data_analysis_v2

```
# # Ensure that pacman is installed for package management and loading.
# if (!require("pacman")) install.packages("pacman")
# # for data reading wrangling and visualization
pacman::p_load(tidyverse)
# for working directories
pacman::p_load(here)
# # for cross tabulation and data cleaning
# pacman::p_load(janitor)
# for working with strings
pacman::p_load(glue)
# For randomized inference, also loads randomizr and estimatr
pacman::p_load(ri2)
# for marginal effects from lineal regressions
pacman::p_load(margins)
# Tests for linear regression models
pacman::p_load(lmtest)
pacman::p_load(car)
# Tables
pacman::p_load(kableExtra)
# for updated ggplot2 theme
pacman::p_load(hrbrthemes)
# for updated ggplot2 colorblind-friendly scheme
pacman::p_load(ggthemes)
# theme_set(hrbrthemes::theme_ipsum())
pacman::p_load(reshape2)
# for plotting of covariate balance
pacman::p_load(cobalt)
# for matching only
# pacman::p_load(MatchIt)
```

```
#download the data from GitHub
data <-'https://raw.githubusercontent.com/gsbDBI/ALP301-spr21-project3/main/rla_clean_5_12.csv'
df <- read.csv(data, strip.white = TRUE)
rm(data) #remove data csv file
```

```
#table() creates a contingency table of counts of observations at each combination of treat_pseudo and
with(df, table(treatment_group, useNA = 'ifany')) %>%  # "ifany" includes the NA values in the table
  knitr::kable() %>%                                                  #kabel(x, format) generates tables
  # add in a header to label what we're cross-tabulating with
  add_header_above(c('treat_group' = 2)) %>% #add_header_above(x, col_name=col_span)
  kableExtra::kable_styling(bootstrap_options = "striped") #additional styling options
```

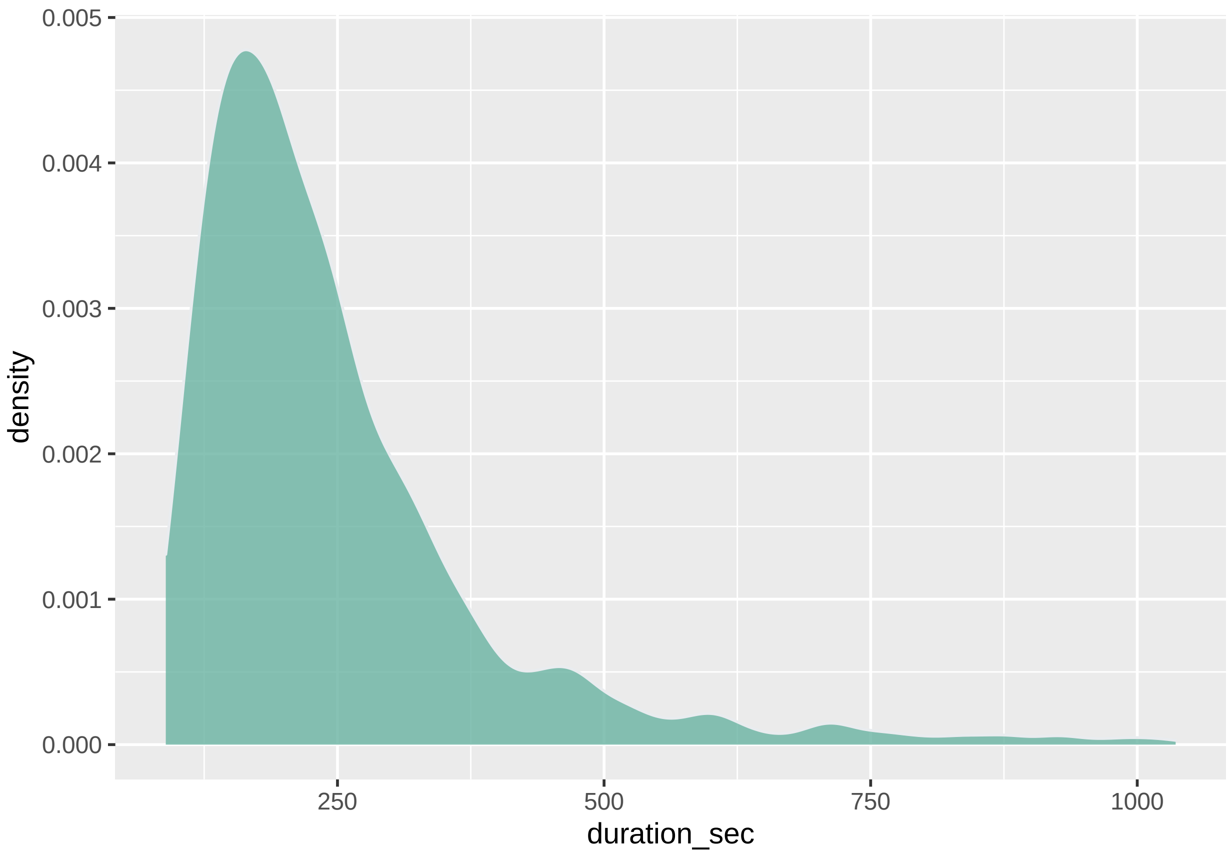
```
#see who is in the top 2.5% of least amount of time spent on the survey
quantile(df$duration_sec, c(0.025, 0.95, .975))
```

```
##     2.5%      95%    97.5%
##   90.000  684.250 1049.175
```

treat_group	
treatment_group	Freq
Bipartisan	328
Control	321
Handcount	324
Local	321
Loser	315
RL_percentage	317
Soup	324

```
# we see the duration is 88 seconds for the top 2.5% and 97.5% (those who spent the least amount)

# make a density plot of the distribution of total time spent on the survey between
df %>%
  filter( duration_sec < 1050, duration_sec > 88 ) %>%
  ggplot( aes(x=duration_sec)) +
    geom_density(fill="#69b3a2", color="#e9ecef", alpha=0.8)
```



Another way of checking for balance is to run a multivariate regression of our treatment assignment variable on covariates. With this method, instead of a t-test, we use an F-test for the hypothesis that all coefficients (non-strata) are equal to zero. If a coefficient on a covariate is not close to zero, that would mean that assignment was not balanced on that covariate.

```
# regression of covariates on treatment assignment variable
balance_lm <- lm(treatment_group_num ~ birthyear + gender + parent + state + trust_federal + trust_stat
```

2016 accuracy	
accuracy_2016	Freq
No	95
Yes	366

2020 accuracy	
accuracy_2020	Freq
	1
No	321
Yes	139

```
# uncomment to report heteroskedasticity-robust standard errors
# lmtest::coeftest(balance_lm,
#   vcov = sandwich::vcovHC(balance_lm, type = "HC2")
# )

# Test whether all coefficients from the balancce_lm regression are equal to zero
# using heteroskedasticity-robust standard errors, denoted by hc2
# car::linearHypothesis(balance_lm, c("birthyear = 0", "gender = 0", "parent = 0",
#                                     "state = 0", "trust_federal = 0",
#                                     "trust_state = 0", "accuracy_2016 = 0", "accuracy_2020 = 0", "inc
#                        test = "F", white.adjust = "hc2", singular.ok = TRUE)
```

If this F-test, if the p-value indicated by Pr(>F) is very small, for example, < 0.05, we may reject the null and conclude that at least one of the coefficient is not equal to zero and treatment assignment was not balanced on that covariate. In our case, we again see that the p-value is quite large, meaning that there is no significantly different pre-experiment covariates that determine treatment status.

```
# of those who identify as democrats how trust 2016 and 2020
dems <- df %>%
  filter(party == "Democrat")

repubs <- df %>%
  filter(party == "Republican")

with(repubs, table(accuracy_2016, useNA = 'ifany')) %>%  # "ifany" includes the NA values in the table
  knitr::kable() %>%                                      #kabel(x, format) generates tables
  # add in a header to label what we're cross-tabulating with
  add_header_above(c('2016 accuracy' = 2)) %>% #add_header_above(x, col_name=col_span)
  kableExtra::kable_styling(bootstrap_options = "striped") #additional styling options
```

```
with(repubs, table(accuracy_2020, useNA = 'ifany')) %>%  # "ifany" includes the NA values in the table
  knitr::kable() %>%                                      #kabel(x, format) generates tables
  # add in a header to label what we're cross-tabulating with
  add_header_above(c('2020 accuracy' = 2)) %>% #add_header_above(x, col_name=col_span)
  kableExtra::kable_styling(bootstrap_options = "striped") #additional styling options
```

```
##
## Please cite as:

##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

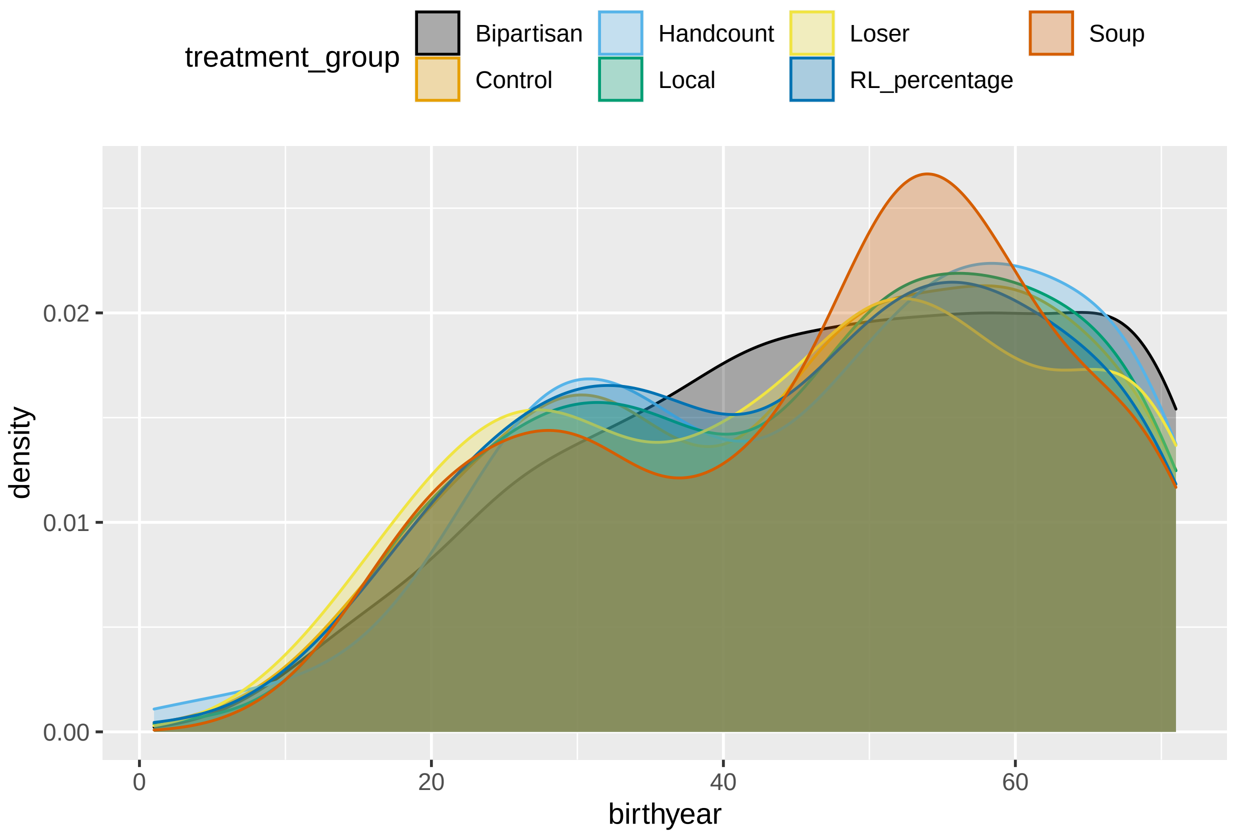
% Table created by stargazer v.5.2.2 by Marek Hlavac, Harvard University. E-mail: hlavac at fas.harvard.edu

% Date and time: Fri, May 14, 2021 - 19:26:26

Table 1: Basic Regression Results Control vs. Treated

	Dependent variable:	
	dv_post_state_conf	dv_post_national_conf
	(1)	(2)
dummy_treat	0.149*	0.293***
	(0.088)	(0.091)
dv_pre_state_conf	0.804***	
	(0.012)	
dv_pre_national_conf		0.800***
		(0.012)
Constant	1.281***	1.250***
	(0.122)	(0.114)
Observations	2,250	2,250
R^2	0.654	0.680
Adjusted R^2	0.654	0.680
Residual Std. Error (df = 2247)	1.453	1.501
F Statistic (df = 2; 2247)	2,122.232***	2,390.021***
Note:		*p<0.1; **p<0.05; ***p<0.01

```r
df %>%
  ggplot(aes(x = birthyear, # aes(x, y, ...) defines how variables are mapped into the aesthetics of th
          color = treatment_group,
          fill = treatment_group
           )
        ) +
  # Density plot
  geom_density(alpha = 0.3) +    # alpha controls the transparency
  # use a colorblind friendly color palette
  scale_color_colorblind() +
  scale_fill_colorblind() +
  theme(legend.position = "top") # legend position
```



```
df %>%
  ggplot(aes(x = gender, # aes(x, y, ...) defines how variables are mapped into the aesthetics of the p
             color = treatment_group,
             fill = treatment_group
              )
         ) +
  # Density plot
  geom_density(alpha = 0.3) +     # alpha controls the transparency
  # use a colorblind friendly color palette
  scale_color_colorblind() +
  scale_fill_colorblind() +
  theme(legend.position = "top") # legend position
```

```
## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.

## Warning: Groups with fewer than two data points have been dropped.
## Warning: Groups with fewer than two data points have been dropped.
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

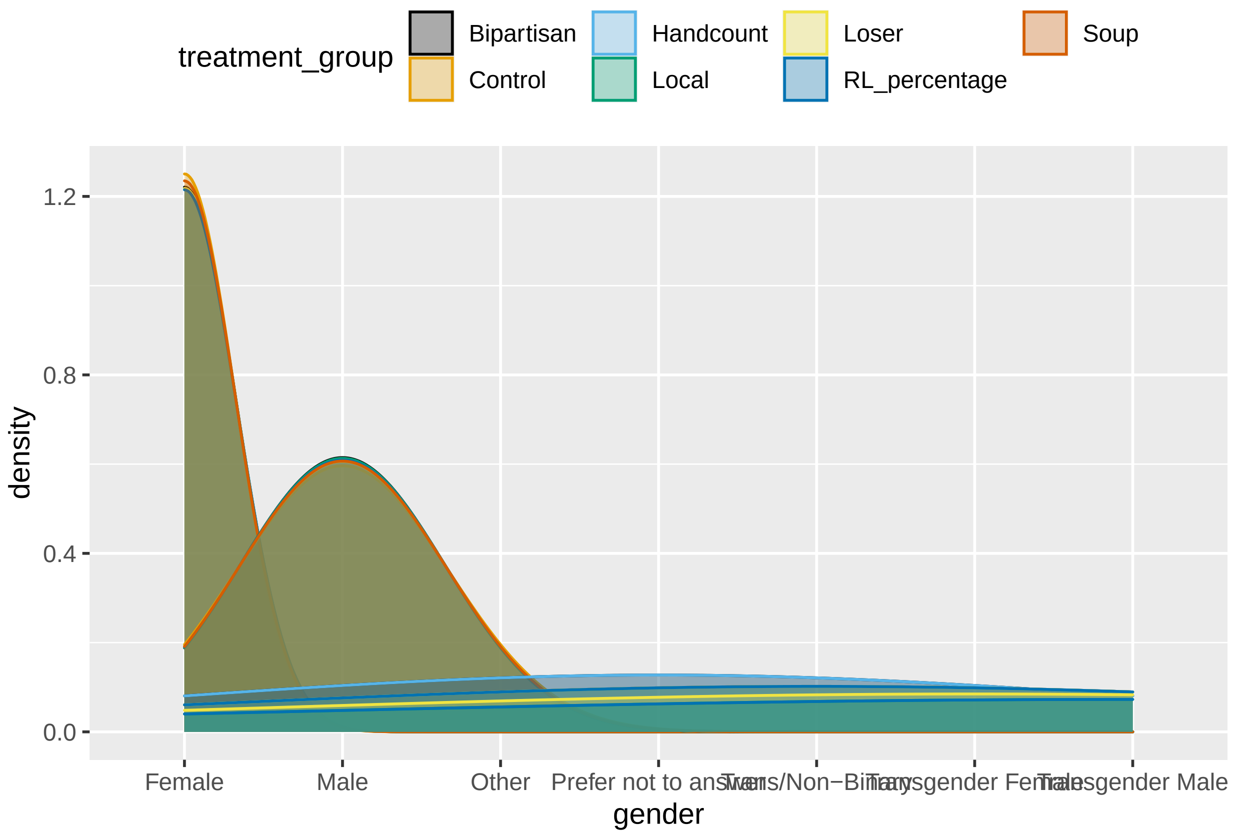
```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

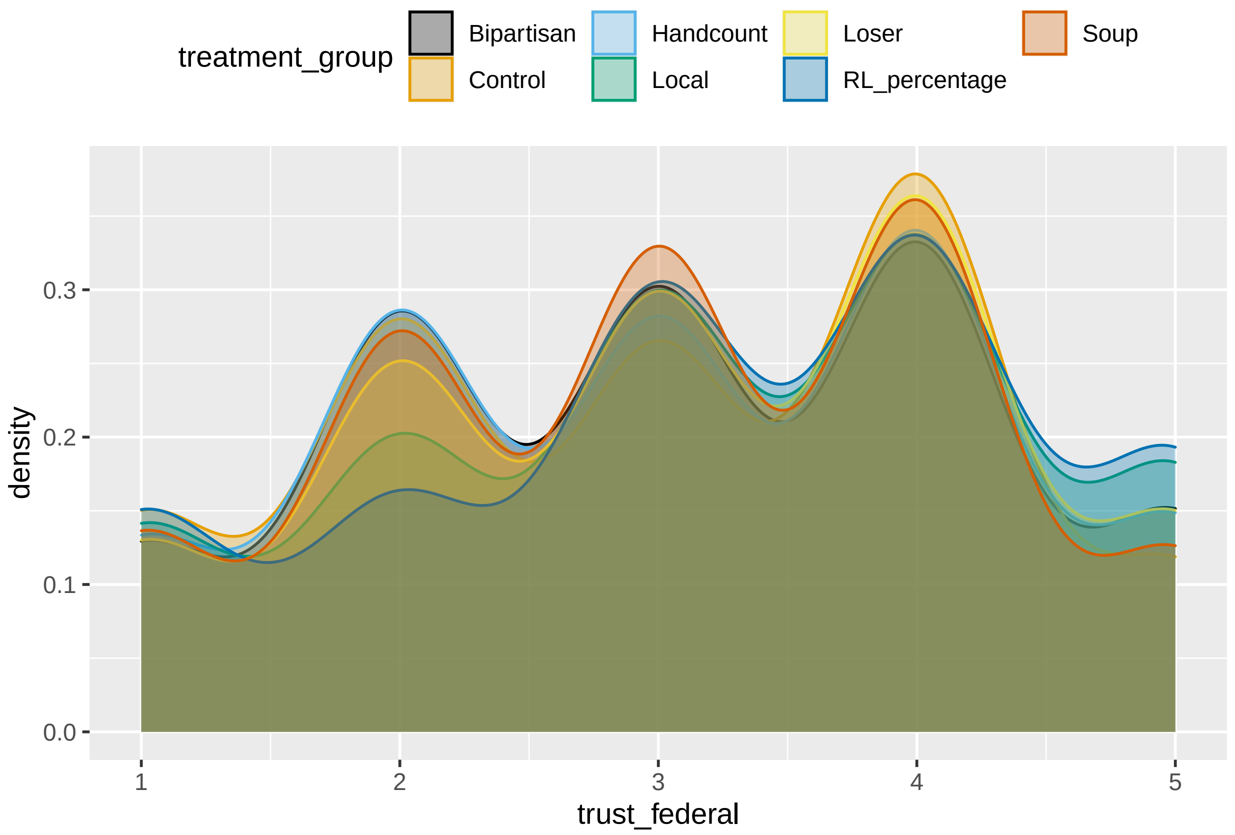
```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```

```
## Warning in max(ids, na.rm = TRUE): no non-missing arguments to max; returning
## -Inf
```



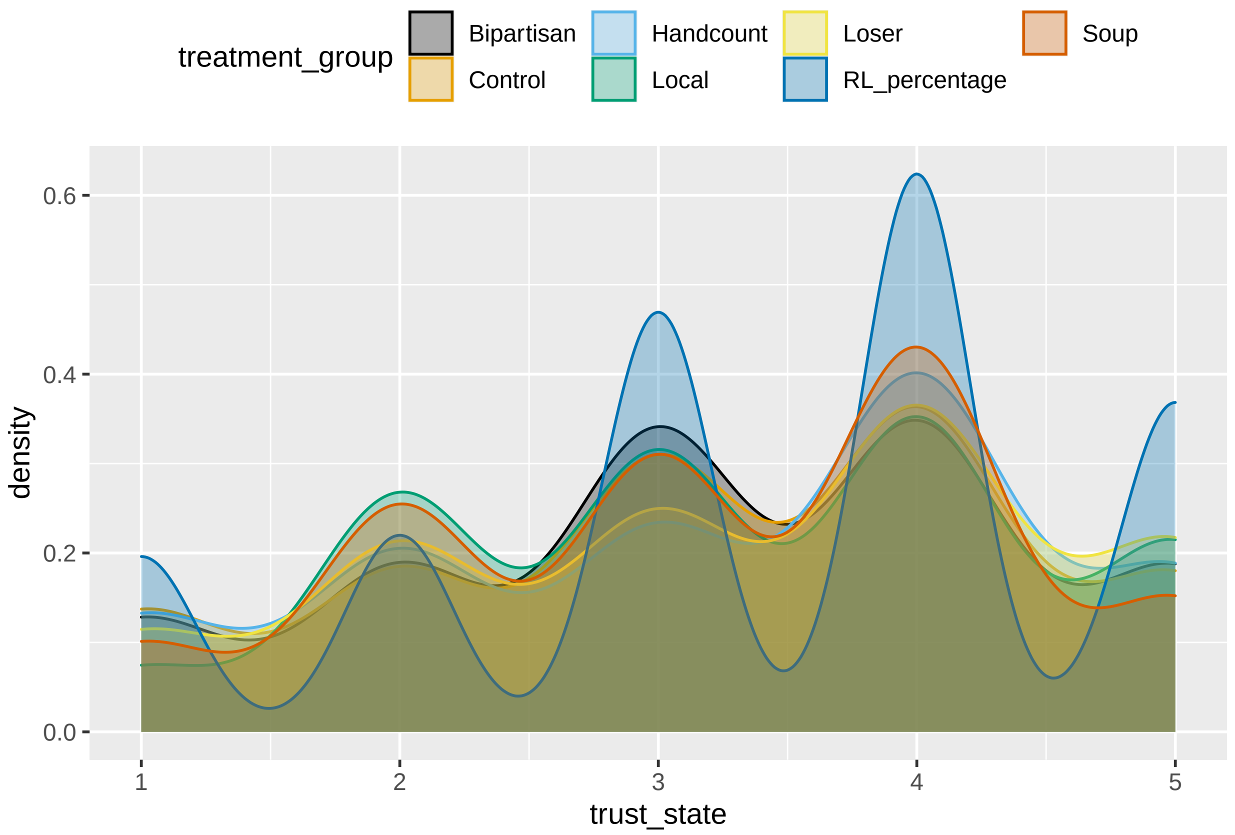
```r
# plot trust in federal government
df %>%
  ggplot(aes(x = trust_federal, # aes(x, y, ...) defines how variables are mapped into the aesthetics o
             color = treatment_group,
            fill = treatment_group
             )
         ) +
  # Density plot
  geom_density(alpha = 0.3) +     # alpha controls the transparency
  # use a colorblind friendly color palette
  scale_color_colorblind() +
  scale_fill_colorblind() +
  theme(legend.position = "top") # legend position
```

```
## Warning: Removed 10 rows containing non-finite values (stat_density).
```



```
# trust in state government
df %>%
  ggplot(aes(x = trust_state, # aes(x, y, ...) defines how variables are mapped into the aesthetics of
             color = treatment_group,
             fill = treatment_group
              )
         ) +
  # Density plot
  geom_density(alpha = 0.3) +     # alpha controls the transparency
  # use a colorblind friendly color palette
  scale_color_colorblind() +
  scale_fill_colorblind() +
  theme(legend.position = "top") # legend position
```

```
## Warning: Removed 14 rows containing non-finite values (stat_density).
```



```
# means under each condition
# ybars <- aggregate(yobs, by = list(w), mean)$x # aggregate(x, by=list, FUN) applied function to x by
# sigma <- sqrt(sum((yobs - ybars[w])^2) / (N - K)) # calculation of standard deviation
#
# # difference in means estimates
# taus <- ybars[-1] - ybars[1] # subtracting the control outcome from each of the three treatment outco
#
# # Z-stat
# Z_stat <- taus / (sigma * sqrt(2 * K / N)) # calculation of Z stat according to formula above
# Z_stat
```