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



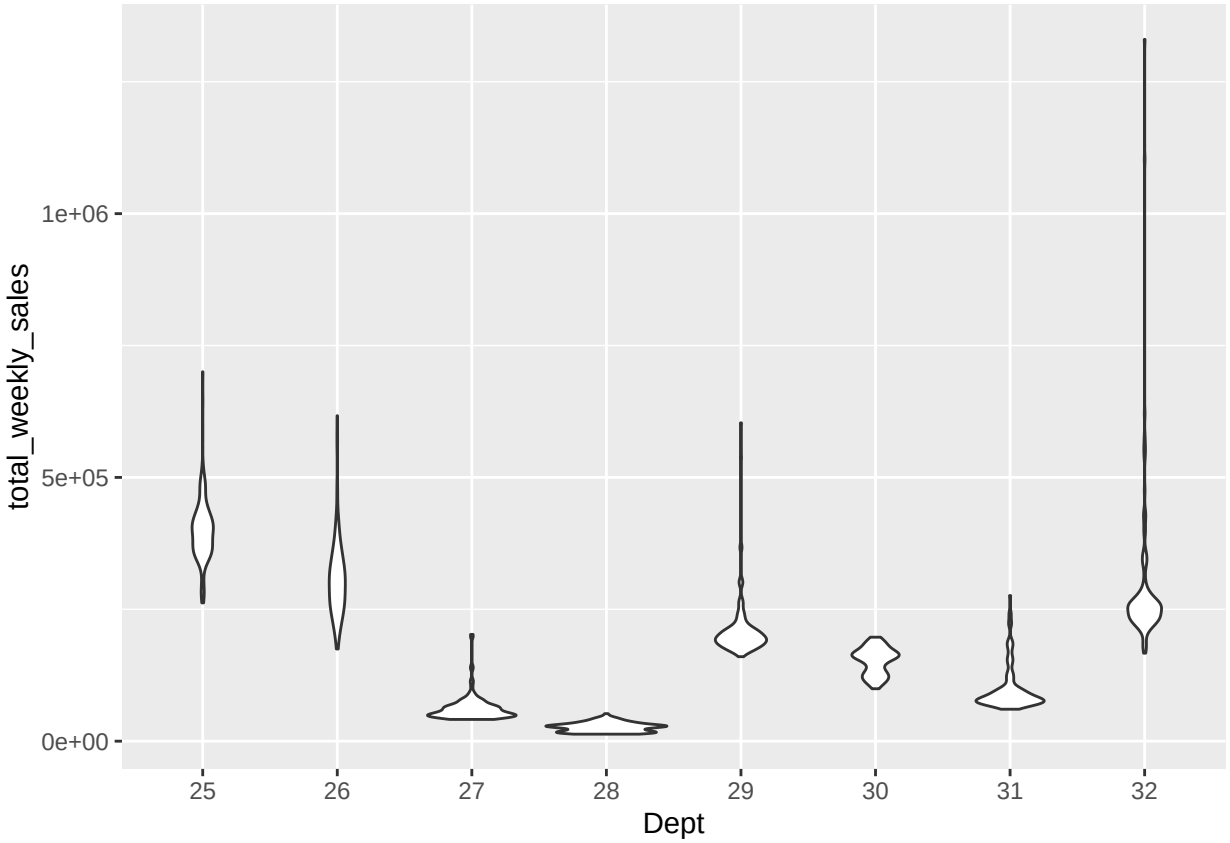

```
ggplot(data = second_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



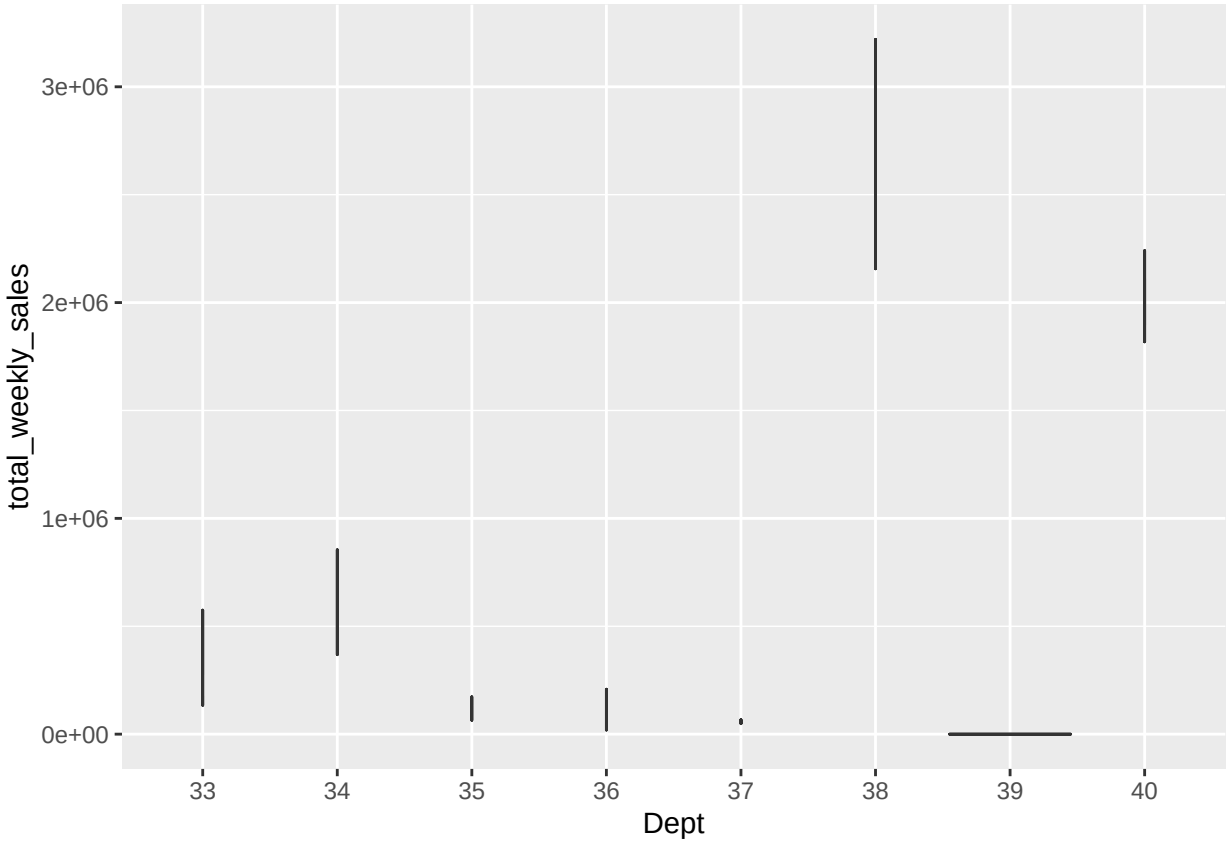

```
ggplot(data = third_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



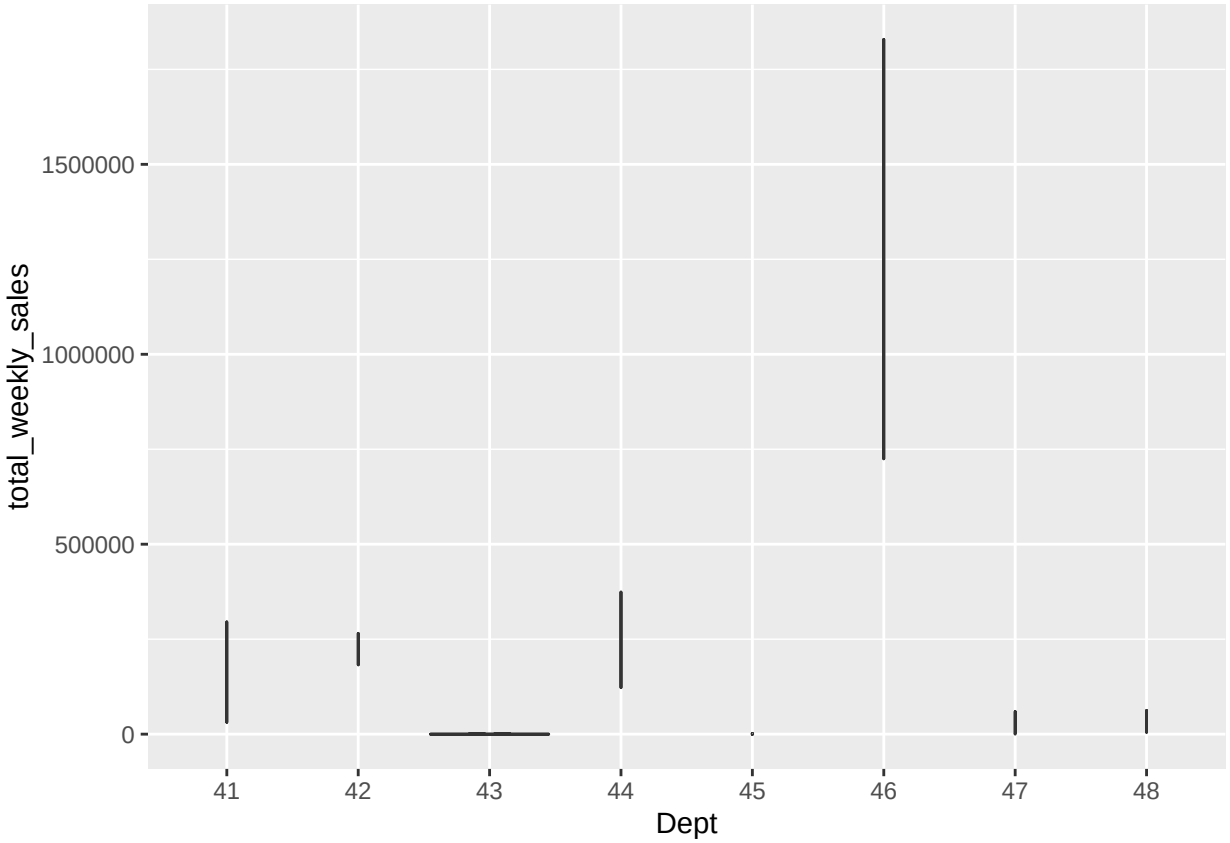

```
ggplot(data = fourth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



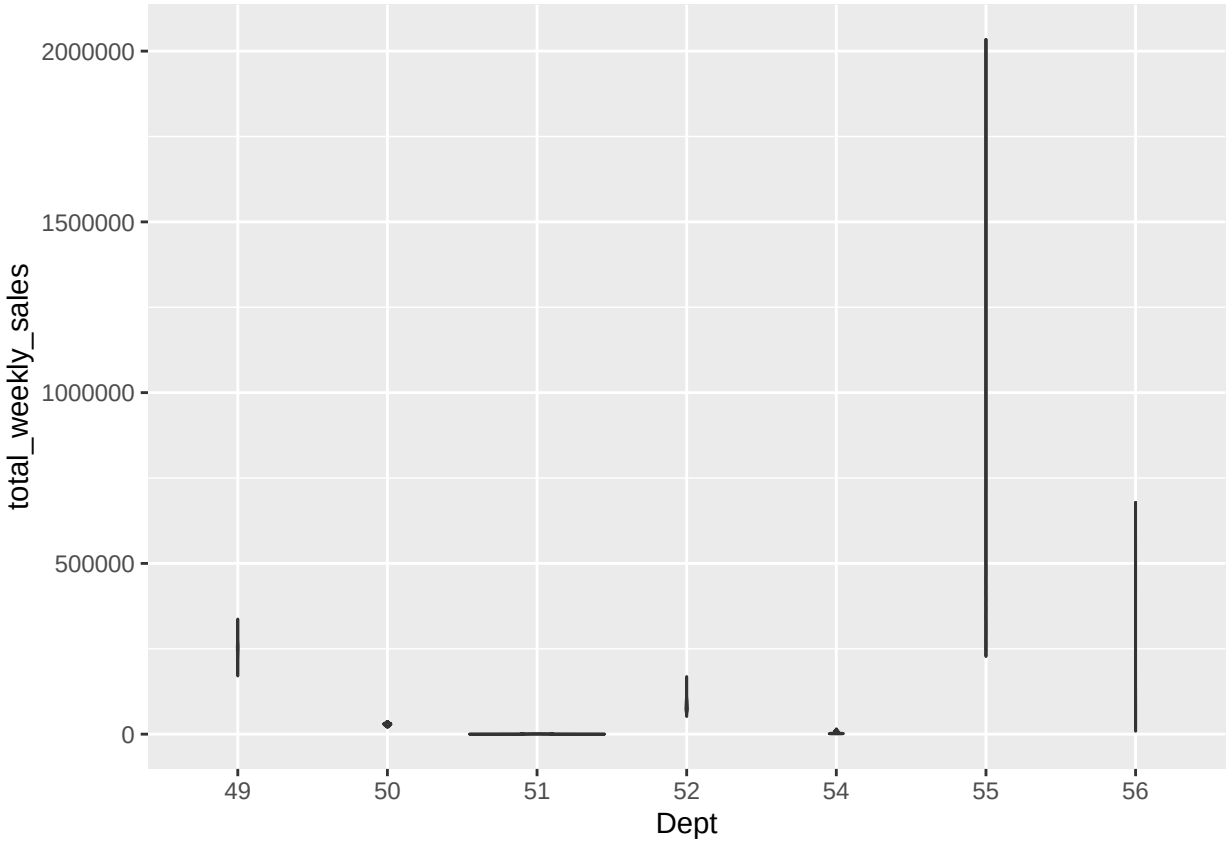

```
ggplot(data = fifth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



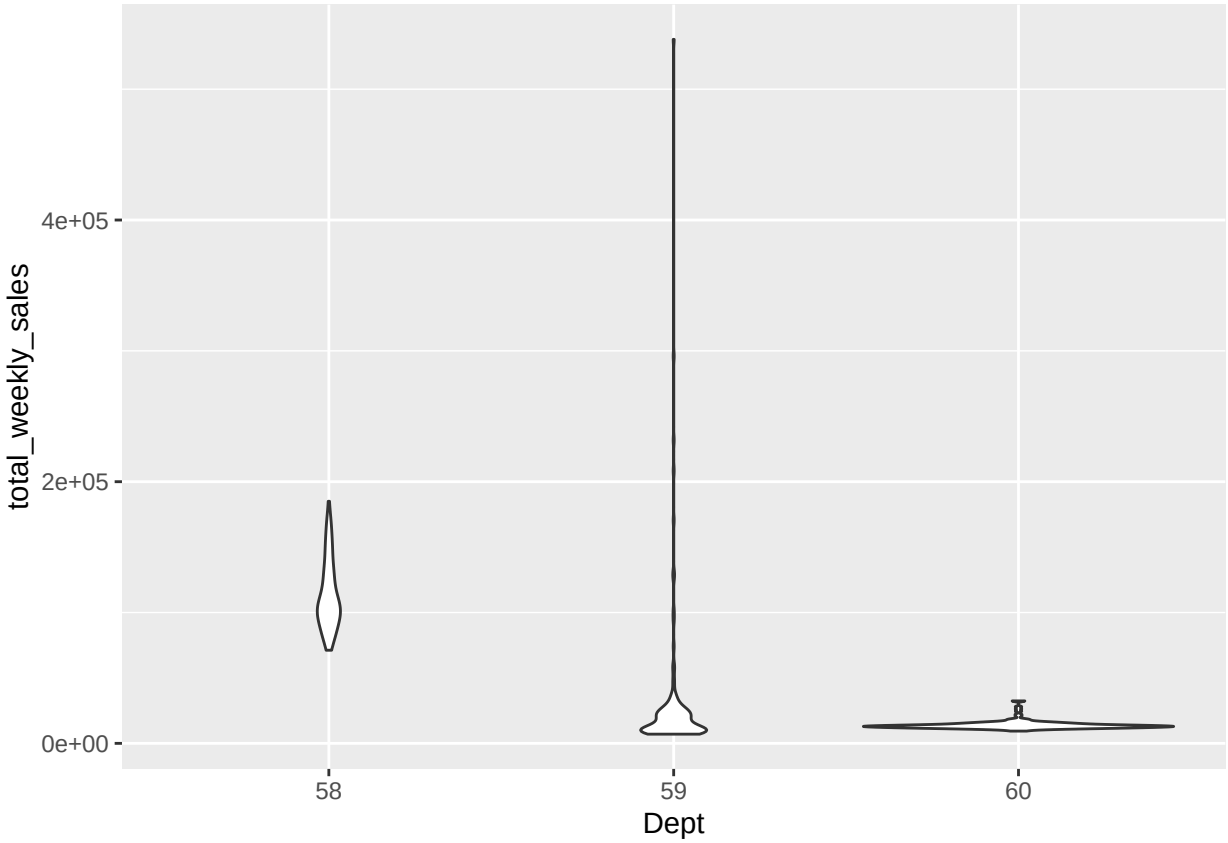

```
ggplot(data = sixth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



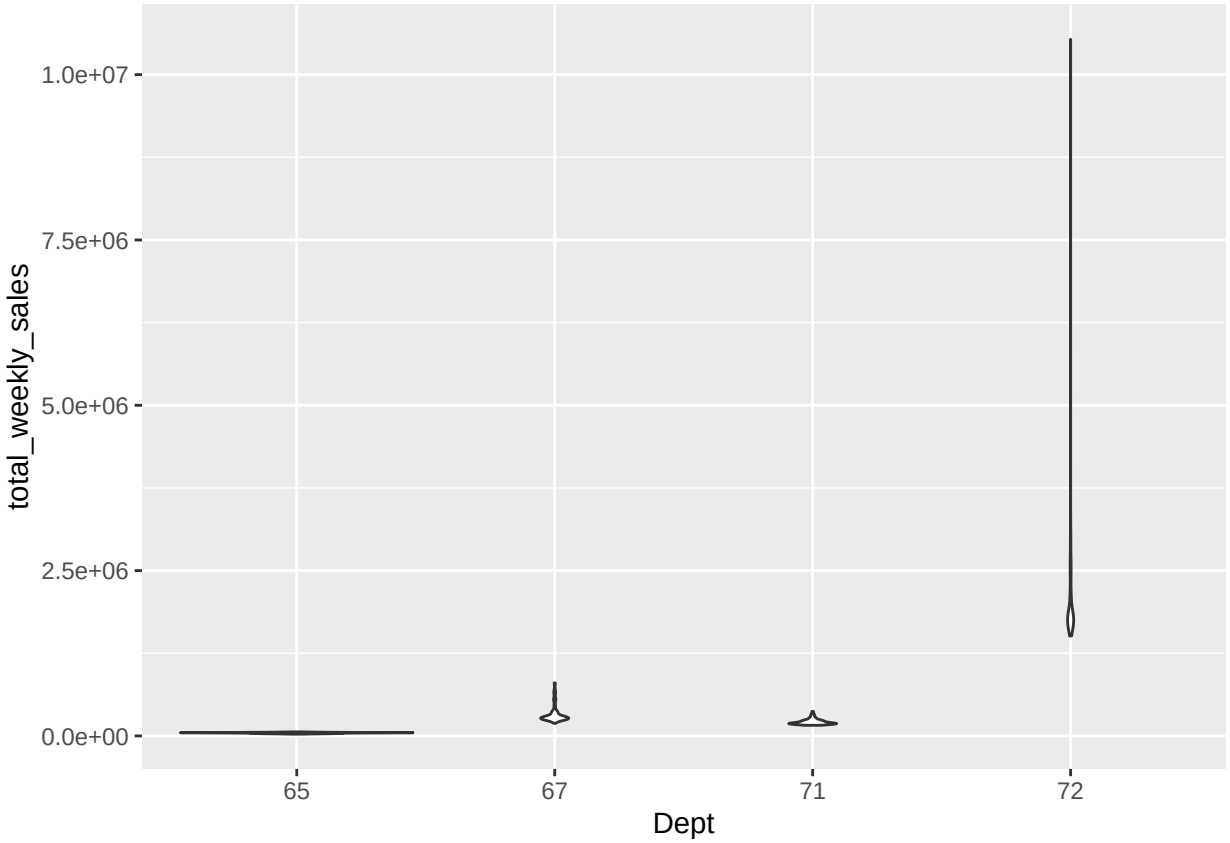

```
ggplot(data = seventh_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



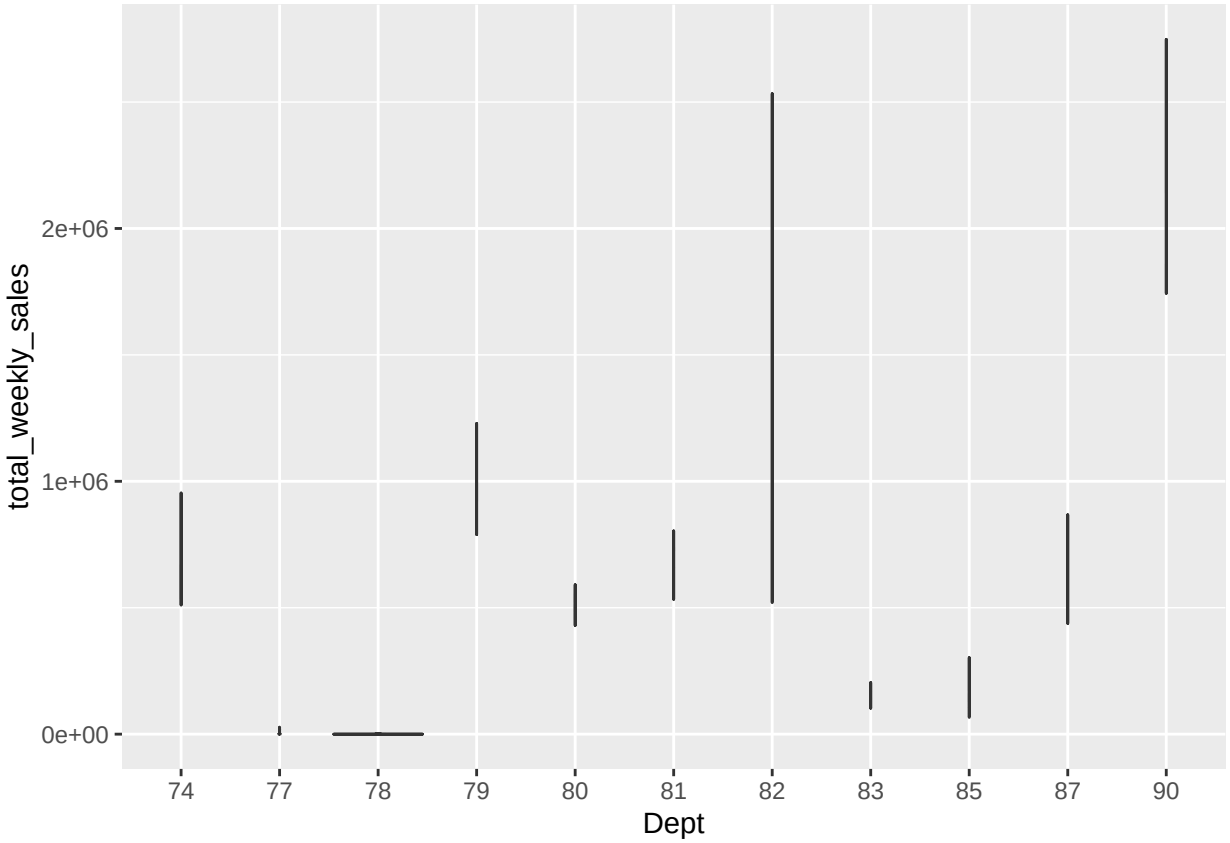

```
ggplot(data = eighth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



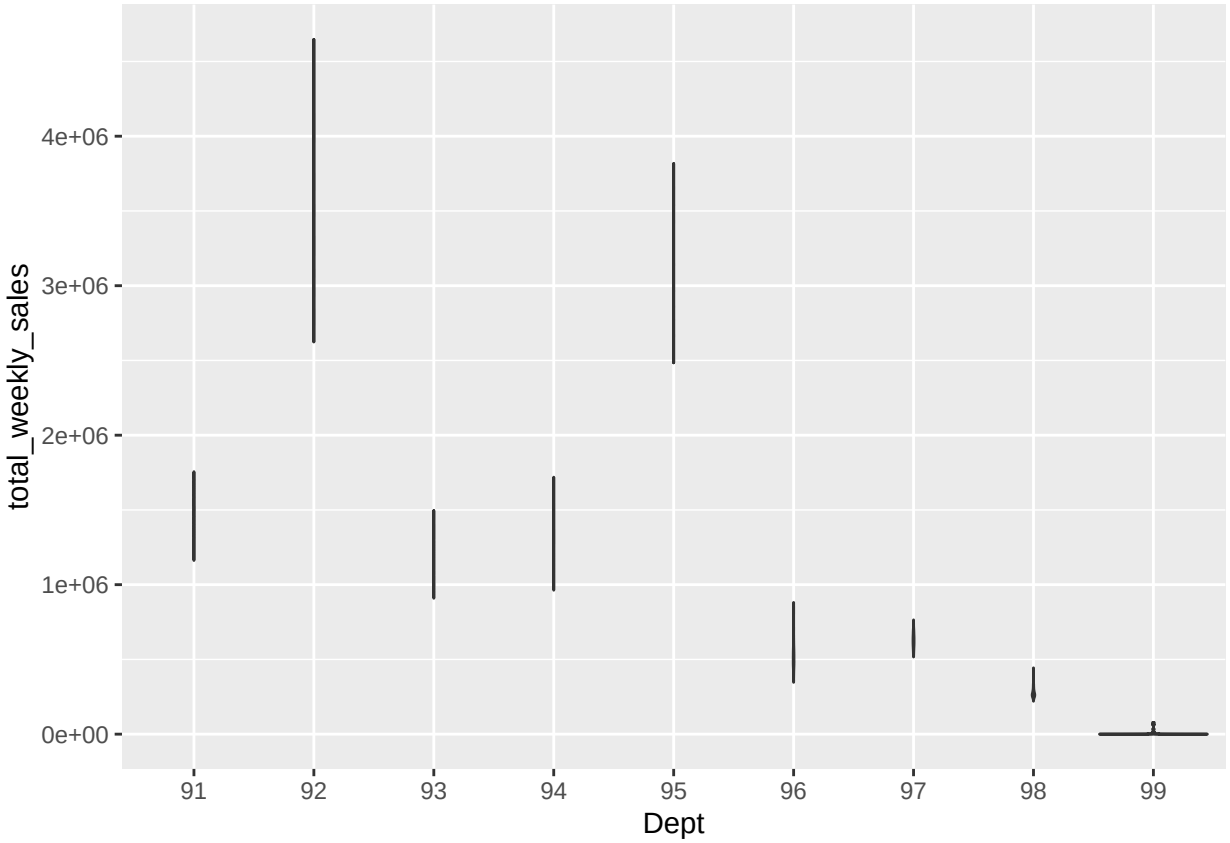

```
ggplot(data = ninth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```




```
ggplot(data = tenth_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```




```
ggplot(data = eleventh_group_weekly, aes(x = Dept, y = total_weekly_sales)) +
  geom_violin()
```



It's very hard to say anything about specific department links, as there are many departments and we don't know what each of them mean. Many of the violins (those above 73, many between 30 and 60) are very thin, because they may not have too many observations, so it is hard to comment on those too specifically, although some have higher average sales volumes than others. The most spread out department (with some reasonably sales volume density) is 16 - it has had some fairly high and low sales volumes reasonably frequently. 72 is the department that has the week with the biggest volume of sales. 60, 28 and 6 for example have had a big density of weeks with zero sales. Departments 9 and 58 appear to be well-balanced. Dept 72 seems to have the highest average sales volume.

Task Three: Investigate the effect of holidays on sales volume. Are all holidays accounted for with the `IsHoliday` flag in the data set?

To answer the latter question, we can assess the dates where 'IsHoliday' is TRUE.

```
sales_data %>% filter(IsHoliday == TRUE) %>% distinct(Date)
```

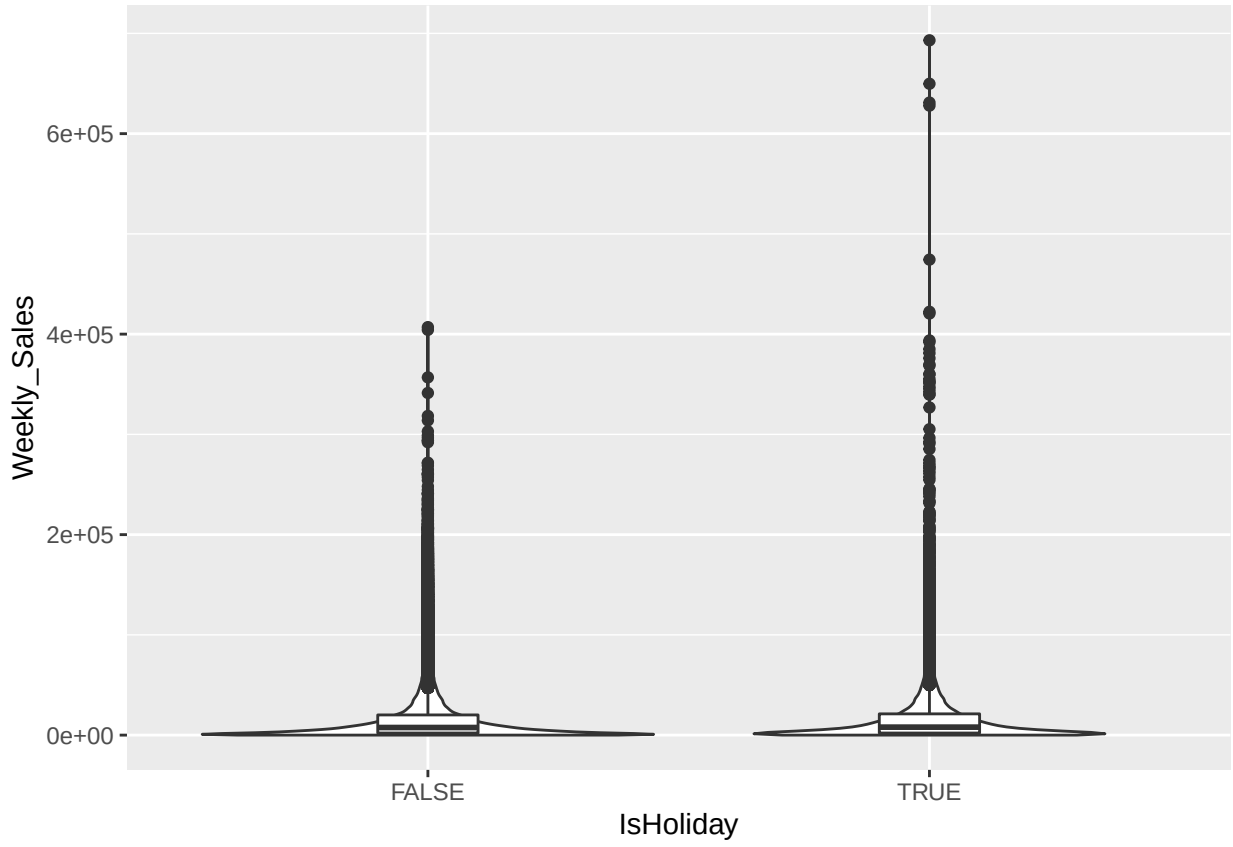
```
## # A tibble: 10 x 1
##    Date
##    <date>
##  1 2010-02-12
##  2 2010-09-10
##  3 2010-11-26
##  4 2010-12-31
##  5 2011-02-11
##  6 2011-09-09
```

```
##  7 2011-11-25
##  8 2011-12-30
##  9 2012-02-10
## 10 2012-09-07
```

From this, we can see that in our dataset, there are only four weeks in a year that are considered to contain at least one 'holiday'. We are not given indication on what the holiday is in each case, so we can infer, even if we don't know where the data is from. We have the end of the year (Christmas/New Year presumably), towards the end of November (from research, perhaps something like Thanksgiving or Hanukkah depending on where the retailer stores are based?), early September (if we assume Thanksgiving is our November holiday, then given that this was taken from America this would probably be Labor Day) and early February (Abraham Lincoln's birthday in America?).

While there is a possibility that these stores are based in America, without comprehensive evidence we can't conclude that so I will avoid speaking specifically about American holidays from here onwards. However, Easter is celebrated quite widely around the world and it clearly is not flagged as a holiday in this data, so there is a good chance that the IsHoliday flag should be true there. Otherwise, depending on the country there probably will be at least a few other days that would be considered public holidays there (such as ANZAC day in New Zealand). However, without specific knowledge of location, it is risky to make specific inferences on what the holidays are so I will proceed without making any changes to the data here. As such, I will comment on what was originally presented by the dataset.

```
ggplot(data = clean_sales_data, aes(x = IsHoliday, y = Weekly_Sales)) +
  geom_violin() +
  geom_boxplot(width = 0.2)
```



The plot here seems to support what we would suspect - which is that days that are flagged as public holidays tend to have a higher volume of sales than days that aren't. They have 0 sales less frequently, and a greater spread at higher volumes of sales than those days that aren't flagged as public holidays. Perhaps this gulf might be even clearer if we added more appropriate public holidays (if there are any). Regardless, the distinction is clear - and the very highest volumes of sales out of any days happen on public holidays.

Task Four: The directors of the retailer want to know how many sales to expect in total for the months of November and December 2012, to inform decision making around their marketing

a) Outline a few different solutions you might use to solve the problem.

The problem is predicting a count - not classifying something. Hence the methods, and consequently metrics we are considering using need to be those that facilitate this type of prediction.

We could use many different models to make predictions about the number of sales. We could use linear/lasso/ridge/polynomial regression, a generalised linear model dealing with count (like Poisson), a neural network, a decision tree, ensemble methods like random trees, or time series specific methods (like ARIMA and Holt-Winters). Below I will discuss methods for regression, generalised linear models and time series models.

The metrics we focus on would depend on what type of model we focus on. For linear regression, we need to focus on assumption checks (like homoscedasticity, points following a relatively normal distribution and observations are independent of each other). If these are not satisfied, we would consider log transformations, and using Box-Cox may help us with this. A generalised additive model is also something to consider if it seems like a linear model is not the best approach. Variance inflation factors or correlation matrices are good to consider when you have many numeric variables - we will have problems if we have multicollinearity in our data. But otherwise, given those are satisfied, model selection is quite important - and for this we tend to consider metrics like AIC and BIC with forwards or backwards selection (but overly relying on this if we have many potential explanatory variables could lead to choosing a model that is overfit). k-fold Cross-Validation helps mitigate these concerns, especially with parameter tuning to help us choose a model that avoids overfitting and might be relatively good at prediction compared to other choices. We also need to train the model on one subset of our data, and test it on the rest (this should help prevent overfitting). After this, we would measure our model's success with things like adjusted R squared (to see if it is good at explaining variation in the data), mean squared prediction error and residual variance.

For a generalised linear model (like Poisson) a heavily similar approach may be followed. The assumptions of a generalised linear model should be satisfied (Variance equals Mean, independence of observations) before we apply one. This often involves checking Pearson and deviance residuals, and sometimes randomised quantile residuals are worth checking too. We do not want to use a generalised linear model if it appears to be over or underdispersed (often easily gauged by checking model residual deviance against degrees of freedom) with the data. But even when some assumptions are not met, we can use bootstrapping to get a strong approximation of our desired distribution so adequacy can be feasibly assessed with the chi-squared test (but something to note is that this requires a lot of computational power and can be time-consuming). Otherwise, AIC/BIC also can help with model selection. Instead of least squares, things like maximum likelihood and deviance can help us estimate how well a model fits to the data (and consequently can make predictions), although we also can use cross-validation as a tool here too.

Time series models tend to be a bit simpler. For example, for ARIMA we just need to see that our data appears to have a stationary trend, and is univariate. Similar for Holt-Winters - we need to see if our data doesn't appear to be random (as this makes exponential smoothing a viable option), has some overall trend and has seasonality. Given these are satisfied, we then would check the residuals of our model and we would check for autocorrelation via the Ljung-Box test. Our model's residuals should be approximately normally distributed and there should be little correlation in residuals. If there are issues with the residual distribution, then a Box-Cox transformation would be good. For autocorrelation, we could either model

the autocorrelation (also by trying a transformation like log transformation) or consider changing models. Otherwise, we can use our model for predictions.

b) Select the method you think is the best approach and explain why.

Out of these, I think time series methods are the best for this problem. I say this because we are specifically making predictions related to time (November and December 2012); not necessary dependent on any specific variable. Given that earlier on, I noticed that there seemed to be some clear seasonality, barely any stochasticity and a reasonably stationary trend with the time series graph, it seems that time series models are likely to work well for our purposes. That is not to say none of the other methods can work as well, if not better. However, I believe that the time series methods can get relatively reliable predictions with much less effort and computational power than any of the other approaches I suggested, which makes them the best starting point for our solution

c) Show an implementation of one of your solutions (doesn't need to be your selected method), and show the final forecast alongside the historical time series

I will try Holt Winters.

```
monthly_sales_data <- clean_sales_data %>%
  group_by(year, month) %>%
  summarise(total_weekly_sales = sum(Weekly_Sales)) %>%
  ungroup()



head(monthly_sales_data)
```

```
## # A tibble: 6 x 3
##    year month total_weekly_sales
##   <dbl> <dbl>              <dbl>
## 1  2010     2         190538662.
## 2  2010     3         182018250.
## 3  2010     4         231459357.
## 4  2010     5         186747015.
## 5  2010     6         192321731.
## 6  2010     7         232634634.
```

```
monthly_sales_data = select(monthly_sales_data, -c("year", "month"))
monthly_sales_data.ts = ts(monthly_sales_data, start = c(2010, 2), end = c(2012, 10), frequency = 12)
head(monthly_sales_data.ts)
```

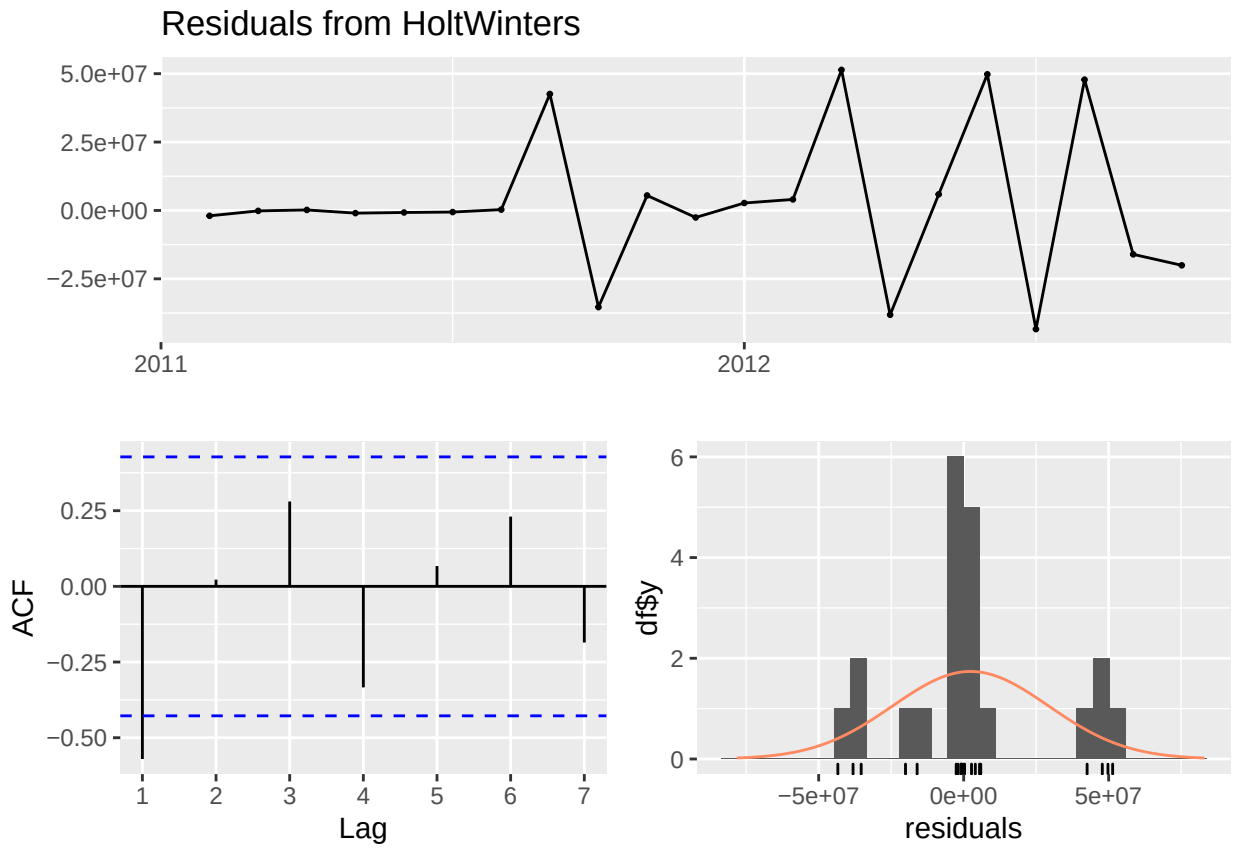
```
##      total_weekly_sales
## [1,]          190538662
## [2,]          182018250
## [3,]          231459357
## [4,]          186747015
## [5,]          192321731
## [6,]          232634634
```

```
library(forecast)
```

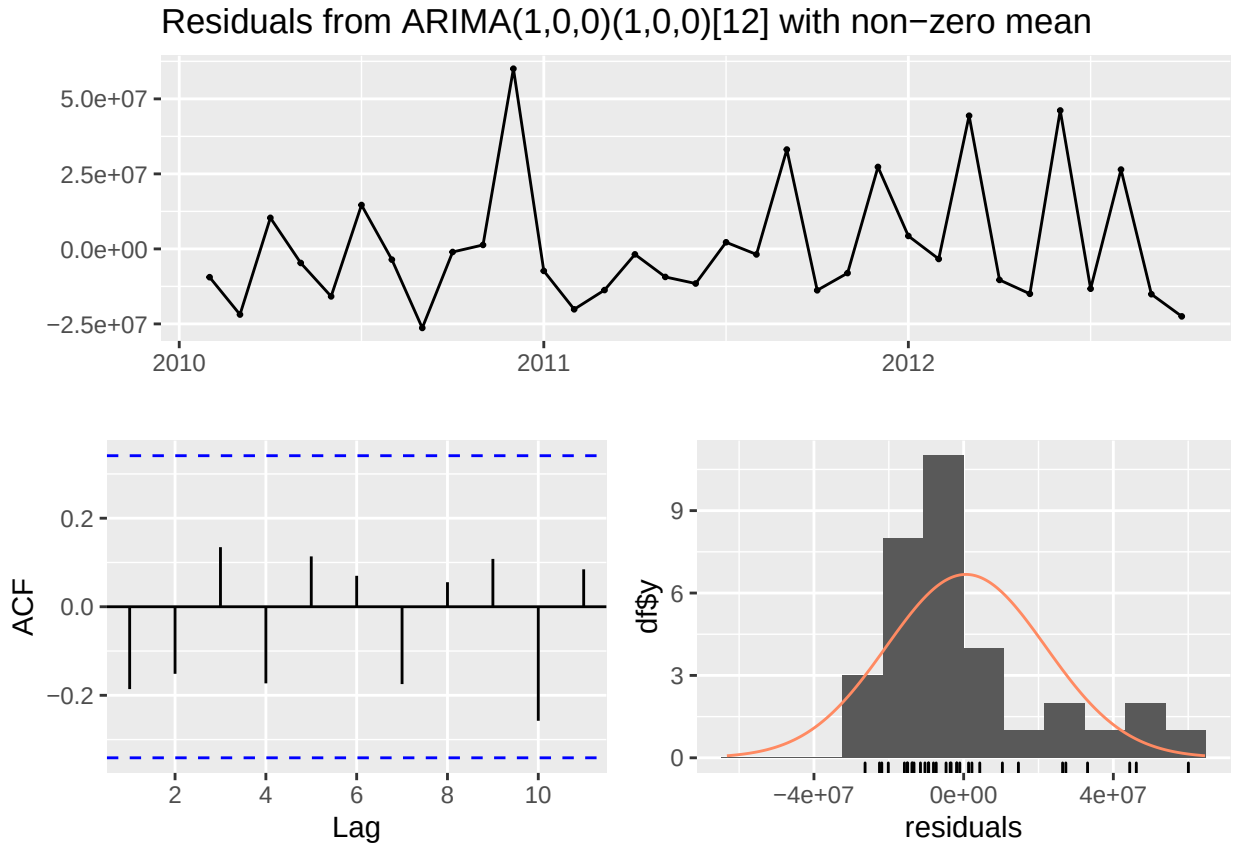
```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
monthly_sales_data.hw = HoltWinters(monthly_sales_data.ts, seasonal="additive")
checkresiduals(monthly_sales_data.hw)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```




```
monthly_sales_data.ar = arima(monthly_sales_data.ts, order = c(1,0,0), seasonal=c(1,0,0))
checkresiduals(monthly_sales_data.ar)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean
## Q* = 6.0921, df = 4, p-value = 0.1924
##
## Model df: 3.   Total lags used: 7
```

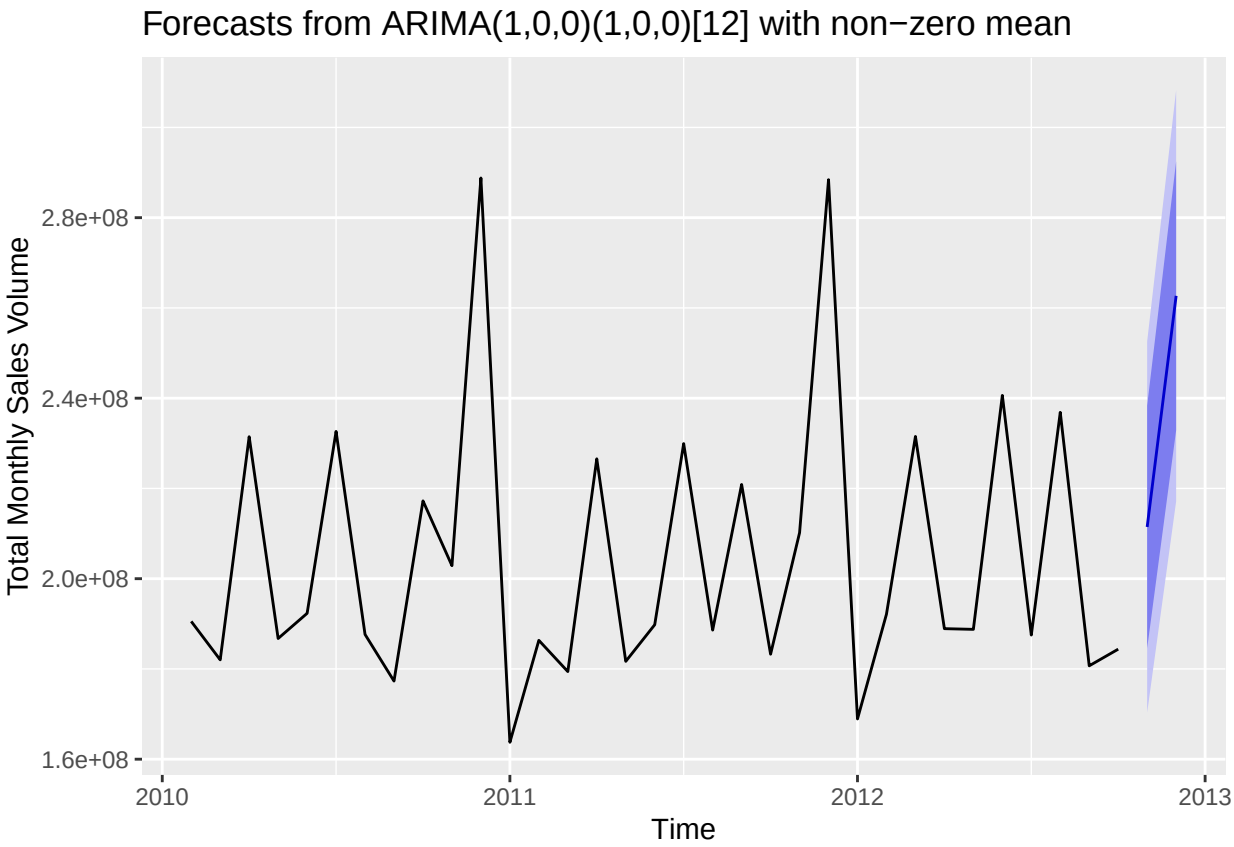
```
preds.hw=predict(monthly_sales_data.hw,n.ahead=2,prediction.interval = TRUE)
preds.ar=predict(monthly_sales_data.ar,n.ahead=2,prediction.interval = TRUE)
preds.hw
```

```
##                fit       upr       lwr
## Nov 2012 207618521 260289090 154947951
## Dec 2012 290527356 343197926 237856787
```

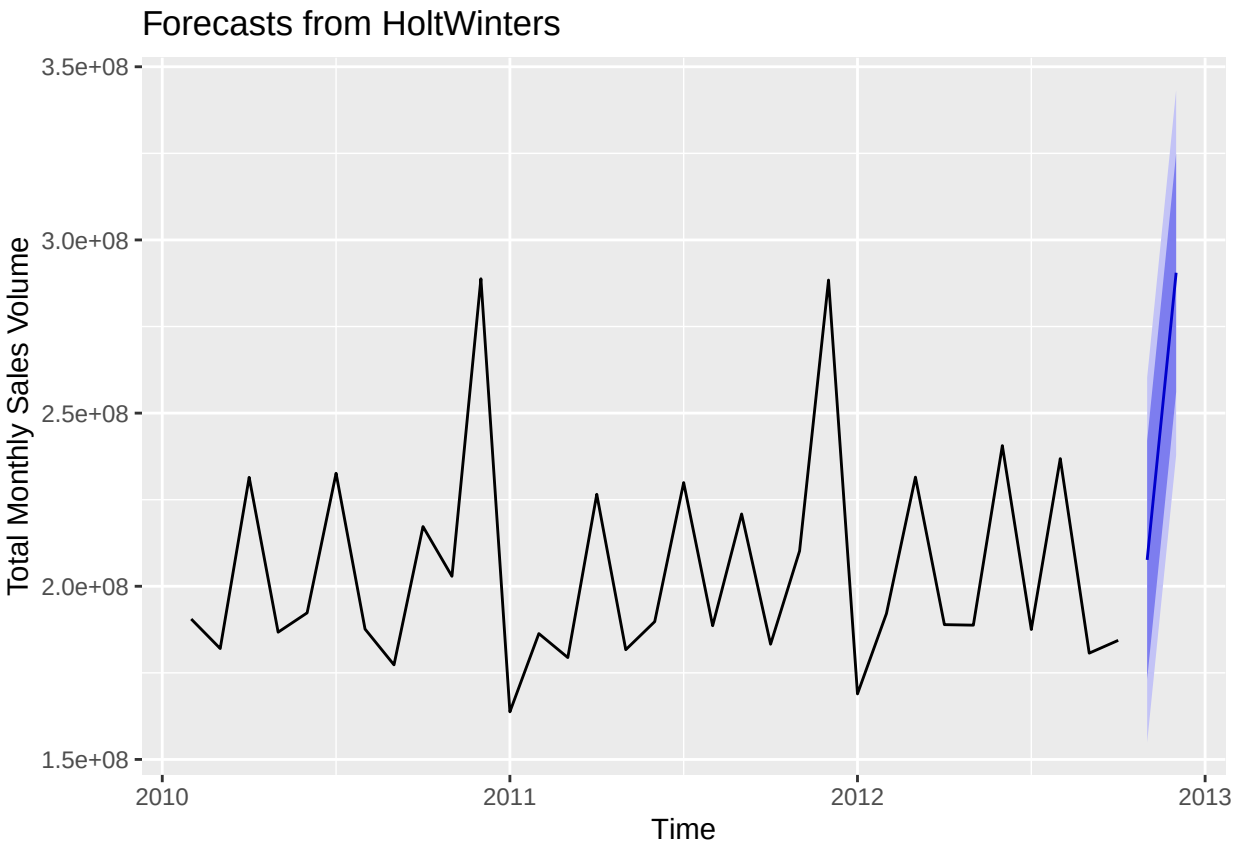
```
preds.ar$pred
```

```
##            Nov       Dec
## 2012 211463141 262679683
```

```
monthly_sales_data.ar %>% forecast(h=2) %>% autoplot() + ylab("Total Monthly Sales Volume") + xlab("Tim
```




```
monthly_sales_data.hw %>% forecast(h=2) %>% autoplot() + ylab("Total Monthly Sales Volume") + xlab("Time
```



In terms of Holt-Winters, there were some concerns with autocorrelation so I fitted a first-order ARIMA too for comparison, but inspected its predictions regardless. I think its predictions (point estimates of about 208 million and 291 million for November 2012 and December 2012 respectively) are in line with 2011 and 2010 (in each of November and December) so are reasonable. However, maybe a transformation of the data may be necessary to better fit a Holt-Winters model due to its autocorrelation.

In terms of ARIMA, there was no concerns with autocorrelation and residual normality. while the prediction (211 million November 2012 and 262 million December 2012) doesn't seem too unreasonable, I believe there is room for improvement as 260 million for December 2012 feels like an underestimation especially when you look at its forecast plot (it appears that in December 2011 and 2010, we had around 280 million sales so this may be an underestimate). While it appears 280 million for December 2012 is in the prediction interval, it definitely feels like there could be room for improvement, and Holt-Winters may have actually done a better prediction in this regard, despite its autocorrelation concerns.

If we were to stay with ARIMA, changing the order of the autoregressive model and season (perhaps even considering AIC for comparison) could help us select a model that better fits the data and potentially makes better predictions. Otherwise, we also could predict with other time-series models such as the Harmonic model and compare results. We could also consider non-temporal variables (like when Holidays are, Date, Dept), apply something like a Random Forest, Linear Regression or a Neural Network to fit to that data (as mentioned earlier on), add all the separate predictions together and see if this better fits the data.

Having cleaner data (that is, make it even more representative than what I made it by cleaning) could help improve the precision of the forecast in general. Otherwise, having more years (if applicable or possible) in the data would help clarify any annual patterns, since two years is not a lot to go by in this regard.