assignment_4_MukherjeeChitramoy.R

chitro

2023-01-09

```
# Assignment: ASSIGNMENT 4
# Name: Mukherjee, Chitramoy
# Date: 2023-01-04

#Assignment 4.1
#What are the observational units in this study
#Count and Score columns are observational units in this study.

## Set the working directory to the root of your DSC 520 directory
setwd("C:/Users/chitro/Desktop/dsc520-fork-chitro")

## Load the `data/r4ds/heights.csv` to
scores <- read.csv("data/scores.csv",header=TRUE,sep = ",")

#Identify the variables mentioned in the narrative paragraph and
determine which are categorical and quantitative?
names(scores)

## [1] "Count"   "Score"   "Section"

summary(scores)

##      Count           Score         Section
##  Min.   :10.00   Min.   :200.0   Length:38
##  1st Qu.:10.00   1st Qu.:300.0   Class :character
##  Median :10.00   Median :322.5   Mode  :character
##  Mean   :14.47   Mean   :317.5
##  3rd Qu.:20.00   3rd Qu.:357.5
##  Max.   :30.00   Max.   :395.0

str(scores)

## 'data.frame':    38 obs. of  3 variables:
##  $ Count  : int  10 10 20 10 10 10 10 30 10 10 ...
##  $ Score  : int  200 205 235 240 250 265 275 285 295 300 ...
##  $ Section: chr  "Sports" "Sports" "Sports" "Sports" ...

class(scores$Count)

## [1] "integer"
```

```
class(scores$Score)
```

```
## [1] "integer"
```

```
class(scores$Section)
```

```
## [1] "character"
```

```
#categorical - Section
#quantitative - Count, Score
```

```
Regular_var <- (scores[scores$Section =='Regular', ])
Regular_var
```

```
##    Count Score Section
## 6     10   265 Regular
## 7     10   275 Regular
## 9     10   295 Regular
## 10    10   300 Regular
## 13    10   305 Regular
## 14    10   310 Regular
## 16    20   320 Regular
## 17    10   305 Regular
## 19    20   320 Regular
## 20    10   325 Regular
## 22    20   330 Regular
## 25    10   335 Regular
## 26    20   340 Regular
## 28    30   350 Regular
## 29    20   360 Regular
## 31    20   365 Regular
## 34    10   370 Regular
## 35    20   375 Regular
## 37    20   380 Regular
```

```
sports_var <- (scores[scores$Section =='Sports', ])
sports_var
```

```
##    Count Score Section
## 1     10   200  Sports
## 2     10   205  Sports
## 3     20   235  Sports
## 4     10   240  Sports
## 5     10   250  Sports
## 8     30   285  Sports
## 11    20   300  Sports
## 12    10   305  Sports
## 15    10   310  Sports
```

```
## 18    10   315  Sports
## 21    10   325  Sports
## 23    10   330  Sports
## 24    30   335  Sports
## 27    10   340  Sports
## 30    10   360  Sports
## 32    20   365  Sports
## 33    10   370  Sports
## 36    10   375  Sports
## 38    10   395  Sports
```

```
#Use the Plot function to plot each Sections scores and the number of students achieving that score.
#Use additional Plot Arguments to label the graph and give each axis an appropriate label
library(ggplot2)
ggplot(scores, aes(x=Score, y=Count, group=Section, color=Section)) +
geom_line() + geom_point() +
labs(x="Score", y= "count", title = "Score vs. count ploting based on Section data")
```


```
#Comparing and contrasting the point distributions between the two
section, looking at both tendency and consistency: Can you say that
one section tended to score more points than the other?
#Can't say one section tended to score more than the other as per the
ploting.
#Did every student in one section score more points than every student
in the other section?
#As per the plotting, yes every student of one section scored more
points than every student of other section.

#What could be one additional variable that was not mentioned in the
narrative that could be influencing the point distributions between
the two sections?
#Male and Female student count.

# Assignment 4.2
#Use the apply function on a variable in your dataset
library(readxl)
housing_df <- read_excel("data/week-7-housing.xlsx")
```

```
head (housing_df)
```

```
## # A tibble: 6 × 24
##   `Sale Date`         `Sale Price` sale_…¹ sale_…² sale_…³ sitet…⁴
addr_…⁵  zip5
##   <dttm>                     <dbl>   <dbl>   <dbl> <chr>   <chr>
<chr>   <dbl>
## 1 2006-01-03 00:00:00       698000       1       3 <NA>    R1
17021 … 98052
## 2 2006-01-03 00:00:00       649990       1       3 <NA>    R1
11927 … 98052
## 3 2006-01-03 00:00:00       572500       1       3 <NA>    R1
13315 … 98052
## 4 2006-01-03 00:00:00       420000       1       3 <NA>    R1
3303 1… 98052
## 5 2006-01-03 00:00:00       369900       1       3 15      R1
16126 … 98052
## 6 2006-01-03 00:00:00       184667       1      15 18 51   R1
8101 2… 98053
## # … with 16 more variables: ctyname <chr>, postalctyn <chr>, lon
<dbl>,
## #   lat <dbl>, building_grade <dbl>, square_feet_total_living
<dbl>,
## #   bedrooms <dbl>, bath_full_count <dbl>, bath_half_count <dbl>,
## #   bath_3qtr_count <dbl>, year_built <dbl>, year_renovated <dbl>,
## #   current_zoning <chr>, sq_ft_lot <dbl>, prop_type <chr>,
present_use <dbl>,
## #   and abbreviated variable names ¹sale_reason, ²sale_instrument,
## #   ³sale_warning, ⁴sitetype, ⁵addr_full
```

```
colnames(housing_df)
```

```
##  [1] "Sale Date"                "Sale Price"
##  [3] "sale_reason"              "sale_instrument"
##  [5] "sale_warning"             "sitetype"
##  [7] "addr_full"                "zip5"
##  [9] "ctyname"                  "postalctyn"
## [11] "lon"                      "lat"
## [13] "building_grade"           "square_feet_total_living"
## [15] "bedrooms"                 "bath_full_count"
## [17] "bath_half_count"          "bath_3qtr_count"
## [19] "year_built"               "year_renovated"
## [21] "current_zoning"           "sq_ft_lot"
## [23] "prop_type"                "present_use"
```

```
class(housing_df)
```

```
## [1] "tbl_df"     "tbl"        "data.frame"
```

```r
#Use the apply function on a variable in your dataset
sale_price_df <- data.frame(housing_df$`Sale Price`)
#sale_price_df
apply(sale_price_df,2 , mean)
```

```
## housing_df..Sale.Price.
##                660737.7
```

```r
##install.packages("dplyr")
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```r
library(ggplot2)

#Use the aggregate function on a variable in your dataset
aggregate (`Sale Price` ~ bedrooms, housing_df, mean )
```

```
##    bedrooms Sale Price
## 1         0   844059.5
## 2         1   722814.1
## 3         2   544946.4
## 4         3   564958.6
## 5         4   735910.0
## 6         5   836974.0
## 7         6   767494.3
## 8         7  1307281.7
## 9         8  1122500.0
## 10        9   581500.0
## 11       10   450000.0
## 12       11  1825000.0
```

```r
#Use the plyr function on a variable in your dataset - more
specifically, I want to see you split some data, perform a
modification to the data, and then bring it back together
hdf_2006 <- housing_df[housing_df$`Sale Price` >= 500008,1:3]

dates <- as.POSIXct(housing_df$`Sale Date`, format = "%m/%d/%Y")
years <- format(dates, format="%Y")
str(dates)
```

```
##  POSIXct[1:12865], format: "2006-01-03" "2006-01-03" "2006-01-03"
"2006-01-03" "2006-01-03" ...
```

```
summary(dates)
```

```
##                         Min.                         1st Qu.
## "2006-01-03 00:00:00.0000" "2008-07-07 00:00:00.0000"
##                       Median                           Mean
## "2011-11-17 00:00:00.0000" "2011-07-28 15:07:32.4834"
##                      3rd Qu.                          Max.
## "2014-06-05 00:00:00.0000" "2016-12-16 00:00:00.0000"
```

```
class(dates)
```

```
## [1] "POSIXct" "POSIXt"
```

```
str(housing_df)
```

```
## tibble [12,865 × 24] (S3: tbl_df/tbl/data.frame)
##  $ Sale Date               : POSIXct[1:12865], format: "2006-01-03"
"2006-01-03" ...
##  $ Sale Price              : num [1:12865] 698000 649990 572500
420000 369900 ...
##  $ sale_reason             : num [1:12865] 1 1 1 1 1 1 1 1 1 ...
##  $ sale_instrument         : num [1:12865] 3 3 3 3 3 15 3 3 3 3 ...
##  $ sale_warning            : chr [1:12865] NA NA NA NA ...
##  $ sitetype                : chr [1:12865] "R1" "R1" "R1" "R1" ...
##  $ addr_full               : chr [1:12865] "17021 NE 113TH CT"
"11927 178TH PL NE" "13315 174TH AVE NE" "3303 178TH AVE NE" ...
##  $ zip5                    : num [1:12865] 98052 98052 98052 98052
98052 ...
##  $ ctyname                 : chr [1:12865] "REDMOND" "REDMOND" NA
"REDMOND" ...
##  $ postalctyn              : chr [1:12865] "REDMOND" "REDMOND"
"REDMOND" "REDMOND" ...
##  $ lon                     : num
[1:12865] -122 -122 -122 -122 -122 ...
##  $ lat                     : num [1:12865] 47.7 47.7 47.7 47.6
47.7 ...
##  $ building_grade          : num [1:12865] 9 9 8 8 7 7 10 10 9
8 ...
##  $ square_feet_total_living: num [1:12865] 2810 2880 2770 1620 1440
4160 3960 3720 4160 2760 ...
##  $ bedrooms                : num [1:12865] 4 4 4 3 3 4 5 4 4 4 ...
##  $ bath_full_count         : num [1:12865] 2 2 1 1 1 2 3 2 2 1 ...
##  $ bath_half_count         : num [1:12865] 1 0 1 0 0 1 0 1 1 0 ...
##  $ bath_3qtr_count         : num [1:12865] 0 1 1 1 1 1 1 0 1 1 ...
##  $ year_built              : num [1:12865] 2003 2006 1987 1968
1980 ...
```

```
##  $ year_renovated          : num [1:12865] 0 0 0 0 0 0 0 0 0 0 ...
##  $ current_zoning          : chr [1:12865] "R4" "R4" "R6" "R4" ...
##  $ sq_ft_lot               : num [1:12865] 6635 5570 8444 9600
7526 ...
##  $ prop_type               : chr [1:12865] "R" "R" "R" "R" ...
##  $ present_use             : num [1:12865] 2 2 2 2 2 2 2 2 2 ...
```

```
head(years)
```

```
## [1] "2006" "2006" "2006" "2006" "2006" "2006"
```

```
dates <- NULL
years <- NULL
```

```
housing_df %>% select(`Sale Price`,`Sale Date`) %>% filter(`Sale
Price` > 4000000)
```

```
## # A tibble: 32 × 2
##    `Sale Price` `Sale Date`
##           <dbl> <dttm>
##  1      4400000 2010-03-02 00:00:00
##  2      4400000 2010-03-02 00:00:00
##  3      4380542 2011-11-17 00:00:00
##  4      4380542 2011-11-17 00:00:00
##  5      4380542 2011-11-17 00:00:00
##  6      4380542 2011-11-17 00:00:00
##  7      4380542 2011-11-17 00:00:00
##  8      4380542 2011-11-17 00:00:00
##  9      4380542 2011-11-17 00:00:00
## 10      4380542 2011-11-17 00:00:00
## # … with 22 more rows
```

```
#housing_df <- read.csv("data/week-7-housing.csv")
#str(housing_df)
```

```
#housing_df <- read.csv("data/week-7-housing.csv")
```

```
housing_df %>%
  filter(`Sale Price` > 1000000) %>%
  group_by(`Sale Date`) %>%
  summarize(average_revenue=mean(`Sale Price`),sdev_revenue=sd(`Sale
Price`))
```

```
## # A tibble: 617 × 3
##    `Sale Date`         average_revenue sdev_revenue
##    <dttm>                        <dbl>        <dbl>
##  1 2006-01-04 00:00:00        1050000           NA
##  2 2006-01-12 00:00:00        1392000           NA
##  3 2006-01-23 00:00:00        1445000           NA
##  4 2006-01-26 00:00:00        1053649           NA
```

```
##  5 2006-02-01 00:00:00        1490068.      579732.
##  6 2006-02-03 00:00:00        1075000           NA
##  7 2006-02-13 00:00:00        1520000           NA
##  8 2006-02-15 00:00:00        1390000            0
##  9 2006-03-07 00:00:00        1100000           NA
## 10 2006-03-09 00:00:00        1380000           NA
## # … with 607 more rows
```

```
#housing_df <- read.csv("data/week-7-housing.csv")

#housing_df %>%
#  mutate(year=format(as.Date(Sale.Date, format="%m/%d/%Y"),"%Y"))

#head(housing_df)

housing_df %>%
  filter(`Sale Price` > 1000000) %>%
  mutate(year=format(as.Date(`Sale Date`, format="%m/%d/%Y"),"%Y"))
%>%
  group_by(year) %>%
  summarize(average_sale_price=mean(`Sale
Price`),sdev_sale_price=sd(`Sale Price`)) %>%
  arrange(desc(year))
```

```
## # A tibble: 11 × 3
##    year  average_sale_price sdev_sale_price
##    <chr>              <dbl>           <dbl>
##  1 2016            1376730.         530913.
##  2 2015            1244531.         259357.
##  3 2014            1254733.         263183.
##  4 2013            1353820.         443343.
##  5 2012            2236266.         905252.
##  6 2011            2841839.        1361869.
##  7 2010            1462957.         645088.
##  8 2009            1369636.         292656.
##  9 2008            2574278.         866200.
## 10 2007            1624951.         588392.
## 11 2006            1383097.         369128.
```

```
#housing_df %>%
#  mutate(Sale.month=format(as.Date(Sale.Date, format="%m/%d/%Y"),"%
m"))

housing_df %>%
  filter(`Sale Price` > 1000000) %>%
  mutate(monthyear = format(as.Date(`Sale Date`, format="%m/%d/%Y"),"%
m/%Y")) %>%
  group_by(monthyear) %>%
  summarize(average_sale_price=mean(`Sale
```

```
Price`),sdev_sale_price=sd(`Sale Price`)) %>%
  arrange(monthyear)
```

```
## # A tibble: 124 × 3
##    monthyear average_sale_price sdev_sale_price
##    <chr>                  <dbl>           <dbl>
##  1 01/2006             1235162.         212808.
##  2 01/2007             1387500          300520.
##  3 01/2008             1050000              NA
##  4 01/2009             1400000              NA
##  5 01/2011             1483333.         335012.
##  6 01/2012             1462500          441942.
##  7 01/2014             1240000           75498.
##  8 01/2015             1045990            5657.
##  9 01/2016             1415398          351044.
## 10 02/2006             1392522.         307312.
## # … with 114 more rows
```

```
# housing_df$year <- NULL

# split sales date column, derive month on it and attach back to the
frame.

housing_df$year <- format(as.Date(housing_df$`Sale Date`, format="%m/%
d/%Y"),"%Y")
tail(housing_df)
```

```
## # A tibble: 6 × 25
##   `Sale Date`         `Sale Price` sale_…¹ sale_…² sale_…³ sitet…⁴
addr_…⁵  zip5
##   <dttm>                     <dbl>   <dbl>   <dbl> <chr>   <chr>
<chr>   <dbl>
## 1 2016-12-15 00:00:00       824000       1       3 <NA>    R1
11314 … 98052
## 2 2016-12-15 00:00:00       798930       1       3 <NA>    R1
22506 … 98053
## 3 2016-12-15 00:00:00       750000       1       3 <NA>    R1
13315 … 98052
## 4 2016-12-15 00:00:00       629000       1       3 <NA>    R1
17716 … 98052
## 5 2016-12-16 00:00:00       835000       1       3 <NA>    R1
9917 1… 98052
## 6 2016-12-16 00:00:00       455500       1       3 <NA>    R1
8826 1… 98052
## # … with 17 more variables: ctyname <chr>, postalctyn <chr>, lon
<dbl>,
##   lat <dbl>, building_grade <dbl>, square_feet_total_living
<dbl>,
##   bedrooms <dbl>, bath_full_count <dbl>, bath_half_count <dbl>,
```

```
## #   bath_3qtr_count <dbl>, year_built <dbl>, year_renovated <dbl>,
## #   current_zoning <chr>, sq_ft_lot <dbl>, prop_type <chr>,
present_use <dbl>,
## #   year <chr>, and abbreviated variable names ¹sale_reason, ²
sale_instrument,
## #   ³sale_warning, ⁴sitetype, ⁵addr_full

# More examples.
housing_df$year <- format(mean(housing_df$`Sale Price`))
housing_df$Zip <- housing_df$zip5 == "98052"
head(housing_df$Zip)

## [1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE

housing_df$million_above <- housing_df$`Sale Price` >= 1000000
head(housing_df$million_above)

## [1] FALSE FALSE FALSE FALSE FALSE FALSE

# split sales date column, derive month on it and attach back to the
frame.

housing_df$year <- format(as.Date(housing_df$`Sale Date`, format="%m/%
d/%Y"),"%Y")
tail(housing_df)

## # A tibble: 6 × 27
##   `Sale Date`         `Sale Price` sale_…¹ sale_…² sale_…³ sitet…⁴
addr_…⁵  zip5
##   <dttm>                     <dbl>   <dbl>   <dbl> <chr>   <chr>
<chr>   <dbl>
## 1 2016-12-15 00:00:00       824000       1       3 <NA>    R1
11314 … 98052
## 2 2016-12-15 00:00:00       798930       1       3 <NA>    R1
22506 … 98053
## 3 2016-12-15 00:00:00       750000       1       3 <NA>    R1
13315 … 98052
## 4 2016-12-15 00:00:00       629000       1       3 <NA>    R1
17716 … 98052
## 5 2016-12-16 00:00:00       835000       1       3 <NA>    R1
9917 1… 98052
## 6 2016-12-16 00:00:00       455500       1       3 <NA>    R1
8826 1… 98052
## # … with 19 more variables: ctyname <chr>, postalctyn <chr>, lon
<dbl>,
## #   lat <dbl>, building_grade <dbl>, square_feet_total_living
<dbl>,
## #   bedrooms <dbl>, bath_full_count <dbl>, bath_half_count <dbl>,
## #   bath_3qtr_count <dbl>, year_built <dbl>, year_renovated <dbl>,
```

```
## #   current_zoning <chr>, sq_ft_lot <dbl>, prop_type <chr>,
present_use <dbl>,
## #   year <chr>, Zip <lgl>, million_above <lgl>, and abbreviated
variable names
## #   ¹sale_reason, ²sale_instrument, ³sale_warning, ⁴sitetype, ⁵
addr_full
```

```
# More examples.
housing_df$year <- format(mean(housing_df$`Sale Price`))
housing_df$Zip <- housing_df$zip5 == "98052"
head(housing_df$Zip)
```

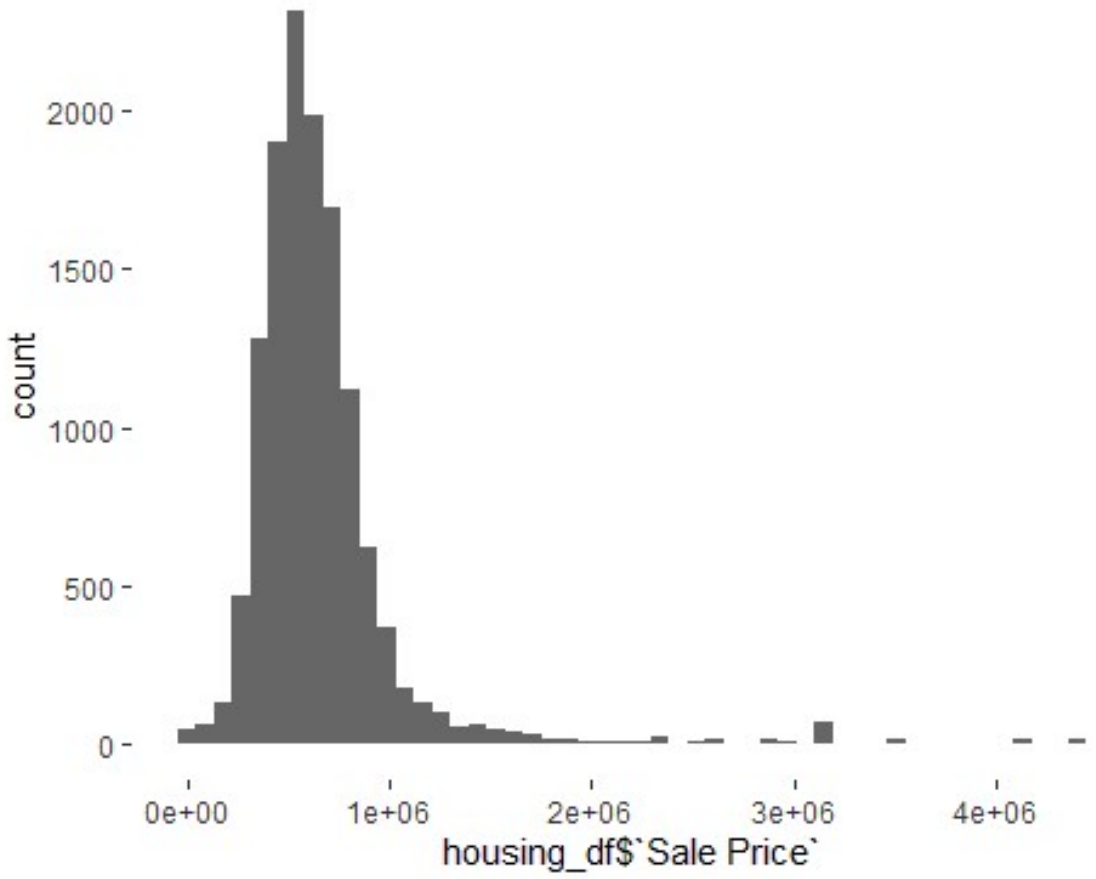
```
## [1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE
```

```
housing_df$million_above <- housing_df$`Sale Price` >= 1000000
head(housing_df$million_above)
```

```
## [1] FALSE FALSE FALSE FALSE FALSE FALSE
```

```
ggplot(housing_df, aes(housing_df$`Sale Price`)) + geom_histogram(bins
= 50)
```

```
## Warning: Use of `` housing_df$`Sale Price` `` is discouraged.
## ℹ Use `Sale Price` instead.
```




```
Mean_df <- housing_df %>%
  filter(`Sale Price`> 1000000) %>%
  mutate(year = format(as.Date(`Sale Date`, format="%m/%d/%Y"),"%Y"))
%>%
  group_by(year) %>%
  summarize(average_sale_price=mean(`Sale
Price`),sdev_sale_price=sd(`Sale Price`)) %>%
  arrange(year)

head(Mean_df)
```

```
## # A tibble: 6 × 3
##   year  average_sale_price sdev_sale_price
##   <chr>              <dbl>           <dbl>
## 1 2006            1383097.         369128.
## 2 2007            1624951.         588392.
## 3 2008            2574278.         866200.
## 4 2009            1369636.         292656.
## 5 2010            1462957.         645088.
```

```
## 6 2011            2841839.        1361869.
```

```
#str(Mean_df)
#Length(Mean_df$monthyear)
```

```
ggplot(Mean_df, aes(x = Mean_df$year, y = Mean_df$average_sale_price,
label=Mean_df$year)) +
  geom_point(size = 2.1, color="Blue") +
  geom_line() +
  ggtitle("Mean Sales Transaction Per Year") +
  xlab("Year") +
  ylab("Sales Mean Prices") +
  ##  geom_text() +
  geom_errorbar(aes(ymin=Mean_df$average_sale_price - Mean_df
$sdev_sale_price),
                ymax=Mean_df$average_sale_price + Mean_df
$sdev_sale_price,
                width=0.5 )
```

```
## Warning: Use of `Mean_df$year` is discouraged.
## ℹ Use `year` instead.
```

```
## Warning: Use of `Mean_df$average_sale_price` is discouraged.
## ℹ Use `average_sale_price` instead.
```

```
## Warning: Use of `Mean_df$year` is discouraged.
## ℹ Use `year` instead.
## Use of `Mean_df$year` is discouraged.
## ℹ Use `year` instead.
```

```
## Warning: Use of `Mean_df$average_sale_price` is discouraged.
## ℹ Use `average_sale_price` instead.
```

```
## Warning: Use of `Mean_df$year` is discouraged.
## ℹ Use `year` instead.
```

```
## Warning: Use of `Mean_df$average_sale_price` is discouraged.
## ℹ Use `average_sale_price` instead.
```

```
## Warning: Use of `Mean_df$sdev_sale_price` is discouraged.
## ℹ Use `sdev_sale_price` instead.
```

```
## Warning: Use of `Mean_df$year` is discouraged.
## ℹ Use `year` instead.
```

```
## Warning: Use of `Mean_df$average_sale_price` is discouraged.
## ℹ Use `average_sale_price` instead.
```

```
## Warning: Use of `Mean_df$year` is discouraged.
```

```
## ℹ Use `year` instead.
```

```
## `geom_line()`: Each group consists of only one observation.
## ℹ Do you need to adjust the group aesthetic?
```