Class 3 Exercise

Swati Singh

2/26/2019

Scenario

You work as a data scientist at a company that sells widgets. The CEO and owner is extremely engaged in looking at the most recent data on sales but is not a statistician and is prone to pay too much attention to meaningless day-to-day and month-to-month fluctuations.

January 2, 8 AM: The CEO comes into your office and expresses worry about widget sales for the most recent month, December. She thinks sales have tanked and wants you to look into the situation further and provide a brief report by noon. Widget demand is seasonal and the business depends on strong holiday sales. She wants a brief report on her desk by noon.

Topics

1. EDA workflow
2. Practice data manipulation and visualization
3. Statistical inference
4. Communication

Process

- Please gather into groups to work on this project!
- Put group members' names up above in the yaml heading under "author" (where it currently says "Names of those in your group").
- Collaborate on one document.
- When you are done, compile to HTML (or PDF), and submit through Canvas.

EDA workflow

1. Formulate a question: Are last month's sales (month 12 of year 5) down?
2. Read in your data. For this exercise we will simulate a dataset.

```
widget <- expand.grid(year = 1:5, month = 1:12, day = 1:30)

# The expand.grid() function creates a dataframe with unique combinations of the values
# from each variable. Here we are setting up a data.frame for 5 years of data.

head(widget)
```

```
##   year month day
## 1    1     1   1
## 2    2     1   1
## 3    3     1   1
## 4    4     1   1
## 5    5     1   1
## 6    1     2   1
```

```r
set.seed(1126) # Use set.seed() to ensure identical datasets

# Now, simulate sales data using the uniform distribution and the normal distribution:
# runif() and rnorm().  For simplicity we will pretend each month has exactly
# 30 days.

widget %<>%
  mutate(sales = ifelse(month < 12,
                        runif(5 * 11 * 30, min = 800, max = 1200) +
                          rnorm(5 * 11 * 30, mean = 100, sd = 100),
                        runif(5 * 11 * 30, min = 1100, max = 1300) +
                          rnorm(5 * 11 * 30, mean = 100, sd = 100)),
         sales = ifelse(month == 12 & year == 5,
                        sales - rnorm(30, mean = 100, sd = 30),
                        sales),
         year = factor(year),
         month = factor(month)) %>%
  arrange(year, month, day) %>%
  mutate(instance = 1:(5*12*30)) # instance is a row counter.

head(widget)
```

```
##   year month day     sales instance
## 1    1     1   1 1279.7602        1
## 2    1     1   2 1286.5776        2
## 3    1     1   3  915.0599        3
## 4    1     1   4 1046.2471        4
## 5    1     1   5 1326.8424        5
## 6    1     1   6 1254.3826        6
```

3. Check the packaging: dim(), nrow(), ncol()

```r
dim(widget)
```

```
## [1] 1800    5
```

```r
nrow(widget)
```

```
## [1] 1800
```

```r
ncol(widget)
```

```
## [1] 5
```

4. Inspect the dataset: str(), glimpse(), View()

```r
str(widget)
```

```
## 'data.frame':    1800 obs. of  5 variables:
##  $ year    : Factor w/ 5 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ month   : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ day     : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ sales   : num  1280 1287 915 1046 1327 ...
##  $ instance: int  1 2 3 4 5 6 7 8 9 10 ...
```

```r
glimpse(widget)
```

```
## Observations: 1,800
```

```
## Variables: 5
## $ year     <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ month    <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ day      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
## $ sales    <dbl> 1279.7602, 1286.5776, 915.0599, 1046.2471, 1326.8424,...
## $ instance <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...
```

```
view(widget)
```

5. Look at the top and the bottom of your data: head(), tail()

```
head(widget)
```

```
##   year month day     sales instance
## 1    1     1   1 1279.7602        1
## 2    1     1   2 1286.5776        2
## 3    1     1   3  915.0599        3
## 4    1     1   4 1046.2471        4
## 5    1     1   5 1326.8424        5
## 6    1     1   6 1254.3826        6
```

```
tail(widget)
```

```
##      year month day    sales instance
## 1795    5    12  25 1193.226     1795
## 1796    5    12  26 1117.642     1796
## 1797    5    12  27 1243.048     1797
## 1798    5    12  28 1099.529     1798
## 1799    5    12  29 1283.868     1799
## 1800    5    12  30 1021.569     1800
```

6. Summarize the data: summary(), table(), hist()

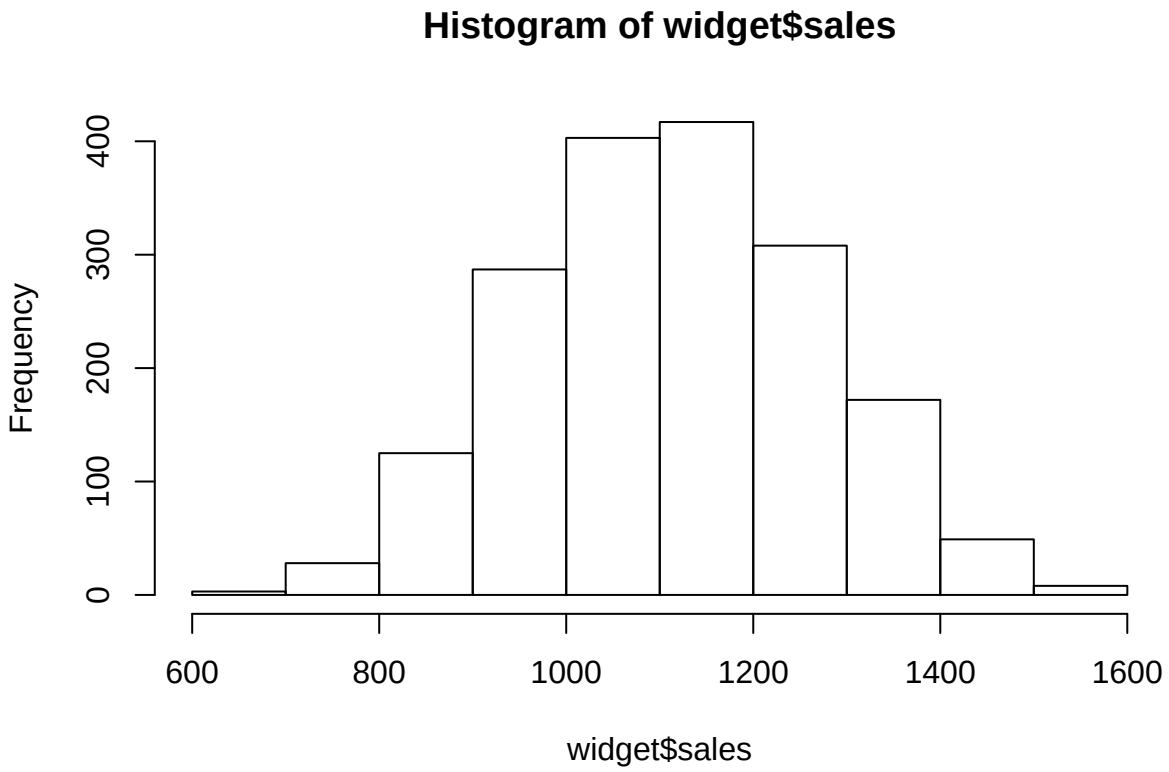
```
summary(widget)
```

```
##  year        month          day           sales           instance
##  1:360   1      :150   Min.   : 1.0   Min.   : 628.4   Min.   :   1.0
##  2:360   2      :150   1st Qu.: 8.0   1st Qu.:1002.0   1st Qu.: 450.8
##  3:360   3      :150   Median :15.5   Median :1115.3   Median : 900.5
##  4:360   4      :150   Mean   :15.5   Mean   :1113.8   Mean   : 900.5
##  5:360   5      :150   3rd Qu.:23.0   3rd Qu.:1226.1   3rd Qu.:1350.2
##          6      :150   Max.   :30.0   Max.   :1554.4   Max.   :1800.0
##          (Other):900
```

```
numericcol<-names(select_if(widget,is.numeric))
numericcol
```

```
## [1] "day"      "sales"    "instance"
```

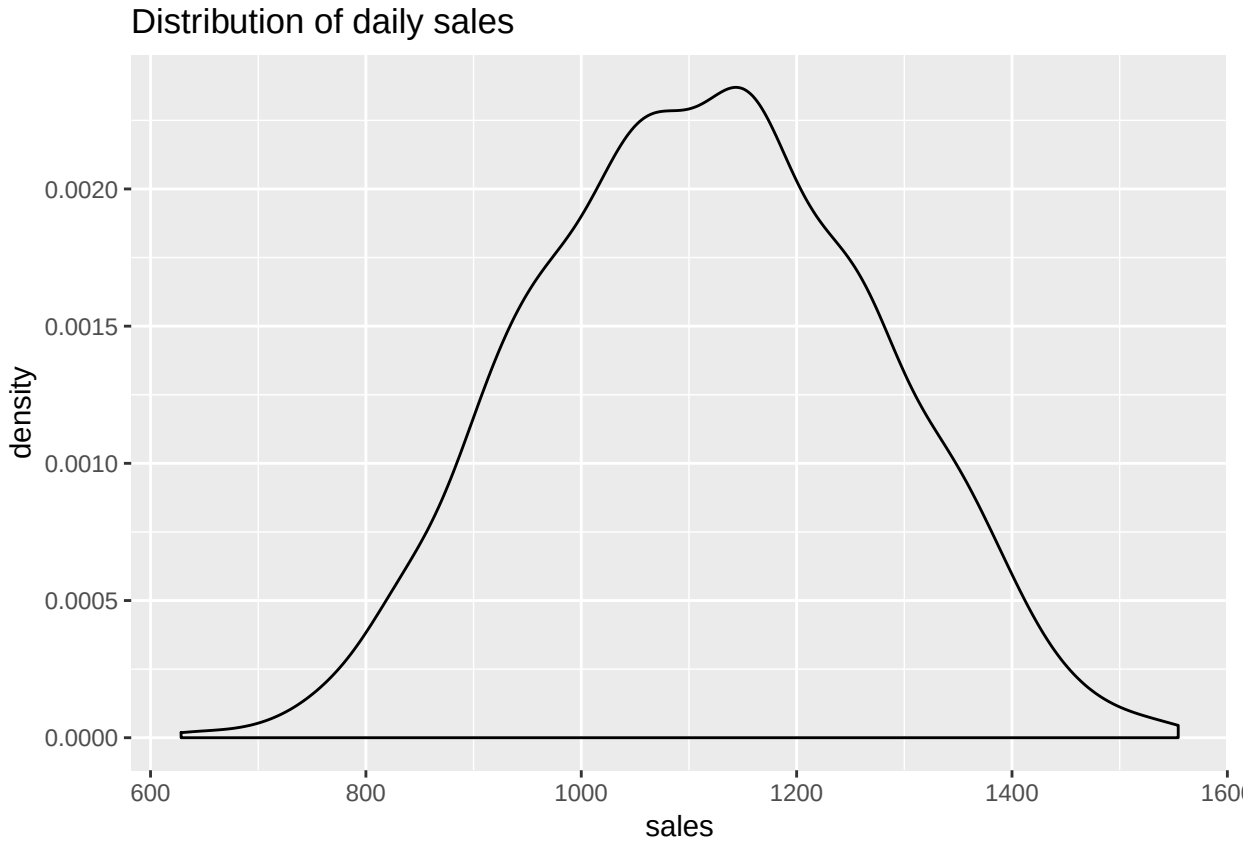
```
hist(widget$sales)
```



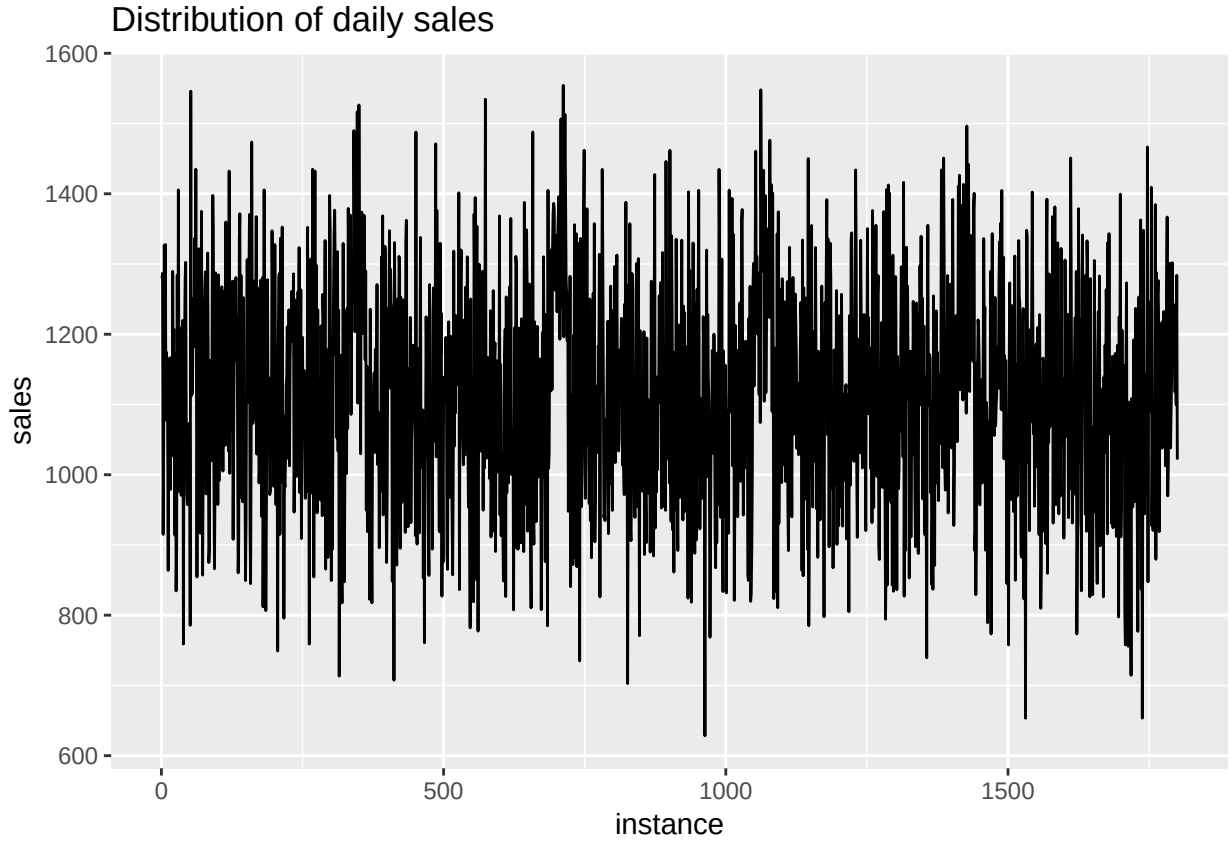
7. Try the easy solution first

    - Plot daily sales

```
ggplot(widget,aes(sales))+
  geom_density()+
  labs(title="Distribution of daily sales")
```




```
ggplot(widget,aes(instance,sales))+
  geom_line()+
  labs(title="Distribution of daily sales")
```



+ Plot monthly sales

```r
widget %>%
  group_by(month) %>%
  summarise(sales_over_month=sum(sales)) %>%
  ggplot(aes(month,sales_over_month,group=1))+
  geom_line() +
  labs(title="Total sales per month")
```



```r
#Alternatively
ggplot(widget,aes(month,sales,group=1))+
  stat_summary(fun.y = sum,geom="line")
```


Widget sales are **much** higher in December

+ Plot yearly sales

```r
```

```
ggplot(widget,aes(year,sales,group=1))+
  stat_summary(fun.y = sum,geom="line")
```

```
![](class3exercise_spring2019_files/figure-latex/unnamed-chunk-8-1.pdf)<!-- -->
```

The sales in last year dropped drastically as compared to previous months

```
+ Summarize total and average sales by month, and calculate confidence intervals.
```

```
widget %>%
  group_by(month) %>%
  summarize(total_sales=sum(sales),
            avg_sales=mean(sales),
            n=n(),
            SEM=sd(sales),
            lower=avg_sales-2*SEM,
            upper=avg_sales+2*SEM)
```

```
## # A tibble: 12 x 7
##    month total_sales avg_sales     n   SEM lower upper
##    <fct>       <dbl>     <dbl> <int> <dbl> <dbl> <dbl>
##  1 1         163818.     1092.   150  159.  773. 1411.
##  2 2         167397.     1116.   150  147.  823. 1409.
##  3 3         164400.     1096.   150  145.  807. 1385.
##  4 4         165643.     1104.   150  152.  800. 1409.
##  5 5         165205.     1101.   150  147.  808. 1395.
##  6 6         168521.     1123.   150  136.  851. 1396.
##  7 7         164629.     1098.   150  170.  757. 1438.
##  8 8         165385.     1103.   150  145.  813. 1392.
##  9 9         162495.     1083.   150  149.  786. 1380.
## 10 10        161517.     1077.   150  161.  755. 1398.
## 11 11        165093.     1101.   150  157.  786. 1415.
## 12 12        190720.     1271.   150  122. 1026. 1516.
```

```
+ Summarize total and average sales by  year, and calculate confidence intervals. For the CIs remember
```

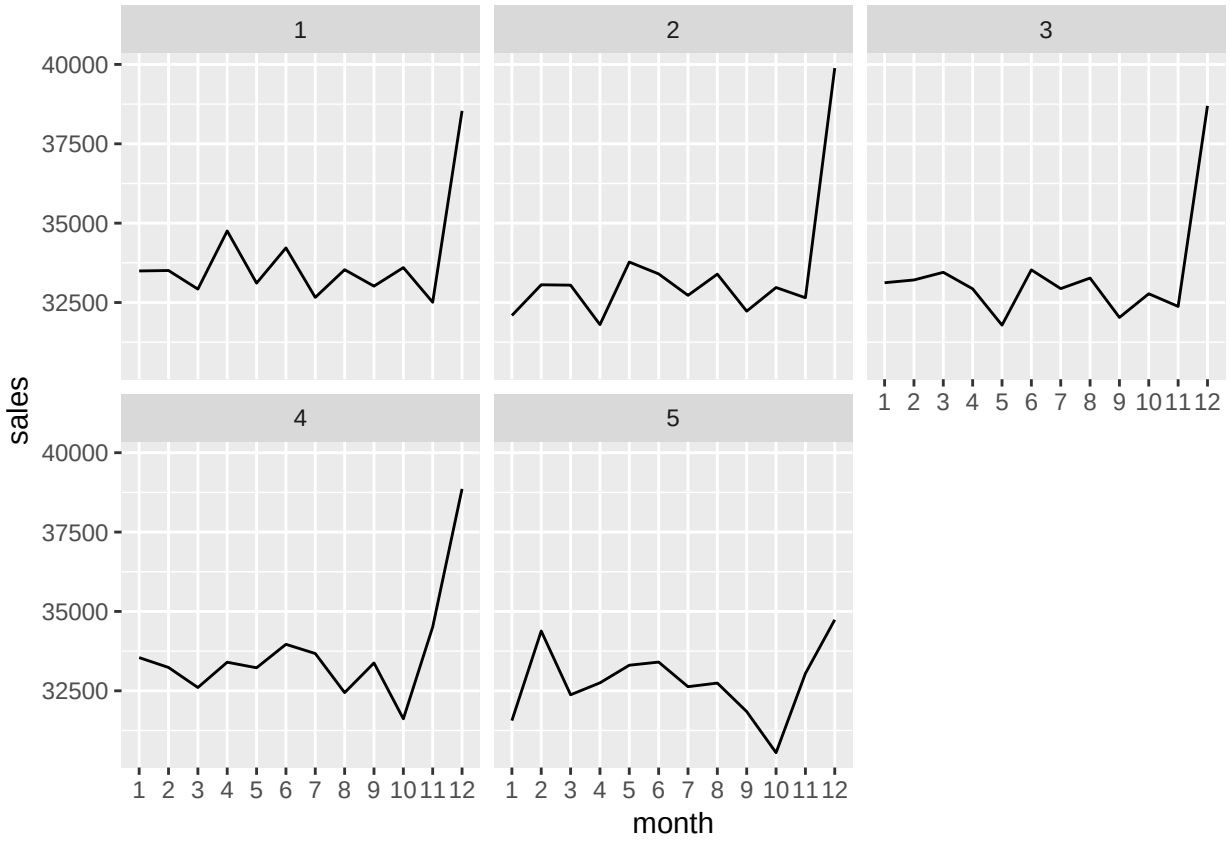
```
widget %>%
  group_by(year) %>%
  summarize(total_sales=sum(sales),
            avg_sales=mean(sales),
            n=n(),
            SEM=sd(sales)/sqrt(n),
            lower=avg_sales-1.96*SEM,
            upper=avg_sales+1.96*SEM)
```

```
## # A tibble: 5 x 7
##   year  total_sales avg_sales     n   SEM lower upper
##   <fct>       <dbl>     <dbl> <int> <dbl> <dbl> <dbl>
## 1 1         405872.     1127.   360  8.28 1111. 1144.
## 2 2         401035.     1114.   360  8.55 1097. 1131.
## 3 3         400118.     1111.   360  8.37 1095. 1128.
## 4 4         404461.     1124.   360  8.12 1108. 1139.
## 5 5         393338.     1093.   360  7.98 1077. 1108.
```

8. Challenge your solution

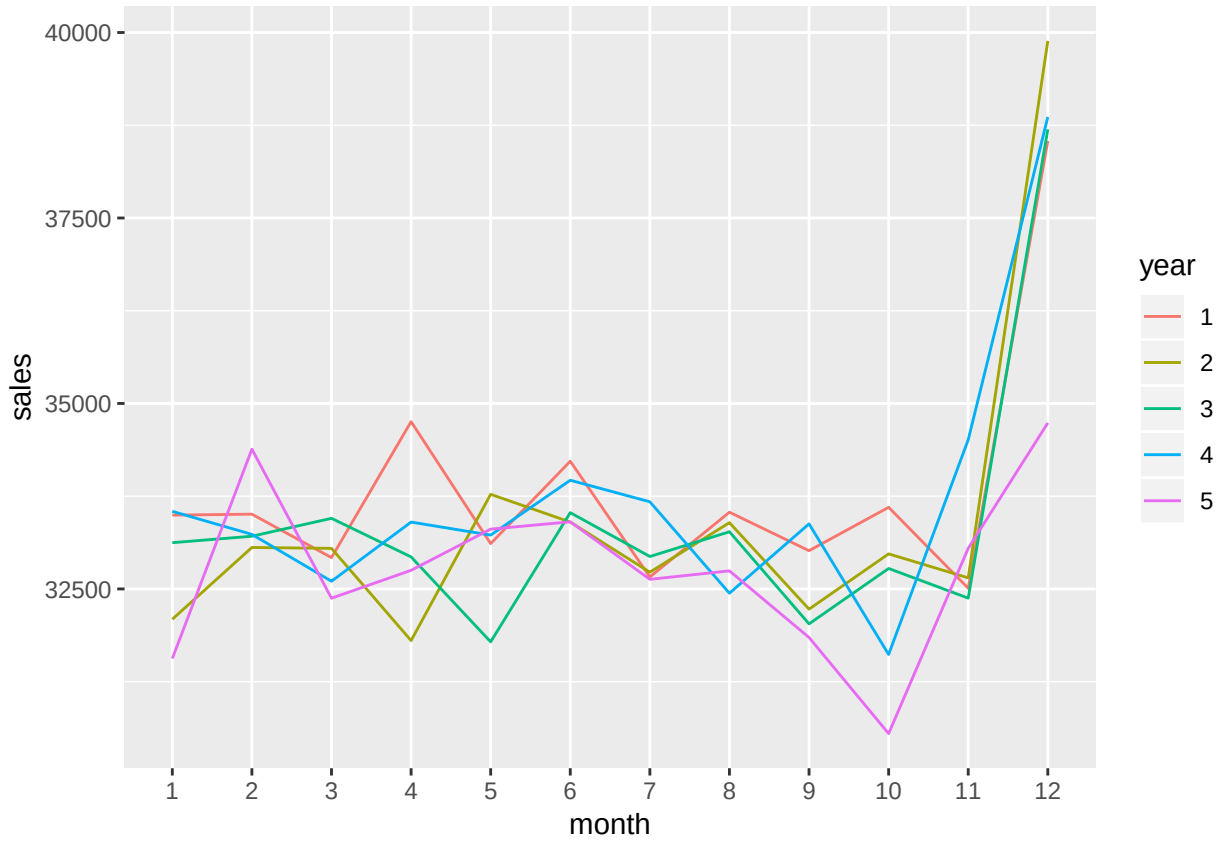
    - Plot monthy sales facetted by year

```
ggplot(widget,aes(month,sales,group=1))+
  stat_summary(fun.y=sum,geom="line")+
  facet_wrap(~year)
```

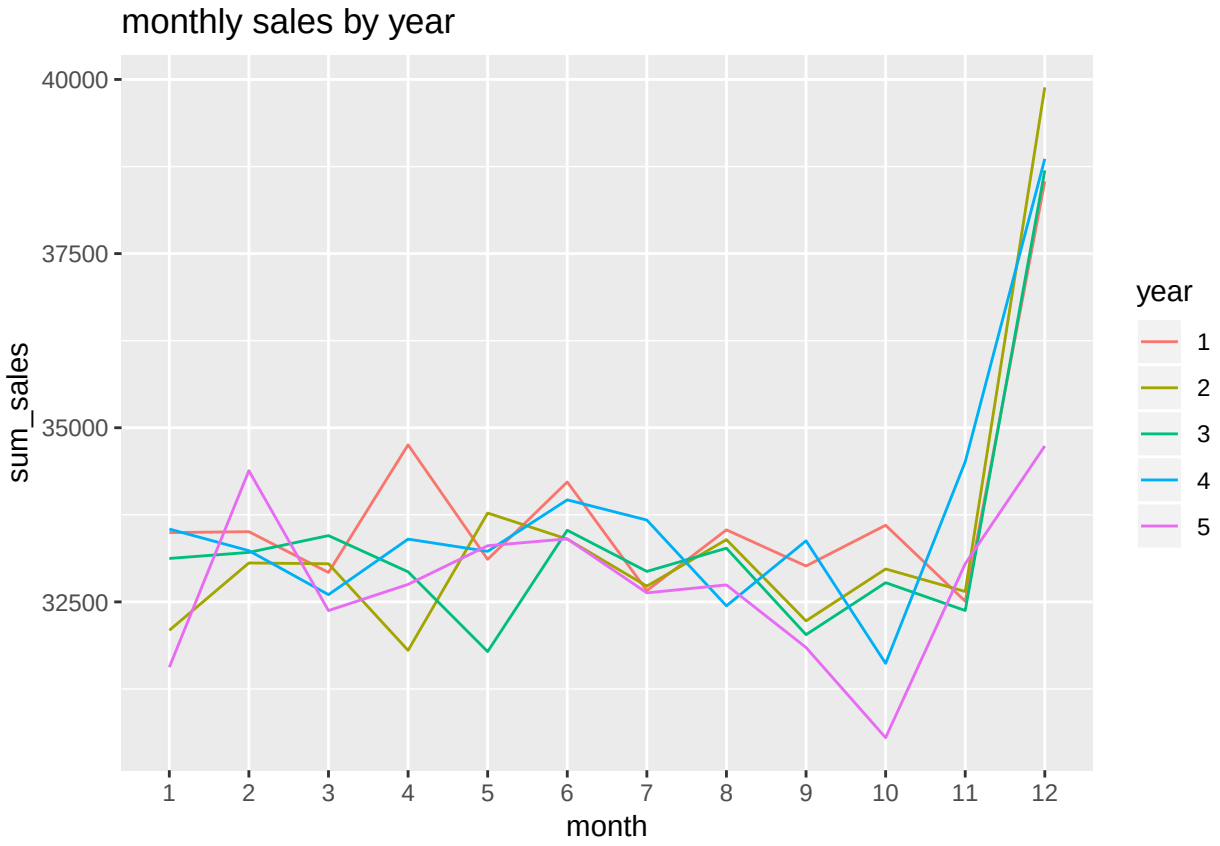


+ Plot monthly sales colored by year

```
ggplot(widget,aes(month,sales,group=year,color=year))+
  stat_summary(fun.y=sum,geom="line")
```




```
widget %>%
  group_by(month,year) %>%
  summarize(sum_sales=sum(sales))%>%
  ggplot(aes(month,sum_sales,group=year,color=year))+
  geom_line()+
  labs(title="monthly sales by year")
```



+ Summarize total and average sales by month and year, and calculate confidence intervals.

```
widget %>%
  group_by(month,year) %>%
  summarize(total_sales=sum(sales),
            avg_sales=mean(sales),
            n=n(),
            SEM=sd(sales)/sqrt(n),
            lower=avg_sales-2*SEM,
            upper=avg_sales+2*SEM)  %>%
            filter(month==12)
```

```
## # A tibble: 5 x 8
## # Groups:   month [1]
##   month year  total_sales avg_sales     n   SEM lower upper
##   <fct> <fct>       <dbl>     <dbl> <int> <dbl> <dbl> <dbl>
## 1 12    1           38538.     1285.    30  21.2 1242. 1327.
## 2 12    2           39887.     1330.    30  18.7 1292. 1367.
## 3 12    3           38695.     1290.    30  20.3 1249. 1330.
## 4 12    4           38861.     1295.    30  20.8 1254. 1337.
## 5 12    5           34738.     1158.    30  18.1 1122. 1194.
```

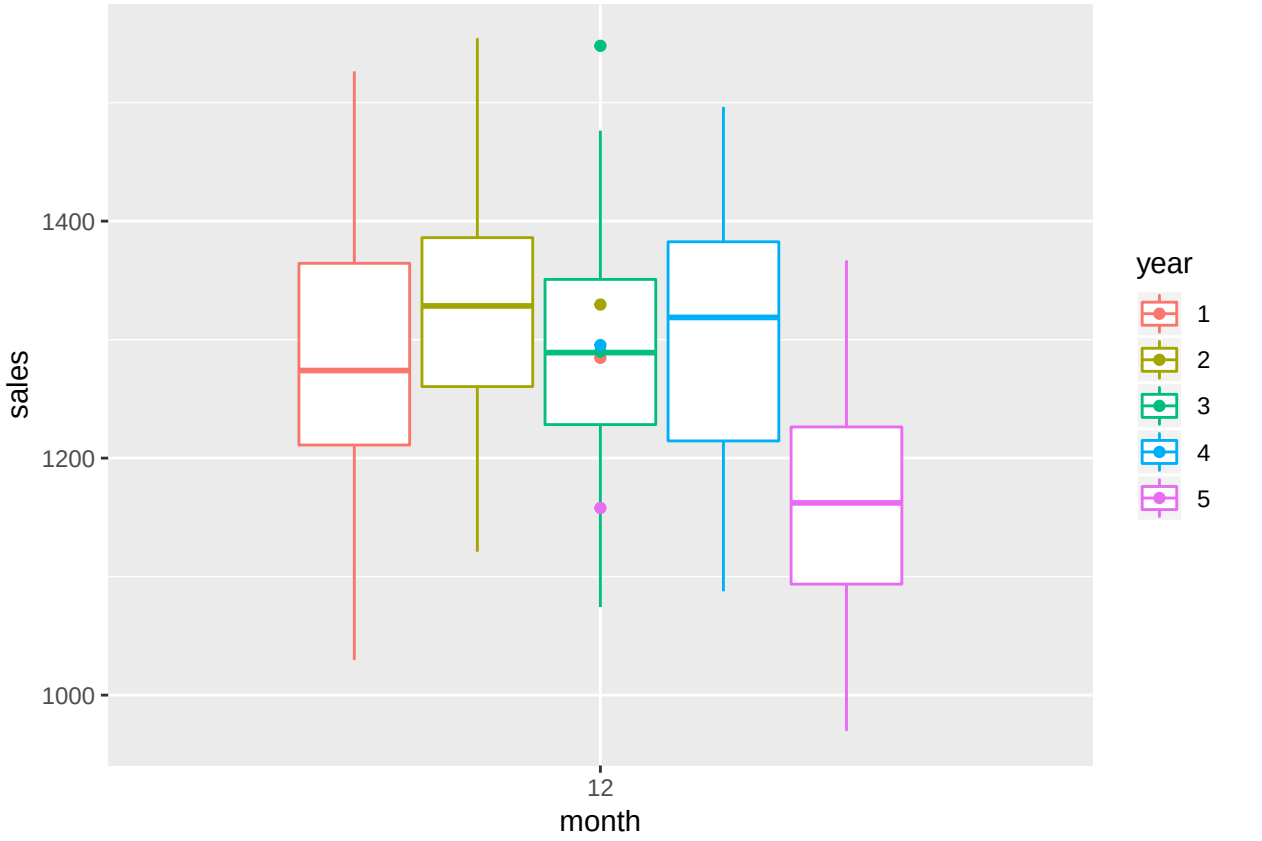
9. Follow up questions. See next section.

Statistical inference

Answer the CEO's question: Is there a real drop in year 5 month 12 sales or is the difference just due to random variation?

How would you approach this question?

```
ggplot(subset(widget,month==12),aes(month,sales,color=year))+ geom_boxplot()+
  stat_summary(fun.y=mean,geom="point")
```



Communication

Write a paragraph summary of your descriptive and inferential findings.

On comparing the yearly sales for month of December, we can see that the sales have much dropped as cmpared to previous years. Plotting the box plot we can see the difference between the groups and can see their mean and lower and upper boundaries are not the same. hence we can reject the null hypothesis. Now we will look at the statistical importance of the month and year for month 12 and year 2015. We see while month plays an significant impact on the sales, the year has no significant impact.

```
model<-lm(sales~month*year,widget)
summary(model)
```

```
##
## Call:
## lm(formula = sales ~ month * year, data = widget)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -439.27 -106.84   -1.01   97.63  429.13
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)   1116.4982    27.2286  41.005  < 2e-16 ***
## month2           0.4324    38.5071   0.011   0.9910
## month3         -19.0911    38.5071  -0.496   0.6201
## month4          41.9929    38.5071   1.091   0.2756
## month5         -12.7976    38.5071  -0.332   0.7397
## month6          24.2214    38.5071   0.629   0.5294
## month7         -27.7801    38.5071  -0.721   0.4707
## month8           1.3174    38.5071   0.034   0.9727
## month9         -15.9869    38.5071  -0.415   0.6781
## month10          3.5122    38.5071   0.091   0.9273
## month11        -32.8453    38.5071  -0.853   0.3938
## month12        168.1170    38.5071   4.366 1.34e-05 ***
## year2          -46.7688    38.5071  -1.215   0.2247
## year3          -12.3772    38.5071  -0.321   0.7479
## year4            1.6993    38.5071   0.044   0.9648
## year5          -64.4303    38.5071  -1.673   0.0945 .
## month2:year2    31.8004    54.4573   0.584   0.5593
## month3:year2    50.9343    54.4573   0.935   0.3498
## month4:year2   -51.5909    54.4573  -0.947   0.3436
## month5:year2    68.9144    54.4573   1.265   0.2059
## month6:year2    19.4344    54.4573   0.357   0.7212
## month7:year2    48.9099    54.4573   0.898   0.3692
## month8:year2    42.0789    54.4573   0.773   0.4398
## month9:year2    20.5047    54.4573   0.377   0.7066
## month10:year2   25.8499    54.4573   0.475   0.6351
## month11:year2   51.4282    54.4573   0.944   0.3451
## month12:year2   91.7188    54.4573   1.684   0.0923 .
## month2:year3     2.4619    54.4573   0.045   0.9639
## month3:year3    30.0279    54.4573   0.551   0.5814
## month4:year3   -48.3845    54.4573  -0.888   0.3744
## month5:year3   -31.7680    54.4573  -0.583   0.5597
## month6:year3   -10.7348    54.4573  -0.197   0.8438
## month7:year3    21.5689    54.4573   0.396   0.6921
## month8:year3     3.5797    54.4573   0.066   0.9476
## month9:year3   -20.4908    54.4573  -0.376   0.7068
## month10:year3  -15.0970    54.4573  -0.277   0.7816
## month11:year3    7.9313    54.4573   0.146   0.8842
## month12:year3   17.6107    54.4573   0.323   0.7464
## month2:year4   -10.7798    54.4573  -0.198   0.8431
## month3:year4   -12.3103    54.4573  -0.226   0.8212
## month4:year4   -46.8178    54.4573  -0.860   0.3901
## month5:year4     2.1207    54.4573   0.039   0.9689
## month6:year4   -10.2536    54.4573  -0.188   0.8507
## month7:year4    32.0758    54.4573   0.589   0.5559
## month8:year4   -38.0917    54.4573  -0.699   0.4843
## month9:year4    10.3555    54.4573   0.190   0.8492
## month10:year4  -67.8060    54.4573  -1.245   0.2133
## month11:year4   65.0113    54.4573   1.194   0.2327
## month12:year4    9.0663    54.4573   0.166   0.8678
## month2:year5    93.6257    54.4573   1.719   0.0857 .
## month3:year5    46.1978    54.4573   0.848   0.3964
## month4:year5    -2.3591    54.4573  -0.043   0.9655
## month5:year5    70.9398    54.4573   1.303   0.1929
## month6:year5    37.2102    54.4573   0.683   0.4945
```

```
## month7:year5    63.3688    54.4573   1.164   0.2447
## month8:year5    38.0786    54.4573   0.699   0.4845
## month9:year5    25.4463    54.4573   0.467   0.6404
## month10:year5  -37.2338    54.4573  -0.684   0.4942
## month11:year5   82.3414    54.4573   1.512   0.1307
## month12:year5  -62.2535    54.4573  -1.143   0.2531
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 149.1 on 1740 degrees of freedom
## Multiple R-squared:  0.1278, Adjusted R-squared:  0.09827
## F-statistic: 4.323 on 59 and 1740 DF,  p-value: < 2.2e-16
```