EDA and data visualization

YUHONG ZHANG

22/01/24

Table of contents

1. Using the `delay_2022` data, plot the five stations with the highest mean delays. Facet the graph by `line`

```
library(opendatatoronto)
library(tidyverse)
library(stringr)
library(skimr) # EDA
library(visdat) # EDA
library(janitor)
library(lubridate)
library(ggrepel)
```

1. Using the delay_2022 data, plot the five stations with the highest mean delays. Facet the graph by line .

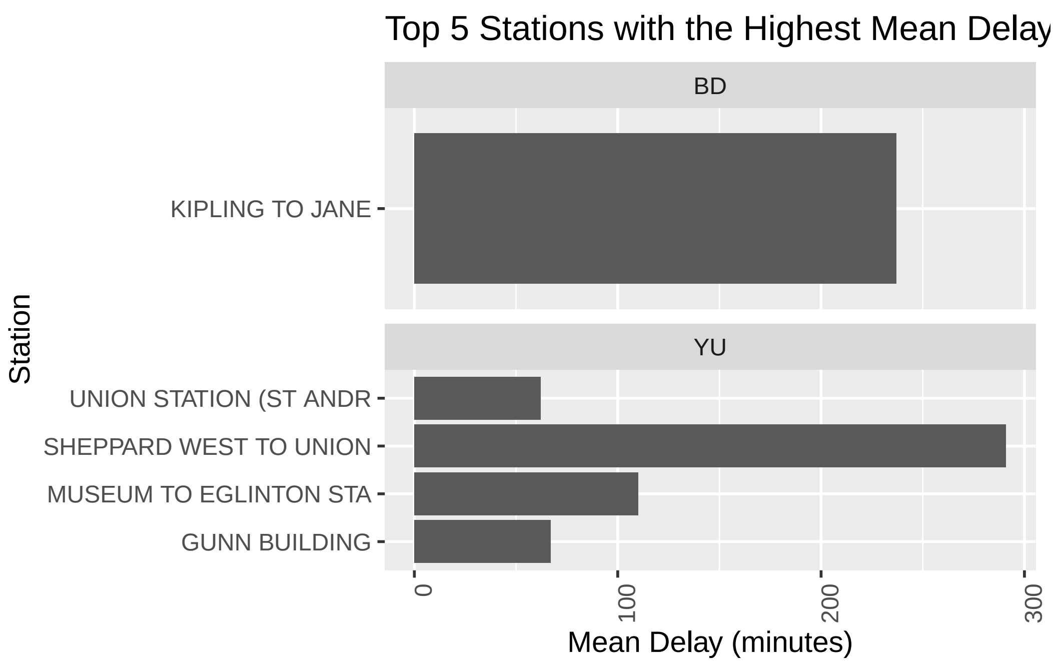
```
library(ggplot2)
library(dplyr)
library(readr)
delay_2022 <- read_csv("delay_2022.csv")

mean_delays <- delay_2022 |>
  group_by(station) |>
  summarize(mean_delay = mean(min_delay, na.rm = TRUE)) |>
  arrange(desc(mean_delay)) |>
  head(5)
mean_delays
```

```
# A tibble: 5 x 2
  station                   mean_delay
  <chr>                          <dbl>
1 SHEPPARD WEST TO UNION           291
2 KIPLING TO JANE                  237
3 MUSEUM TO EGLINTON STA           110
4 GUNN BUILDING                     67
5 UNION STATION (ST ANDR            62
```

```r
delay_2022_filtered <- delay_2022 |>
  inner_join(mean_delays, by = "station")

ggplot(delay_2022_filtered, aes(x = station, y = mean_delay)) +
  geom_bar(stat = "identity") +
  facet_wrap(vars(line),
             scales = "free_y",
             nrow = 4) +
  coord_flip()+
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Top 5 Stations with the Highest Mean Delays",
       x = "Station", y = "Mean Delay (minutes)")
```



2. Restrict the `delay_2022` to delays that are greater than 0 and to only have delay reasons

that appear in the top 50% of most frequent delay reasons. Perform a regression to study the association between delay minutes, and two covariates: line and delay reason. It's up to you how to specify the model, but make sure it's appropriate to the data types. Comment briefly on the results, including whether results generally agree with the exploratory data analysis above.

```r
#Identify the top 50% of delay reasons
delay_2022_50 <- delay_2022 %>%
  filter(min_delay > 0) %>%
  group_by(code_red) %>%
  summarise(n = n()) %>%
  mutate(freq = n / sum(n),na.rm = TRUE) %>%
  arrange(desc(freq)) %>%
  filter(cumsum(freq) <= 0.5,na.rm = TRUE) %>%
  select(code_red)
#Filter the original dataset based on the top 50% delay reasons and min_delay>0
delay_2022_filtered<-delay_2022 %>%
  filter(min_delay>0, code_red %in% delay_2022_50$code_red)

model <- lm(min_delay ~ line + code_red, data = delay_2022_filtered)
summary(model)
```

```
Call:
lm(formula = min_delay ~ line + code_red, data = delay_2022_filtered)

Residuals:
    Min      1Q  Median      3Q     Max
-11.299  -2.665  -1.408   0.592 150.317

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)           5.7892     0.3004  19.269  < 2e-16 ***
lineSHP               0.8126     0.5071   1.602 0.109125
lineSRT               5.6885     0.6768   8.405  < 2e-16 ***
lineYU               -0.3291     0.2224  -1.480 0.139007
code_redDisorderly    0.9477     0.2857   3.317 0.000917 ***
code_redInjured       2.8216     0.3147   8.967  < 2e-16 ***
code_redNo Operator  -1.4177     0.3303  -4.291 1.81e-05 ***
code_redOPTO         -0.8978     0.2910  -3.085 0.002047 **
code_redPassenger     1.2049     0.2863   4.209 2.62e-05 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.587 on 4467 degrees of freedom
Multiple R-squared:  0.07062,   Adjusted R-squared:  0.06895
F-statistic: 42.43 on 8 and 4467 DF,  p-value: < 2.2e-16
```

```r
unique(delay_2022_filtered$line)
```

```
[1] "YU"  "BD"  "SRT" "SHP"
```

Based on the result, since min_delay is a continuous variable and will be influenced by reasons, so we fit a linear regression there. We can find the results do not generally agree with the exploratory data analysis above. Since in the last question, the five stations with the highest mean delays always occurs on Line YU and BD, however, if we check the coefficient, when line is YU and code_red keeps the unchanged, compare to line is BD (baseline), the average estimated delay time will decrease by 0.3291 minutes, which is not consistent with the eda result. Meanwhile, the r square is about 0.07, which means the model is not fitted data well, so we need to find a better model to fit data.

3. Using the `opendatatoronto` package, download the data on mayoral campaign contributions for 2014 and clean it up. Hints:

    - find the ID code you need for the package you need by searching for 'campaign' in the `all_data` tibble above
    - you will then need to `list_package_resources` to get ID for the data file
    - note: the 2014 file you will get from `get_resource` has a bunch of different campaign contributions, so just keep the data that relates to the Mayor election
    - clean up the data format (fixing the parsing issue and standardizing the column names using `janitor`)

```r
library(opendatatoronto)
library(janitor)
all_data <- search_packages("campaign")
campaign_data_id <- all_data$id
campaign_data_id
```

```
[1] "e869d365-2c15-4893-ad2a-744ca867be3b"
[2] "7d0df7b0-6a0a-49a1-aadc-28b1221fa379"
```

```r
resources <- list_package_resources(campaign_data_id[1])
resources
```

```
# A tibble: 4 x 4
  name                                id                        format last_modified
  <chr>                               <chr>                     <chr>  <date>
1 Campaign Contributions 2018 Data    5f54ab3d-44d7-4e5c-9c~ ZIP    2023-04-26
2 Campaign Contributions 2018 Readme eea9eecd-75ba-4a27-9f~ XLSX   2023-04-26
3 Campaign Contributions 2014 Data    8b42906f-c894-4e93-a9~ ZIP    2023-04-26
4 Campaign Contributions 2014 Readme 10158522-4f3b-4957-9f~ XLS    2023-04-26
```

```r
mayor_campaign_data <- get_resource('8b42906f-c894-4e93-a98e-acac200f34a4')
mayor_contributions <-  mayor_campaign_data[[2]]
colnames(mayor_contributions) <- as.character(mayor_contributions[1, ])
mayor_contributions <- mayor_contributions[-1, ]
rownames(mayor_contributions) <- NULL
clean_mayor_contributions <- mayor_contributions %>%
  clean_names()
clean_mayor_contributions
```

```
# A tibble: 10,199 x 13
   contributors_name  contributors_address contributors_postal_code
   <chr>              <chr>                <chr>
 1 A D'Angelo, Tullio <NA>                 M6A 1P5
 2 A Strazar, Martin  <NA>                 M2M 3B8
 3 A'Court, K Susan   <NA>                 M4M 2J8
 4 A'Court, K Susan   <NA>                 M4M 2J8
 5 A'Court, K Susan   <NA>                 M4M 2J8
 6 Aaron, Robert B    <NA>                 M6B 1H7
 7 Abadi, Babak       <NA>                 M5S 2W7
 8 Abadi, Babak       <NA>                 M5S 2W7
 9 Abadi, David       <NA>                 M5S 2W7
10 Abate, Frank       <NA>                 L4H 2K7
# i 10,189 more rows
# i 10 more variables: contribution_amount <chr>, contribution_type_desc <chr>,
#   goods_or_service_desc <chr>, contributor_type_desc <chr>,
#   relationship_to_candidate <chr>, president_business_manager <chr>,
#   authorized_representative <chr>, candidate <chr>, office <chr>, ward <chr>
```

4. Summarize the variables in the dataset. Are there missing values, and if so, should we be worried about them? Is every variable in the format it should be? If not, create new variable(s) that are in the right format.

```r
skim(clean_mayor_contributions)
```

Table 1: Data summary

Name	clean_mayor_contributions
Number of rows	10199
Number of columns	13
Column type frequency:	
character	13
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
contributors_name	0	1	4	31	0	7545	0
contributors_address	10197	0	24	26	0	2	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
contributors_postal_code	0	1	7	7	0	5284	0
contribution_amount	0	1	1	18	0	209	0
contribution_type_desc	0	1	8	14	0	2	0
goods_or_service_desc	10188	0	11	40	0	9	0
contributor_type_desc	0	1	10	11	0	2	0
relationship_to_candidate	10166	0	6	9	0	2	0
president_business_manager	10197	0	13	16	0	2	0
authorized_representative	10197	0	13	16	0	2	0
candidate	0	1	9	18	0	27	0
office	0	1	5	5	0	1	0
ward	10199	0	NA	NA	0	0	0

```
clean_mayor_contributions <- mayor_contributions %>%
  clean_names()
na_columns <- sapply(clean_mayor_contributions, function(x) all(!is.na(x)))
df_cleaned <- clean_mayor_contributions[, na_columns]
df_cleaned$contribution_amount<-as.numeric(as.character(df_cleaned$contribution_amount))
skim(df_cleaned)
```

Table 3: Data summary

Name	df_cleaned
Number of rows	10199
Number of columns	7
Column type frequency:	
character	6
numeric	1
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
contributors_name	0	1	4	31	0	7545	0
contributors_postal_code	0	1	7	7	0	5284	0
contribution_type_desc	0	1	8	14	0	2	0
contributor_type_desc	0	1	10	11	0	2	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
candidate	0	1	9	18	0	27	0
office	0	1	5	5	0	1	0

Variable type: numeric

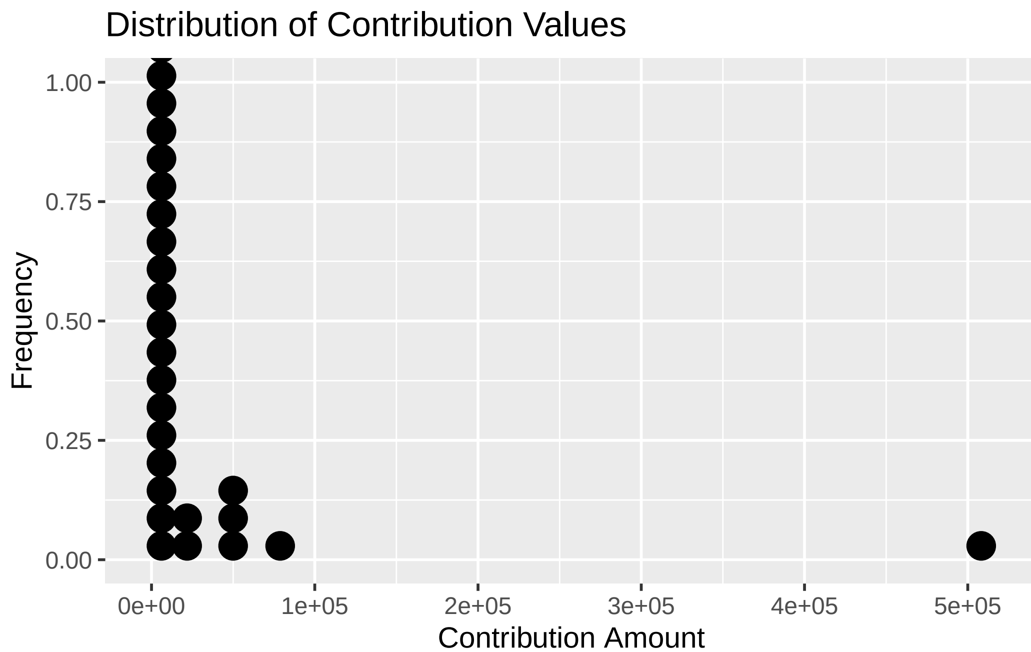
skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
contribution_amount	0	1	607.95	5211.31	1	100	300	500	508224.7	

There are some missing values, some columns are almost empty, which are 'contributors_address', ' goods_or_service_desc', ' relationship_to_candidate', ' president_business_manager', 'authorized_representative' and 'ward', may be due to lack of information or privacy reasons. The variable type of 'contribution_amount' is not correct, since it describes the amount of money of contribution, so it should be numeric variable instead of character, so I change the type of it as numeric. For other variables, the types of them are character which are correct(some can be changed to factor, such as 'contribution_type_desc' and 'contributor_type_desc'), since they formed by letters or mixing letter and numbers.

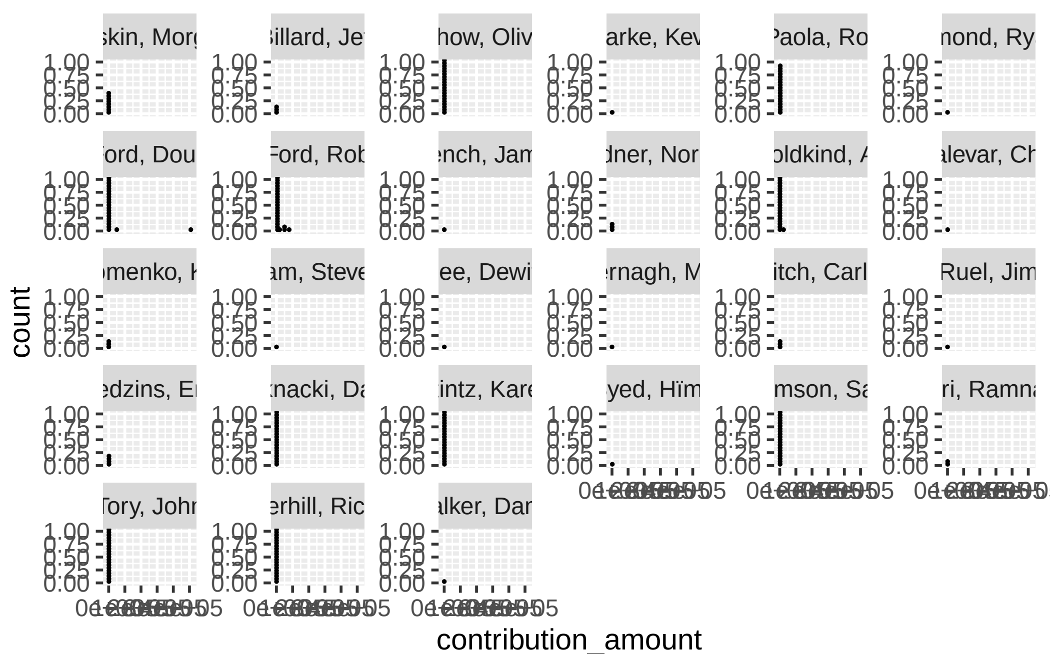
5. Visually explore the distribution of values of the contributions. What contributions are notable outliers? Do they share a similar characteristic(s)? It may be useful to plot the distribution of contributions without these outliers to get a better sense of the majority of the data.

```
df_cleaned$contribution_amount<-as.numeric(df_cleaned$contribution_amount)

ggplot(df_cleaned, aes(x = contribution_amount)) +
  geom_dotplot() +
  labs(title = "Distribution of Contribution Values",
       x = "Contribution Amount", y = "Frequency")
```




```
ggplot(df_cleaned,aes(contribution_amount)) + geom_dotplot() + facet_wrap(~candidate, scal
```



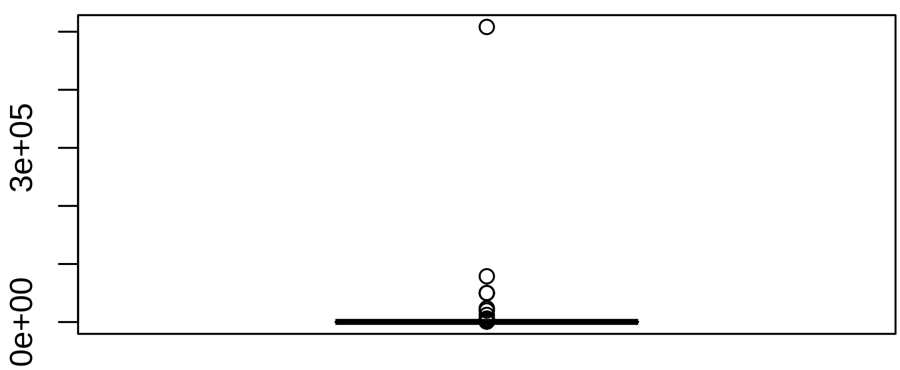
```r
IQR(df_cleaned$contribution_amount)
```

```
[1] 400
```

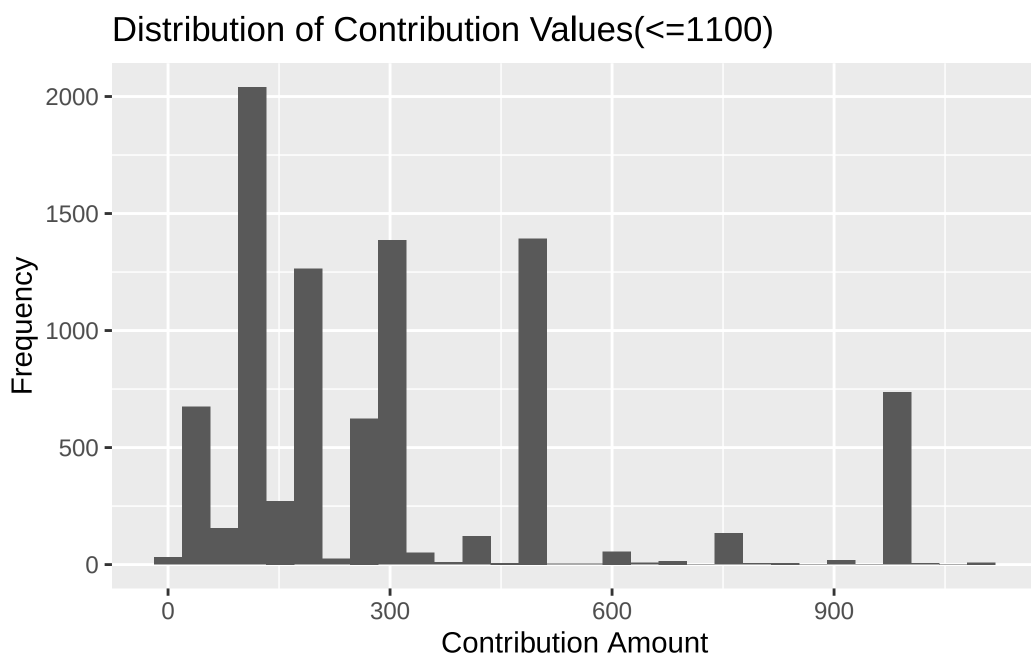
```r
df_cleaned%>% filter(contribution_amount>=1100) %>%
  arrange(-contribution_amount)%>%
  head(10)
```

```
# A tibble: 10 x 7
   contributors_name contributors_postal_code contribution_amount
   <chr>             <chr>                                  <dbl>
 1 Ford, Doug        M9A 2C3                              508225.
 2 Ford, Rob         M9A 3G9                               78805.
 3 Ford, Doug        M9A 2C3                               50000
 4 Ford, Rob         M9A 3G9                               50000
 5 Ford, Rob         M9A 3G9                               50000
 6 Goldkind, Ari     M5P 1P5                               23624.
 7 Ford, Rob         M9A 3G9                               20000
 8 Ford, Rob         M9A 3G9                               12210
 9 Di Paola, Rocco   M3H 2T1                                6000
10 Thomson, Sarah    M4W 2X6                                4426.
# i 4 more variables: contribution_type_desc <chr>,
#   contributor_type_desc <chr>, candidate <chr>, office <chr>
```

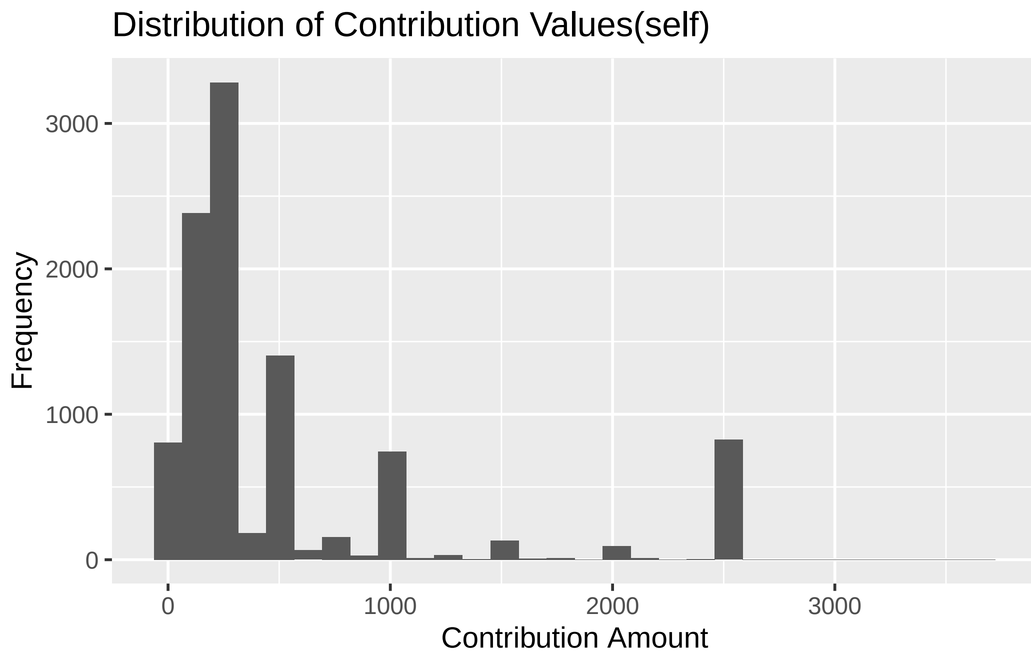
```r
boxplot(df_cleaned$contribution_amount)
```



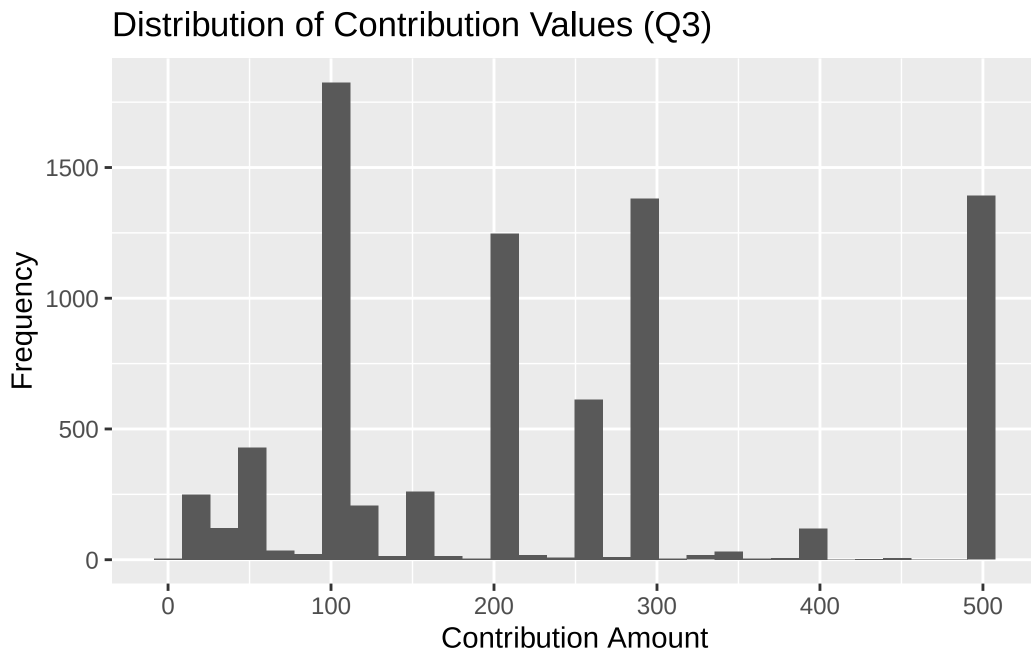
```
outlier4<-df_cleaned%>%
         filter(contribution_amount<=1100)
ggplot(outlier4, aes(x = contribution_amount)) +
  geom_histogram() +
  labs(title = "Distribution of Contribution Values(<=1100)",
       x = "Contribution Amount", y = "Frequency")
```




```
outlier5<-df_cleaned%>%
         filter(contribution_amount<4000)
ggplot(outlier5, aes(x = contribution_amount)) +
  geom_histogram() +
  labs(title = "Distribution of Contribution Values(self)",
       x = "Contribution Amount", y = "Frequency")
```




```
outlier6<-df_cleaned%>%
         filter(contribution_amount<501)
ggplot(outlier6, aes(x = contribution_amount)) +
  geom_histogram() +
  labs(title = "Distribution of Contribution Values (Q3)",
       x = "Contribution Amount", y = "Frequency")
```



From previous question, we know the mean is 607.9521 and 75th percentile is 500 and max is 508224.7, from previous courses, we know the outlier is greater than Q3 + (1.5 * IQR) = 500+400*1.5 = 1100. Some outliers, such as self donation of 508224.7 by Doug Ford and 78804.8, 50000.0 and some other notable outliers, we can find that all donations that greater than 4000 are contributed by the candidates themselves. These outliers lead the graph shows right skewed. If we filter for less than 1100 contributions, we see that the distribution is less skewed and get a better sense of the majority of the data.

6. List the top five candidates in each of these categories:

   - total contributions
   - mean contribution
   - number of contributions

```
candidate_stats <- df_cleaned %>%
  group_by(candidate) %>%
  summarise(
    total_contributions = sum(contribution_amount, na.rm = TRUE),
    mean_contribution = mean(contribution_amount, na.rm = TRUE),
    number_of_contributions = n()
  )
```

```r
top_total_contributions <- candidate_stats %>%
  arrange(desc(total_contributions)) %>%
  select(candidate,total_contributions)%>%
  head(5)

top_mean_contribution <- candidate_stats %>%
  arrange(desc(mean_contribution)) %>%
  select(candidate,mean_contribution)%>%
  head(5)

top_number_of_contributions <- candidate_stats %>%
  arrange(desc(number_of_contributions)) %>%
  select(candidate,number_of_contributions)%>%
  head(5)

top_total_contributions
```

```
# A tibble: 5 x 2
  candidate     total_contributions
  <chr>                       <dbl>
1 Tory, John               2767869.
2 Chow, Olivia             1638266.
3 Ford, Doug                889897.
4 Ford, Rob                 387648.
5 Stintz, Karen             242805
```

```r
top_mean_contribution
```

```
# A tibble: 5 x 2
  candidate        mean_contribution
  <chr>                        <dbl>
1 Sniedzins, Erwin             2025
2 Syed, Himy                   2018
3 Ritch, Carlie                1887.
4 Ford, Doug                   1456.
5 Clarke, Kevin                1200
```

```r
top_number_of_contributions
```

```
# A tibble: 5 x 2
  candidate       number_of_contributions
  <chr>                             <int>
1 Chow, Olivia                       5708
2 Tory, John                         2602
3 Ford, Doug                          611
4 Ford, Rob                           538
5 Soknacki, David                     314
```

7. Repeat 6 but without contributions from the candidates themselves.

```r
df_without_self_contributions <- df_cleaned %>%
  filter(contributors_name != candidate)

candidate_stats_self <- df_without_self_contributions %>%
  group_by(candidate) %>%
  summarise(
    total_contributions_self = sum(contribution_amount, na.rm = TRUE),
    mean_contribution_self = mean(contribution_amount, na.rm = TRUE),
    number_of_contributions_self = n()
  )

top_total_contributions_self <- candidate_stats_self %>%
  arrange(desc(total_contributions_self)) %>%
  select(candidate,total_contributions_self)%>%
  head(5)

top_mean_contribution_self <- candidate_stats_self %>%
  arrange(desc(mean_contribution_self)) %>%
  select(candidate,mean_contribution_self)%>%
  head(5)

top_number_of_contributions_self <- candidate_stats_self %>%
  arrange(desc(number_of_contributions_self)) %>%
  select(candidate,number_of_contributions_self)%>%
  head(5)

top_total_contributions_self
```

```
# A tibble: 5 x 2
  candidate     total_contributions_self
  <chr>                            <dbl>
```

```
1 Tory, John                    2765369.
2 Chow, Olivia                  1634766.
3 Ford, Doug                     331173.
4 Stintz, Karen                  242805
5 Ford, Rob                      174510.
```

```
top_mean_contribution_self
```

```
# A tibble: 5 x 2
  candidate         mean_contribution_self
  <chr>                              <dbl>
1 Ritch, Carlie                      1887.
2 Sniedzins, Erwin                   1867.
3 Tory, John                         1063.
4 Gardner, Norman                    1000
5 Tiwari, Ramnarine                  1000
```

```
top_number_of_contributions_self
```

```
# A tibble: 5 x 2
  candidate       number_of_contributions_self
  <chr>                                  <int>
1 Chow, Olivia                            5706
2 Tory, John                              2601
3 Ford, Doug                               608
4 Ford, Rob                                531
5 Soknacki, David                          314
```

8. How many contributors gave money to more than one candidate?

```
contributors_multiple_candidates <- df_cleaned %>%
  group_by(contributors_name) %>%
  summarise(unique_candidates = n_distinct(candidate))

num_contributors_multiple_candidates <- sum(contributors_multiple_candidates$unique_candid

num_contributors_multiple_candidates
```

```
[1] 184
```

There are 184 contributors gave money to more than one candidate.