lab_exercise2

```
library(opendatatoronto)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.2     v readr     2.1.4
v forcats   1.0.0     v stringr   1.5.0
v ggplot2   3.4.2     v tibble    3.2.1
v lubridate 1.9.2     v tidyr     1.3.0
v purrr     1.0.1
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(stringr)
library(skimr) # EDA
library(visdat) # EDA
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```
library(lubridate)
library(ggrepel)
```

```
# Load Data
all_data <- list_packages(limit = 500)
res <- list_package_resources("996cfe8d-fb35-40ce-b569-698d51fc683b") # obtained code from
res <- res |> mutate(year = str_extract(name, "202.?"))
delay_2022_ids <- res |> filter(year==2022) |> select(id) |> pull()

delay_2022 <- get_resource(delay_2022_ids)

# Make the column names nicer to work with
delay_2022 <- clean_names(delay_2022)
# Delay code
delay_codes <- get_resource("3900e649-f31e-4b79-9f20-4731bbfd94f7")
```

```
New names:
* `` -> `...1`
* `CODE DESCRIPTION` -> `CODE DESCRIPTION...3`
* `` -> `...4`
* `` -> `...5`
* `CODE DESCRIPTION` -> `CODE DESCRIPTION...7`
```

```
# Data cleaning
delay_2022 <- delay_2022 |>
  mutate(contains_yu_bd = str_detect(str_to_lower(line), "bd")&str_detect(str_to_lower(lin
  mutate(line = ifelse(contains_yu_bd, ifelse(line=="YU/BD", line, "YU/BD"), line))  |>
  select(-contains_yu_bd)


delay_2022 <- delay_2022 |> filter(line %in% c("BD", "YU", "SHP", "SRT", "YU/BD"))
# Delay reason
delay_2022 <- delay_2022 |>
  left_join(delay_codes |> rename(code = `SUB RMENU CODE`, code_desc = `CODE DESCRIPTION..
```

```
Joining with `by = join_by(code)`
```

```
delay_2022 <- delay_2022 |>
  mutate(code_srt = ifelse(line=="SRT", code, "NA")) |>
  left_join(delay_codes |> rename(code_srt = `SRT RMENU CODE`, code_desc_srt = `CODE DESCR
  mutate(code = ifelse(code_srt=="NA", code, code_srt),
         code_desc = ifelse(is.na(code_desc_srt), code_desc, code_desc_srt)) |>
```

```
    select(-code_srt, -code_desc_srt)
```

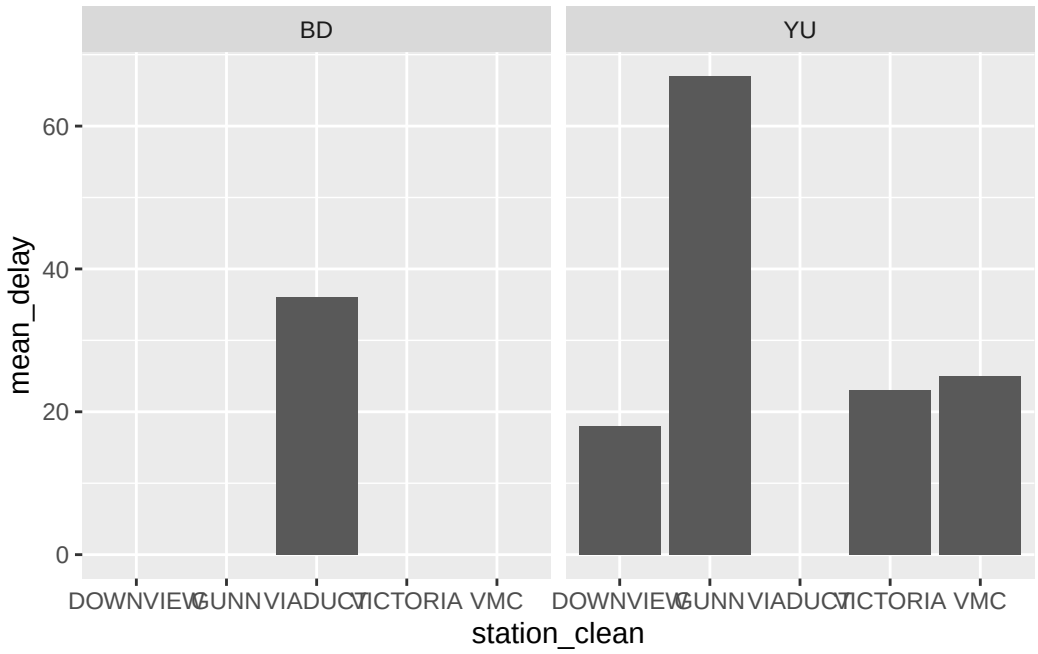
```
Joining with `by = join_by(code_srt)`
```

```
  delay_2022 <- delay_2022 |>
    mutate(station_clean = ifelse(str_starts(station, "ST"), word(station, 1,2), word(statio
```

1

```
  # Code to plot the top five stations with the highest mean delays, faceted by line
  delay_2022 |>
    group_by(station_clean,line) |>
    summarise(mean_delay=mean(min_delay)) |>
    arrange(-mean_delay)|>
    head(5) |>
    ggplot(aes(x = station_clean,
               y = mean_delay))+
    geom_col()+
    facet_wrap(~line)
```

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



2

The regression model seems to have less explanation power on min_delay variable since adjusted r-squared is 0.1596. However, it appears that several code_desc and the line categories have significant associations with the min_delay variable. It seems that regression analysis result does not agree with the EDA since EDA shows that delays were caused by Fire and Rail Related Problem whereas regression result shows that Passenger related problem plays significant role in delay.

```
# Restrict the dataset to delays greater than 0
delay_2022_filtered <- delay_2022 |>
  filter(min_delay > 0)

# Identify the top 50% most frequent delay reasons
top_reasons <- delay_2022_filtered |>
  group_by(code_desc) |>
  summarise(cnt = length(code_desc)) |>
  arrange(desc(cnt)) |>
  mutate(cumulative_sum = cumsum(cnt)) |>
  mutate(half = sum(cnt)/2) |>
  filter(cumulative_sum<=half)
```

```r
# Further filtering the dataset for these reasons
delay_2022_filtered <- delay_2022_filtered |>
  filter(code_desc %in% top_reasons$code_desc)


# Convert character variables to factors for the regression model
delay_2022_filtered$line <- as.factor(delay_2022_filtered$line)
delay_2022_filtered$code_desc <- as.factor(delay_2022_filtered$code_desc)

# Fit a linear regression model
regression_model <- lm(min_delay ~ line + code_desc, data = delay_2022_filtered)

# Output the summary of the model
summary(regression_model)
```

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

Residuals:
    Min      1Q  Median      3Q     Max
-14.204  -2.450  -0.984   0.870 227.747

Coefficients:
                                                                  Estimate
(Intercept)                                                         5.7324
lineSHP                                                             1.3993
lineSRT                                                             3.9510
lineYU                                                             -0.2829
code_descDisorderly Patron                                          0.9983
code_descInjured or ill Customer (On Train) - Medical Aid Refused   1.5209
code_descNo Operator Immediately Available                         -1.4656
code_descOPTO (COMMS) Train Door Monitoring                        -0.9371
code_descPassenger Assistance Alarm Activated - No Trouble Found   -1.7019
code_descPassenger Other                                            5.6523
code_descUnauthorized at Track Level                                7.5205
                                                                  Std. Error
(Intercept)                                                           0.3705
lineSHP                                                               0.6199
lineSRT                                                               0.8097
lineYU                                                                0.2678
code_descDisorderly Patron                                            0.3545
```

```
code_descInjured or ill Customer (On Train) - Medical Aid Refused     0.4705
code_descNo Operator Immediately Available                            0.4163
code_descOPTO (COMMS) Train Door Monitoring                           0.3622
code_descPassenger Assistance Alarm Activated - No Trouble Found      0.3965
code_descPassenger Other                                              0.4598
code_descUnauthorized at Track Level                                  0.4300
                                                                    t value
(Intercept)                                                          15.474
lineSHP                                                               2.257
lineSRT                                                               4.879
lineYU                                                               -1.056
code_descDisorderly Patron                                            2.816
code_descInjured or ill Customer (On Train) - Medical Aid Refused     3.233
code_descNo Operator Immediately Available                           -3.520
code_descOPTO (COMMS) Train Door Monitoring                          -2.587
code_descPassenger Assistance Alarm Activated - No Trouble Found     -4.292
code_descPassenger Other                                             12.294
code_descUnauthorized at Track Level                                 17.490
                                                                    Pr(>|t|)
(Intercept)                                                          < 2e-16 ***
lineSHP                                                             0.024048 *
lineSRT                                                             1.10e-06 ***
lineYU                                                              0.290818
code_descDisorderly Patron                                          0.004886 **
code_descInjured or ill Customer (On Train) - Medical Aid Refused   0.001235 **
code_descNo Operator Immediately Available                          0.000435 ***
code_descOPTO (COMMS) Train Door Monitoring                         0.009711 **
code_descPassenger Assistance Alarm Activated - No Trouble Found    1.81e-05 ***
code_descPassenger Other                                             < 2e-16 ***
code_descUnauthorized at Track Level                                 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.788 on 4474 degrees of freedom
Multiple R-squared:  0.1615,    Adjusted R-squared:  0.1596
F-statistic: 86.16 on 10 and 4474 DF,  p-value: < 2.2e-16
```

3

```
# Compaign Contribution
res <- list_package_resources("e869d365-2c15-4893-ad2a-744ca867be3b")
res <- res |> mutate(year = str_extract(name, "201.?"))
compaign_2014_ids <- res |> filter(year==2014) |> select(id) |> pull()
compaign_2014_ids
```

```
[1] "8b42906f-c894-4e93-a98e-acac200f34a4"
[2] "10158522-4f3b-4957-9f01-4bcfbc87e357"
```

Now, we found data id.

```
compaign_2014 <- get_resource("8b42906f-c894-4e93-a98e-acac200f34a4")
```

```
New names:
New names:
New names:
New names:
New names:
New names:
New names:
* `` -> `...2`
* `` -> `...3`
```

```
# Data processing
mayor_compaign_2014 <- compaign_2014$`2_Mayor_Contributions_2014_election.xls`
colnames(mayor_compaign_2014) <- as.character(mayor_compaign_2014[1,]) # colnames are in t
mayor_compaign_2014 <- mayor_compaign_2014[-1,]
mayor_compaign_2014 <- clean_names(mayor_compaign_2014)
# Final result
head(mayor_compaign_2014)
```

```
# A tibble: 6 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
# 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

As we can see n_missing below, there are columns such as 'contributors_address','president_business_manager', 'relationship_to_candidate' which are almost blank. We should concern this issue becuase these missing values would significantly affect model performance. In addition, all covariates are character type. However, 'contribution_amount' should be numerical variable whereas 'contributor_type_desc' should be recorded as factor variable. Hence, we create new data-type-corrected variables 'contributor_type_desc_fac' and 'contribution_amount_num', respectly.

```
# Summarize the data (find data type, number of missing data)
skim(mayor_compaign_2014)
```

Table 1: Data summary

Name	mayor_compaign_2014
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
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

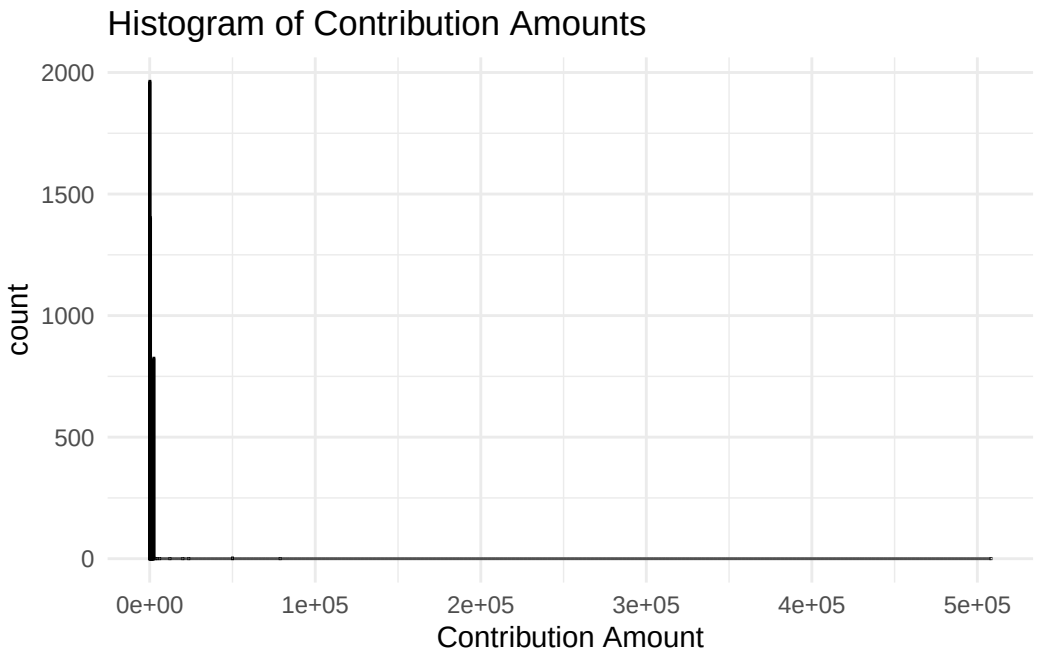
skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
goods_or_service_desc	10188	0	11	40	0	9	0
contributor_type_desc	0	1	10	11	0	2	0
relationship_to_candidate	10166	0	6	9	0	2	0
president_business_manager	10197	0	13	16	0	2	0
authorized_representative	10197	0	13	16	0	2	0
candidate	0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

```
# Create new variables
mayor_compaign_2014$contributor_type_desc_fac <- as.factor(mayor_compaign_2014$contributor
mayor_compaign_2014$contribution_amount_num <- as.numeric(mayor_compaign_2014$contribution
```

5

We first see the distribution of the contribution. As we can see, it is extremly skewed distribution. Next, we determine outliers using IQR and upper whisker, and show the outliers in Box plot. Main outliers are contribution of over $500,000 done by Doug Ford and multiple contribution over 50,000 done by Rob Ford. They share similar characteristic that 'contributors_name' and 'candidate' are the same. It means that the outliers are contributions from the candidates themselves. Lastly, we plot historgram without outliers.

```
# Distribution
ggplot(mayor_compaign_2014, aes(x = contribution_amount_num)) +
  geom_histogram(binwidth = 50, colour = "black", fill = "white") +
  theme_minimal() +
  labs(x = "Contribution Amount", title = "Histogram of Contribution Amounts")
```

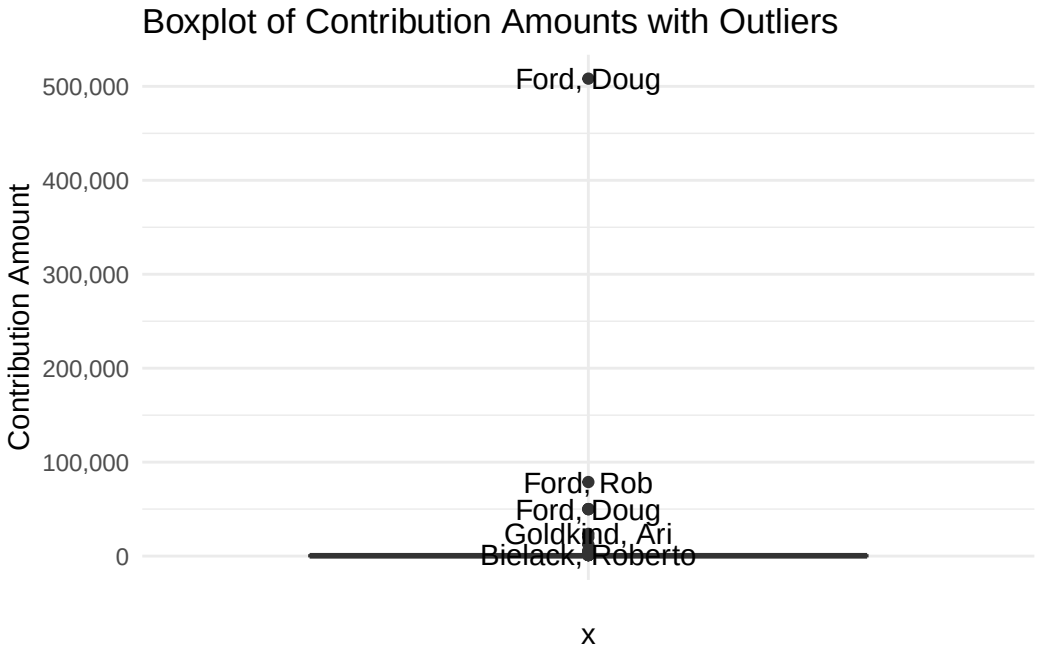



```
library(scales)
```

```
Attaching package: 'scales'

The following object is masked from 'package:purrr':

    discard

The following object is masked from 'package:readr':

    col_factor
```

```
# Calculate IQR to determine outliers
IQR_value <- IQR(mayor_compaign_2014$contribution_amount_num, na.rm = TRUE)
upper_whisker <- quantile(mayor_compaign_2014$contribution_amount_num, 0.75, na.rm = TRUE)
outliers <- mayor_compaign_2014 |>
  filter(contribution_amount_num > upper_whisker) |>
  arrange(desc(contribution_amount_num))

# Boxplot with outlier names
```

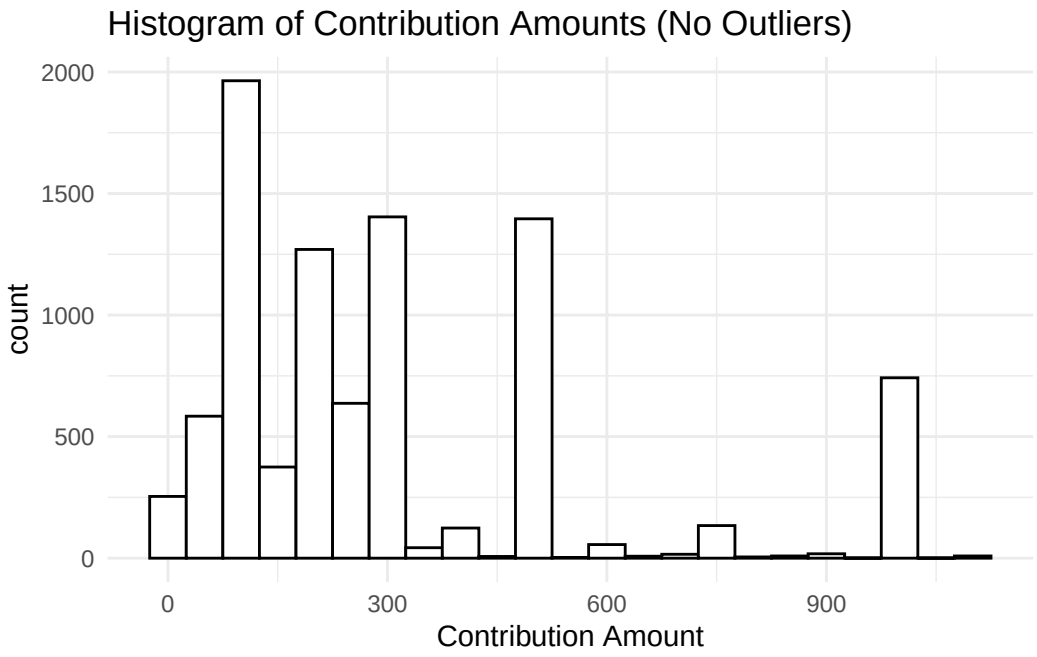
```
ggplot(mayor_compaign_2014, aes(x = "", y = contribution_amount_num)) +
  geom_boxplot() +
  geom_text(data = outliers, aes(label = contributors_name), nudge_y = 1, check_overlap =
  scale_y_continuous(labels = scales::comma) +
  theme_minimal() +
  labs(y = "Contribution Amount", title = "Boxplot of Contribution Amounts with Outliers")
```




```
# Characteristics of outliers
head(outliers |> select(candidate, contributors_name))
```

```
# A tibble: 6 x 2
  candidate     contributors_name
  <chr>         <chr>
1 Ford, Doug    Ford, Doug
2 Ford, Rob     Ford, Rob
3 Ford, Doug    Ford, Doug
4 Ford, Rob     Ford, Rob
5 Ford, Rob     Ford, Rob
6 Goldkind, Ari Goldkind, Ari
```

```r
# Exclude outliers before plotting
non_outliers <- mayor_compaign_2014 |>
  filter(contribution_amount_num <= upper_whisker)

# Plot histogram of contribution amounts without outliers
ggplot(non_outliers, aes(x = contribution_amount_num)) +
  geom_histogram(binwidth = 50, colour = "black", fill = "white") +
  theme_minimal() +
  labs(x = "Contribution Amount", title = "Histogram of Contribution Amounts (No Outliers)")
```



6

```r
# Calculate total contributions for each candidate
total_contributions <- mayor_compaign_2014 |>
  group_by(candidate) |>
  summarize(total = sum(contribution_amount_num, na.rm = TRUE)) |>
  arrange(desc(total)) |>
  top_n(5)
```

```
Selecting by total
```

```
total_contributions
```

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

```
# Calculate mean contribution for each candidate
mean_contributions <- mayor_compaign_2014 |>
  group_by(candidate) |>
  summarize(mean = mean(contribution_amount_num, na.rm = TRUE)) |>
  arrange(desc(mean)) |>
  top_n(5)
```

```
Selecting by mean
```

```
mean_contributions
```

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

```
# Calculate the number of contributions for each candidate
number_contributions <- mayor_compaign_2014 |>
  group_by(candidate) |>
  summarize(count = n()) |>
  arrange(desc(count)) |>
  top_n(5)
```

```
Selecting by count
```

```
number_contributions
```

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

7

```
# First, identify contributions from the candidates themselves
self_contributions <- mayor_compaign_2014 |>
  filter(str_detect(contributors_name, candidate))

# Exclude these self contributions and calculate total, mean, and count for each candidate
mayor_compaign_no_self <- mayor_compaign_2014 |>
  filter(!contributors_name %in% self_contributions$contributors_name)

# Calculate total contributions for each candidate without self-contributions
total_contributions_no_self <- mayor_compaign_no_self |>
  group_by(candidate) |>
  summarize(total = sum(contribution_amount_num, na.rm = TRUE)) |>
  arrange(desc(total)) |>
  top_n(5)
```

```
Selecting by total
```

```
total_contributions_no_self
```

```
# A tibble: 5 x 2
  candidate        total
  <chr>            <dbl>
1 Tory, John    2765369.
2 Chow, Olivia  1634766.
3 Ford, Doug     331173.
```

```
4 Stintz, Karen  242805
5 Ford, Rob      172010.
```

```
# Calculate mean contribution for each candidate without self-contributions
mean_contributions_no_self <- mayor_compaign_no_self |>
  group_by(candidate) |>
  summarize(mean = mean(contribution_amount_num, na.rm = TRUE)) |>
  arrange(desc(mean)) |>
  top_n(5)
```

```
Selecting by mean
```

```
mean_contributions_no_self
```

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

```
# Calculate the number of contributions for each candidate without self-contributions
number_contributions_no_self <- mayor_compaign_no_self |>
  group_by(candidate) |>
  summarize(count = n()) |>
  arrange(desc(count)) |>
  top_n(5)
```

```
Selecting by count
```

```
number_contributions_no_self
```

```
# A tibble: 5 x 2
  candidate       count
  <chr>           <int>
```

```
1 Chow, Olivia     5706
2 Tory, John       2601
3 Ford, Doug        608
4 Ford, Rob         530
5 Soknacki, David   312
```

8

```r
# Group by contributors and count the number of unique candidates they donated to
contributors_multiple_candidates <- mayor_compaign_2014 |>
  group_by(contributors_name) |>
  summarize(num_candidates = n_distinct(candidate)) |>
  ungroup() |>
  filter(num_candidates > 1)

# Count the number of contributors who donated to more than one candidate
num_contributors_multiple_candidates <- nrow(contributors_multiple_candidates)

# Output the number
num_contributors_multiple_candidates
```

```
[1] 184
```