## 0. Setting Up: Loading the tidyverse packages

Before working through this notebook, make sure that your runtime is set to use R (rather than python). This should be set already, but you can double-check by selecting the following from the menu at the top of the notebook.

Runtime > Change runtime type

Then make sure that R appears in the drop-down menu under runtime type.

This first part of the code just installs and loads the R packages we want to use for the following analyses.

```
#Install packages we want to use

#this loads a helper function for installing R packages
source('https://raw.githubusercontent.com/COGS119/tutorials/refs/heads/main/R/loa

#specify the name of all packages we want to use her
packages_to_apt_install = c('tidyverse')

#install packages specified above (note that we're using our old friend the for l
#For more: https://www.w3schools.com/r/r_for_loop.asp
for (package in packages_to_apt_install) load_install_package(package, apt=TRUE)

#load packages we want to use
library(tidyverse)
library(scales)

#set some basic plotting defaults
theme_set(theme_minimal(base_size = 18))

#check the version of R used
print(R.version.string)
```

Show hidden output

## 1. Load the data

In this first section, we load in the data from the experiment and inspect it.

We've already done some significant processing to your data (consult your GitHub project page for detailed code, if you're curious!). Note: If you notice that any key information is missing from the data, please check with your instructor.

```
group_name <- "massive_memory"
#read in your data
processed_data <- read_csv(paste0("https://raw.githubusercontent.com/COGS119/grou
```

```
Rows: 950 Columns: 20
── Column specification ────────────────────────────────────────────
Delimiter: ","
chr (10): participant_id, random_id, trial_phase, trial_kind, choice_images,..
dbl  (9): repeat_false_alarm_rate, repeat_hit_rate, trial_index, trial_numbe..
lgl  (1): correct

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this mess
```

Let's first take a look at your data.

```
glimpse(processed_data)
```

```
Rows: 950
Columns: 20
$ participant_id         <chr> "manatee", "manatee", "manatee", "manatee", "r
$ random_id              <chr> "p3duj2aujg2", "p3duj2aujg2", "p3duj2aujg2", '
$ repeat_false_alarm_rate <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, (
$ repeat_hit_rate        <dbl> 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, (
$ trial_index            <dbl> 349, 351, 353, 355, 357, 359, 361, 363, 365, 3
$ trial_number           <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,
$ time_elapsed           <dbl> 1372371, 1375171, 1378242, 1383217, 1386673, 1
$ trial_phase            <chr> "Test", "Test", "Test", "Test", "Test", "Test"
$ trial_kind             <chr> "exemplar", "state", "state", "state", "exempl
$ response               <dbl> 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, (
$ rt                     <dbl> 1599, 876, 1654, 2162, 1001, 1162, 934, 2929,
$ confidence_response    <dbl> 5, 5, 5, 5, 5, 5, 5, 2, 5, 5, 5, 2, 5, 5, 4, 5
$ correct                <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALS
$ is_right               <dbl> 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1
$ choice_images          <chr> "[\"stimuli\\\\EXEMPLAR\\\\USED\\\\Ewheelchai
$ correct_image          <chr> "stimuli\\EXEMPLAR\\USED\\Ewheelchair1.jpg", '
$ choice                 <chr> "stimuli\\EXEMPLAR\\USED\\Ewheelchair1.jpg", '
$ experiment_purpose     <chr> "testing subjects on their ability to recall a
$ Q1                     <chr> "shape, colour, size, and category of object",
$ technical_issues       <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA
```

```
unique(processed_data$participant_id)
```

'manatee' · 'camel' · 'panda' · 'moose' · 'flamingo' · 'gazelle' · 'trex' · 'lion' · 'giraffe' · 'hedgehog' · 'seal' · 'turtle' · 'falcon' · 'jaguar' · 'tardigrade666' · 'leopard' · 'eagle' · 'cheetah' · 'kangaroo'

```
unique(processed_data$repeat_false_alarm_rate)
```

0 · 0.00666666666666671 · 0.0133333333333333 · 0.0266666666666666 · 0.0333333333333334 · 0.02 · 0.0533333333333333

### 1.1. Codebook

A first important step is to understand your data.

Please complete this section to include a full codebook including a description of each column in your dataset. To get you started, we've included descriptions of the first few columns.

- **participant_id**: the participant code entered by the participant
- **random_id**: a random code generated for each session by the experiment (identifies unique sessions)
- **repeat_false_alarm_rate**: the average rate for participants during the n-back task who falsely pressed spacebar for the falsely recalled item during the memory task.

## 2. Descriptives

In this section, look at the basic overall descriptives, e.g., overall average responses for each condition.

If you have multiple responses per participant, make sure to first cluster/ average your data within participant before deriving an overall average estimate.

Make sure that your averages allow you to derive an estimate of your central condition difference (e.g., a condition difference).

A few other rules to keep in mind:

- typically, we look at reaction times *only* for correct responses
- if you are using a scale, make sure you are transforming any items that are reverse-coded.

```
#take a look at the number of distinct responses
processed_data |>
  distinct(random_id,participant_id)
```

A tibble: 19 × 2

| random_id | participant_id |
|---:|---:|
| <chr> | <chr> |
| p3duj2aujg2 | manatee |
| p3zegmnxubt | camel |
| p48hzab1ry2 | panda |
| p5lyc28scvl | moose |
| p6labg5etft | flamingo |
| p86x33mjzdl | gazelle |
| p920knkg1w3 | trex |
| parq0lw64nk | lion |
| pcfdh9f6su2 | giraffe |
| pe7o54mbpf0 | hedgehog |
| pg0hzpvg3xz | seal |
| phy4ybvdja7 | turtle |
| pjw2f4uz82k | falcon |
| pllo1t2f1nf | jaguar |
| pmkev4nz5vv | tardigrade666 |
| po16pvz0ydw | leopard |
| ppj3b623m4f | eagle |
| prsues5texd | cheetah |
| puwua1qwtby | kangaroo |

```
unique(processed_data$participant_id)
```

```
'manatee' · 'camel' · 'panda' · 'moose' · 'flamingo' · 'gazelle' · 'trex' · 'lion' · 'giraffe' · 'hedgehog' · 'seal' ·
'turtle' · 'falcon' · 'jaguar' · 'tardigrade666' · 'leopard' · 'eagle' · 'cheetah' · 'kangaroo'
```

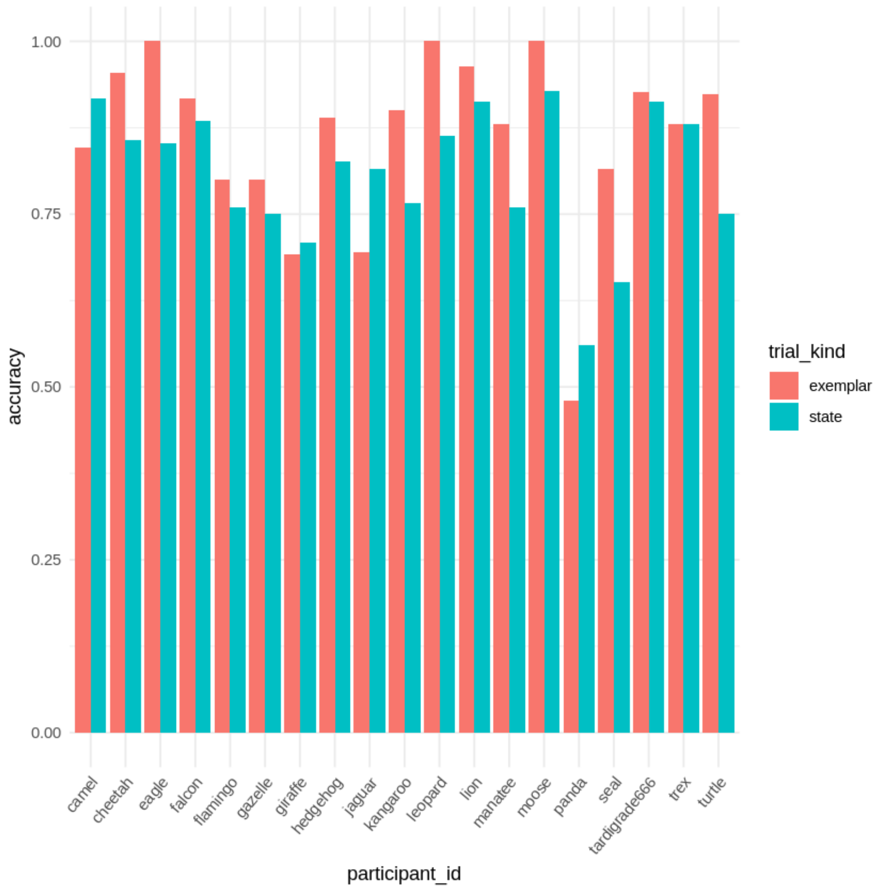
```
proc_data_stat <- processed_data |>
  group_by(participant_id, trial_kind) |>
  summarise(accuracy=mean(is_right), .groups = "drop")
    # Martin: I jumped in and made a small correction here to support! (you want
    # Note that you would actually want to compute sd/ sem only at your *next* st
proc_data_stat
```

A tibble: 38 × 3

| participant_id | trial_kind | accuracy |
|---|---|---|
| <chr> | <chr> | <dbl> |
| camel | exemplar | 0.8461538 |
| camel | state | 0.9166667 |
| cheetah | exemplar | 0.9545455 |
| cheetah | state | 0.8571429 |
| eagle | exemplar | 1.0000000 |
| eagle | state | 0.8518519 |
| falcon | exemplar | 0.9166667 |
| falcon | state | 0.8846154 |
| flamingo | exemplar | 0.8000000 |
| flamingo | state | 0.7600000 |
| gazelle | exemplar | 0.8000000 |
| gazelle | state | 0.7500000 |
| giraffe | exemplar | 0.6923077 |
| giraffe | state | 0.7083333 |
| hedgehog | exemplar | 0.8888889 |
| hedgehog | state | 0.8260870 |
| jaguar | exemplar | 0.6956522 |

```
      jaguar     state  0.8148148
    kangaroo  exemplar  0.9000000
    kangaroo     state  0.7666667
     leopard  exemplar  1.0000000
     leopard     state  0.8636364
        lion  exemplar  0.9629630
        lion     state  0.9130435
     manatee  exemplar  0.8800000
     manatee     state  0.7600000
       moose  exemplar  1.0000000
       moose     state  0.9285714
       panda  exemplar  0.4800000
       panda     state  0.5600000
        seal  exemplar  0.8148148
        seal     state  0.6521739
tardigrade666  exemplar  0.9259259
tardigrade666     state  0.9130435
        trex  exemplar  0.8800000
        trex     state  0.8800000
      turtle  exemplar  0.9230769
      turtle     state  0.7500000
```

```
ggplot(proc_data_stat, aes(x=participant_id, y=accuracy, fill=trial_kind)) +
  geom_bar(stat="identity", position=position_dodge()) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 50, hjust = 1))
```



```
summary_stat <- proc_data_stat |>
  group_by(trial_kind) |>
    summarise(
      mean_accuracy = mean(accuracy),
      sd_accuracy = sd(accuracy),
      n_obs = n(),
      sem = sd(accuracy) / sqrt(n_obs),
      ci_lower = mean_accuracy - 1.96 * sem,
      ci_upper = mean_accuracy + 1.96 * sem)
summary_stat
```

A tibble: 2 × 7

| trial_kind | mean_accuracy | sd_accuracy | n_obs | sem | ci_lower | ci_upper |
|---|---|---|---|---|---|---|
| <chr> | <dbl> | <dbl> | <int> | <dbl> | <dbl> | <dbl> |
| exemplar | 0.8611050 | 0.12954020 | 19 | 0.02971856 | 0.8028566 | 0.9193534 |
| state | 0.8082446 | 0.09889321 | 19 | 0.02268766 | 0.7637768 | 0.8527124 |

```
colnames(summary_stat) = c("Trial Kind", "Mean Accuracy", "Standard Deviation", "I
summary_stat
```

A tibble: 2 × 7

| Trial Kind | Mean Accuracy | Standard Deviation | Number of Observations | Standard Error of the Mean | Lower 95th Percentile | Upper 95th Percentile |
|---|---|---|---|---|---|---|
| <chr> | <dbl> | <dbl> | <int> | <dbl> | <dbl> | <dbl> |
| exemplar | 0.8611050 | 0.12954020 | 19 | 0.02971856 | 0.8028566 | 0.9193534 |
| state | 0.8082446 | 0.09889321 | 19 | 0.02268766 | 0.7637768 | 0.8527124 |

## 3. Plot

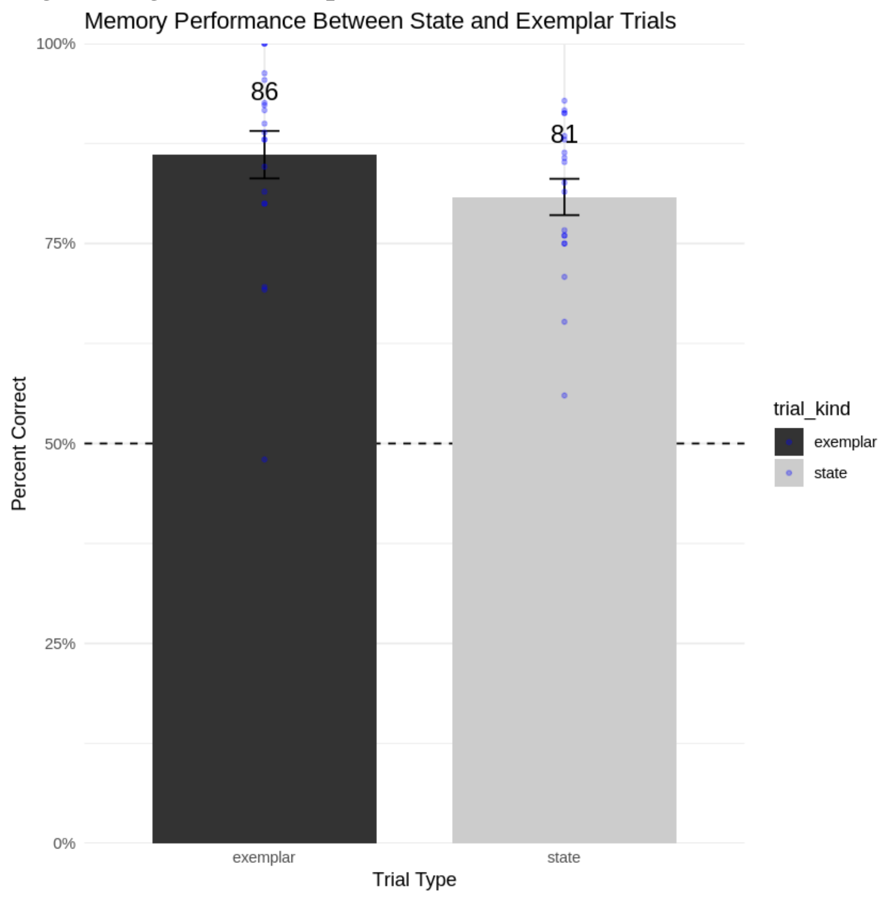
Create a plot of your central effect or effects.

Things to keep in mind:

- **clear axis labels:** make sure your axis labels are clear (don't just use the column names, try to indicate the units for dependent measures, e.g. "Reaction Time (in ms)")
- **variability across participants:** whenever possible, try to represent both the overall average and the underlying variability across participants (e.g., include dots or violin plots of individual participant averages)
- **error bars/ bands:** a good plot will also include error bars/bands for average estimates (e.g., 95% confidence intervals or standard errors)

```
mem_performance_plot <- ggplot(summary_stat, aes(x=trial_kind, y=mean_accuracy, f
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "black", size = 0.5,
  geom_bar(stat="identity", width=0.75) +
  geom_errorbar(aes(ymin=mean_accuracy - sem, ymax=mean_accuracy + sem), width=.1
                position=position_dodge(.9)) +
  geom_text(aes(label = round(mean_accuracy * 100)),
            vjust = -3, color = "black", size = 5) +
  geom_point(data = proc_data_stat, aes(x = trial_kind, y = accuracy),
            width = 0.2, size = 1, alpha = 0.3, color = "blue") +
  scale_y_continuous(labels = percent,
                     limits = c(0, 1),
                     expand = expansion(0,0)) +
  labs(title = "Memory Performance Between State and Exemplar Trials",
       x = "Trial Type",
       y = "Percent Correct") +
  scale_fill_grey() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none") +
  theme_minimal()
mem_performance_plot
```

```
Warning message in geom_point(data = proc_data_stat, aes(x = trial_kind, y = a
“Ignoring unknown parameters: `width`”
```



## Additional EDA on data

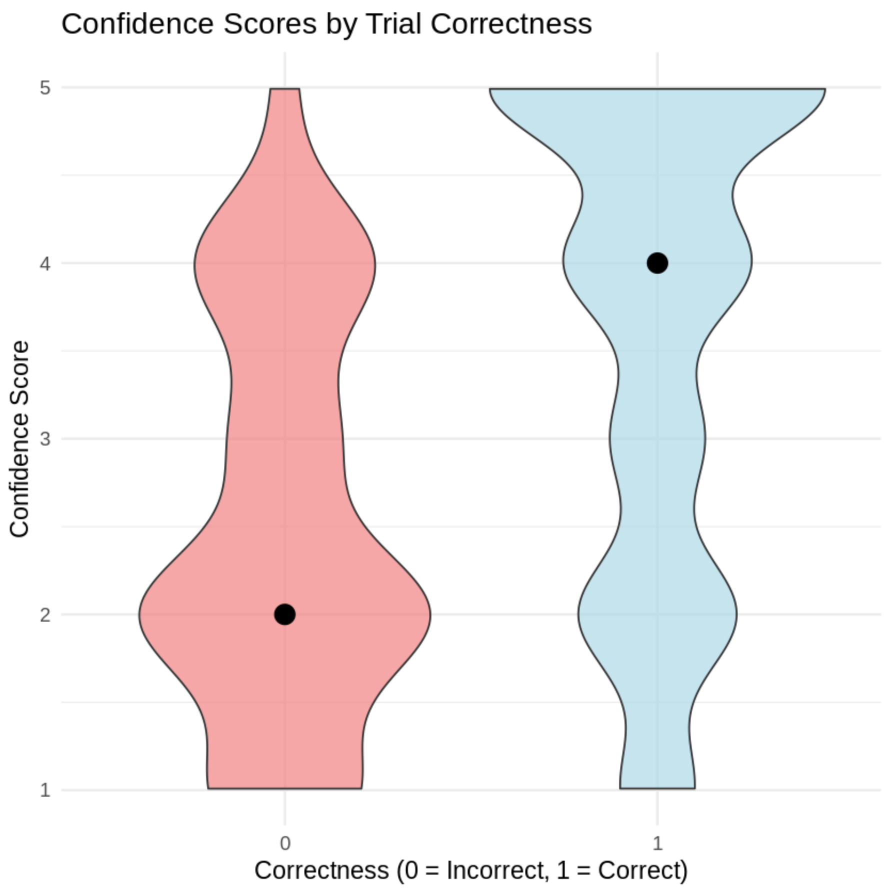
## processed_data$is_right

```
processed_data$is_right
```

```
1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 0 · 1 · 0 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 0 · 0 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 0 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 0 · 0 · 0 · 0 · 1 · 1 · 0 · 0 · 0 · 1 · 1 · 1 · 0 · 1 · 0 · 0 · 0 ·
0 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 0 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ····· 1 · 1 ·
0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 1 · 1 · 0 · 1 · 1 ·
```

```
# Confidence vs. Correct Trials
processed_data$is_right <- as.factor(processed_data$is_right)

ggplot(processed_data, aes(x = is_right, y = confidence_response, fill = is_right
  geom_violin(alpha = 0.7, trim = FALSE) +
  stat_summary(fun = function(x) quantile(x, 0.5), geom = "point", size = 5, colo
  scale_fill_manual(values = c("0" = "lightcoral", "1" = "lightblue")) +
  labs(
    title = "Confidence Scores by Trial Correctness",
    x = "Correctness (0 = Incorrect, 1 = Correct)",
    y = "Confidence Score"
  ) +
  scale_y_continuous(limits = c(1, 5), breaks = 1:5) +
  theme_minimal(base_size = 14) +
  theme(legend.position = "none")
```

```
Warning message:
“Removed 356 rows containing missing values or values outside the scale range
(`geom_violin()`).”
```



```
test_trial <- processed_data |>
  select(participant_id, trial_number, time_elapsed, trial_kind, rt, confidence_r
  group_by(participant_id) |>
  mutate(time_from_start = time_elapsed - first(time_elapsed)) |>
  ungroup()
test_trial
```

A tibble: 950 × 8

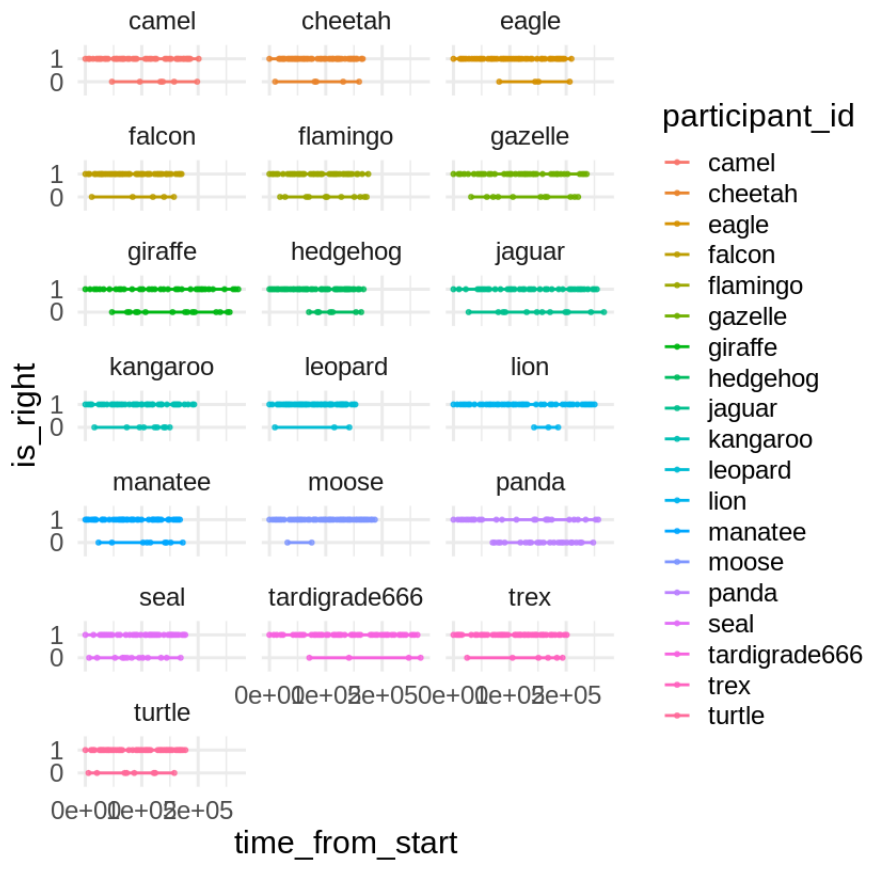
| participant_id | trial_number | time_elapsed | trial_kind | rt | confidence_resp |
|---|---|---|---|---|---|
| <chr> | <dbl> | <dbl> | <chr> | <dbl> | <c |
| manatee | 1 | 1372371 | exemplar | 1599 | |
| manatee | 2 | 1375171 | state | 876 | |
| manatee | 3 | 1378242 | state | 1654 | |
| manatee | 4 | 1383217 | state | 2162 | |
| manatee | 5 | 1386673 | exemplar | 1001 | |
| manatee | 6 | 1389233 | exemplar | 1162 | |
| manatee | 7 | 1391462 | state | 934 | |
| manatee | 8 | 1395658 | state | 2929 | |
| manatee | 9 | 1401492 | state | 1604 | |
| manatee | 10 | 1403736 | exemplar | 953 | |
| manatee | 11 | 1406860 | state | 1619 | |
| manatee | 12 | 1414776 | state | 6636 | |
| manatee | 13 | 1419485 | state | 1390 | |
| manatee | 14 | 1421806 | exemplar | 858 | |
| manatee | 15 | 1426403 | state | 3319 | |
| manatee | 16 | 1431038 | exemplar | 1289 | |
| manatee | 17 | 1433734 | state | 1442 | |
| manatee | 18 | 1436474 | state | 1493 | |
| manatee | 19 | 1438705 | state | 1049 | |
| manatee | 20 | 1440868 | exemplar | 1056 | |
| manatee | 21 | 1445093 | state | 2094 | |
| manatee | 22 | 1447709 | exemplar | 918 | |

|  |  |  |  |  |
|---|---|---|---|---|
| manatee | 23 | 1452986 | exemplar | 3947 |
| manatee | 24 | 1455524 | exemplar | 1096 |
| manatee | 25 | 1458064 | exemplar | 1236 |
| manatee | 26 | 1464073 | state | 2662 |
| manatee | 27 | 1466730 | exemplar | 927 |
| manatee | 28 | 1472157 | exemplar | 4156 |
| manatee | 29 | 1474733 | state | 1367 |
| manatee | 30 | 1481647 | state | 3677 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| kangaroo | 21 | 558824 | state | 1069 |
| kangaroo | 22 | 561859 | exemplar | 1638 |
| kangaroo | 23 | 564424 | exemplar | 975 |
| kangaroo | 24 | 567829 | exemplar | 2073 |
| kangaroo | 25 | 570438 | exemplar | 1260 |
| kangaroo | 26 | 572754 | state | 941 |
| kangaroo | 27 | 575319 | state | 1227 |
| kangaroo | 28 | 579954 | state | 1890 |
| kangaroo | 29 | 582404 | exemplar | 1035 |
| kangaroo | 30 | 585509 | state | 1532 |
| kangaroo | 31 | 589620 | exemplar | 2704 |
| kangaroo | 32 | 591859 | exemplar | 885 |
| kangaroo | 33 | 594618 | state | 1223 |
| kangaroo | 34 | 599192 | state | 1780 |
| kangaroo | 35 | 601611 | state | 909 |
| kangaroo | 36 | 605388 | state | 1476 |
| kangaroo | 37 | 607590 | state | 1035 |
| kangaroo | 38 | 610744 | state | 2008 |
| kangaroo | 39 | 613788 | state | 1810 |

| | | | | |
|---|---|---|---|---|
| kangaroo | 40 | 616431 | state | 1415 |
| kangaroo | 41 | 621452 | state | 1996 |
| kangaroo | 42 | 627883 | exemplar | 3368 |
| kangaroo | 43 | 630623 | state | 1176 |
| kangaroo | 44 | 633247 | exemplar | 874 |
| kangaroo | 45 | 635950 | state | 1465 |
| kangaroo | 46 | 638584 | exemplar | 906 |
| kangaroo | 47 | 647614 | exemplar | 3503 |
| kangaroo | 48 | 653948 | state | 4656 |
| kangaroo | 49 | 661643 | state | 3522 |
| kangaroo | 50 | 664703 | state | 1500 |

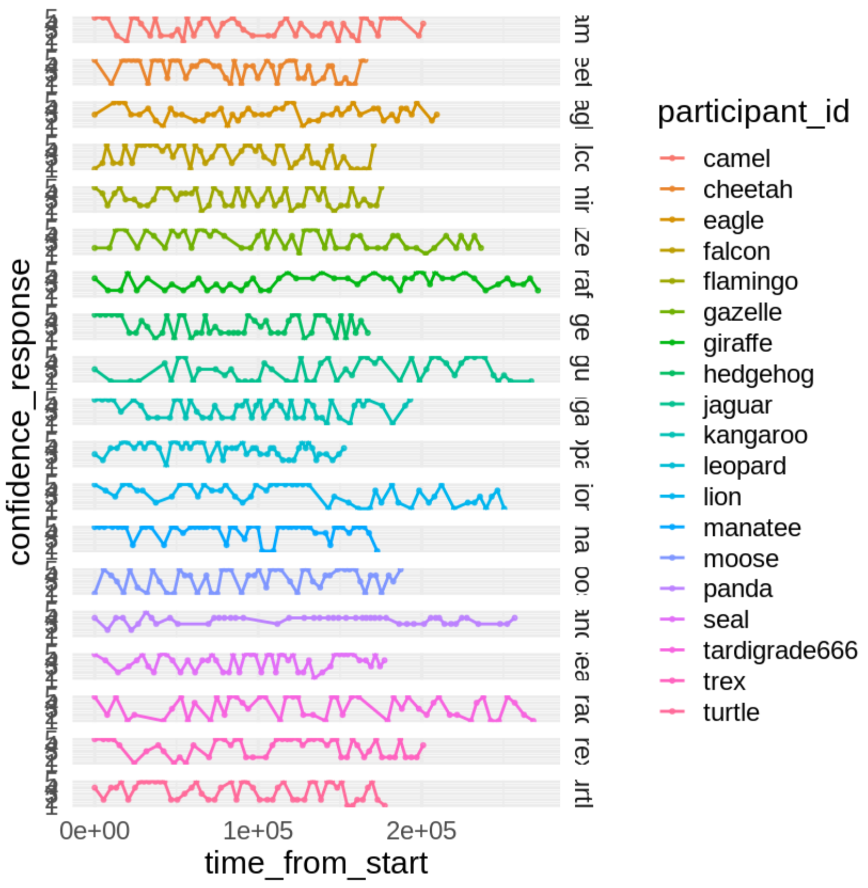
```
# Ensure participant and time elapsed are treated appropriately
test_trial$participant_id <- as.factor(test_trial$participant_id)

# Line plot of accuracy over time (time elapsed on x-axis)
ggplot(test_trial, aes(x = time_from_start, y = is_right, color = participant_id)
  geom_line(alpha = 1, size = 0.8) +
  geom_point(size = 1, alpha = 0.8) +
  facet_wrap(~participant_id, ncol = 3)
  labs(
    title = "Confidence Scores Over Time (Elapsed Time)",
    x = "Time Elapsed (seconds)",
    y = "Confidence Score",
    color = "Participant") +
  theme_minimal(base_size = 14) +
  theme(strip.text = element_text(size = 10), # Adjust size of facet labels
  panel.spacing = unit(1, "lines")) +
  theme(legend.position = "none")
```

```
NULL
```

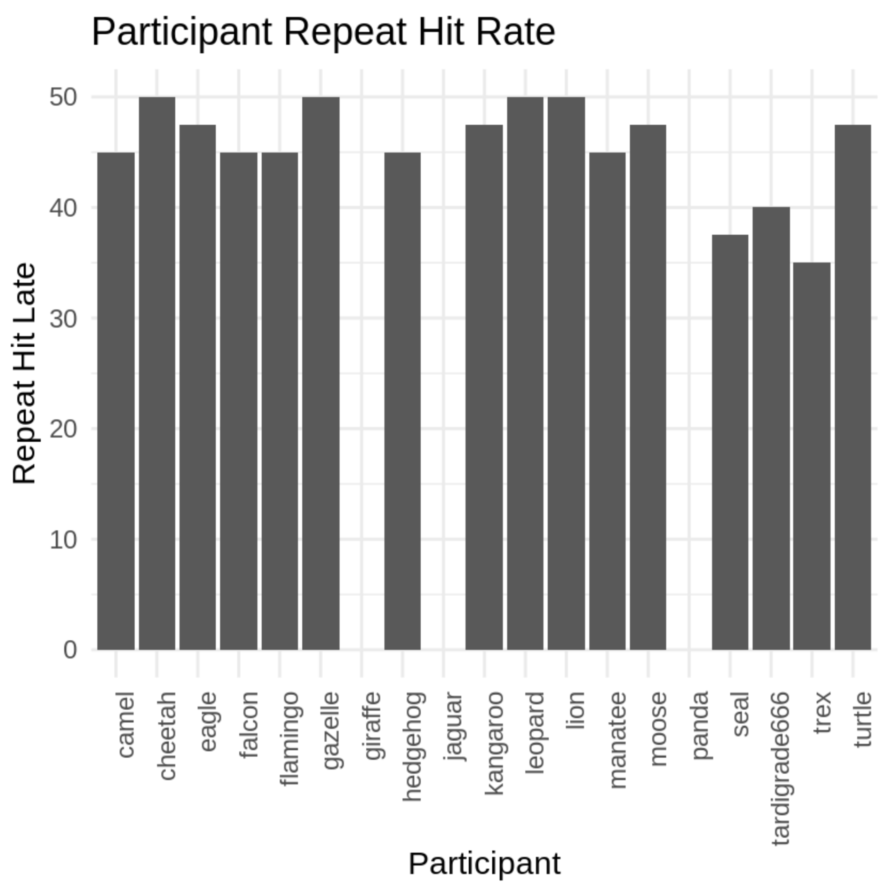


```
# Confidence Overtime between participants

ggplot(test_trial, aes(x = time_from_start, y = confidence_response, color = partic
  geom_line(alpha = 1, size = 0.8) +
  geom_point(size = 1, alpha = 0.8) +
  facet_grid(participant_id ~ .)
  labs(
    title = "Confidence Scores Over Time (Elapsed Time)",
    x = "Time Elapsed (seconds)",
    y = "Confidence Score",
    color = "Participant") +
  theme_minimal(base_size = 14) +
  theme(strip.text = element_text(size = 10), # Adjust size of facet labels
  panel.spacing = unit(1, "lines")) +
  theme(legend.position = "none")
```

```
NULL
```



```
ggplot(processed_data, aes(x = participant_id, y = repeat_hit_rate)) +
  geom_bar(stat="identity") +
  labs(title = "Participant Repeat Hit Rate",
       x = "Participant",
       y = "Repeat Hit Late") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



```
as.integer(as.logical(processed_data$repeat_hit_rate))
```

```
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 0 · 0 ·
0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 ·
0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 0 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · ···· 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 · 1 ·
```

```
as.numeric(processed_data$is_right)
```

```
2·2·2·2·2·2·2·1·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·1·1·2·1·2·2·
2·2·2·2·2·1·1·2·2·1·2·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·2·1·2·2·2·2·2·2·
2·2·2·2·2·2·1·2·2·2·2·2·2·1·1·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·1·2·2·2·
2·2·2·2·2·2·2·2·2·1·1·2·1·1·2·1·2·1·1·1·1·2·2·1·1·1·2·2·2·1·2·1·1·1·
1·1·2·1·1·2·1·2·1·1·2·1·2·2·2·2·2·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·1·2·
2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·⋯·2·2·
1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·
1·2·2·2·2·2·2·2·2·2·1·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·2·1·2·2·
2·2·2·2·2·2·2·2·2·2·2·2·2·1·2·1·2·2·2·2·2·2·2·2·2·2·2·2·1·2·2·1·2·2·
```

```
nback_data <- processed_data |>
  mutate(nback_participation = as.integer(as.logical(repeat_hit_rate)))

nback_data$is_right <- as.numeric(nback_data$is_right)

proc_nback_data <- nback_data |>
  group_by(participant_id, nback_participation) |>
  summarise(accuracy=mean(is_right)) |>
  mutate(accuracy=accuracy - 1)
proc_nback_data
```

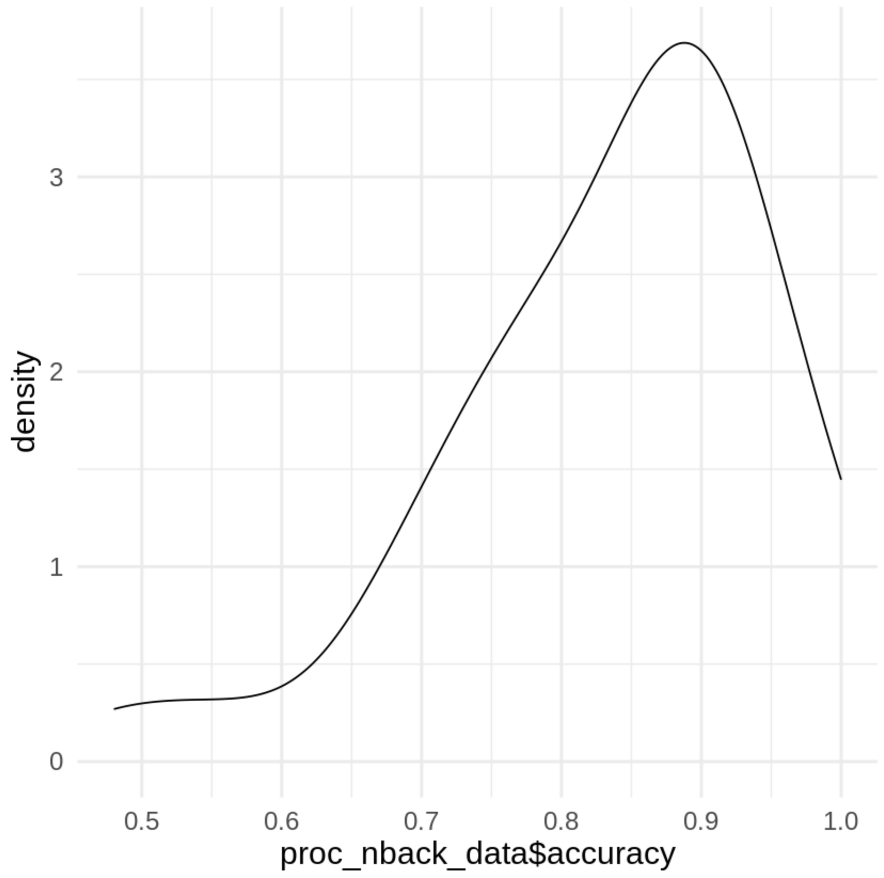
```
`summarise()` has grouped output by 'participant_id'. You can override using
the `.groups` argument.
```

A grouped_df: 19 × 3

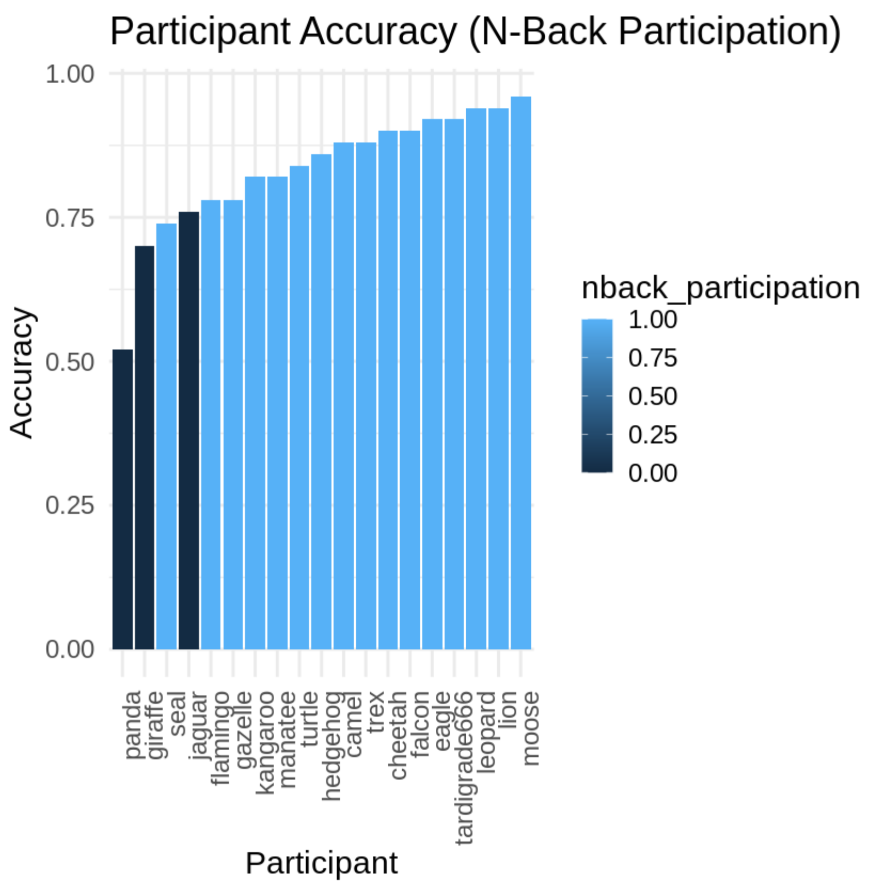
| participant_id | nback_participation | accuracy |
|---:|---:|---:|
| <chr> | <int> | <dbl> |
| camel | 1 | 0.88 |
| cheetah | 1 | 0.90 |
| eagle | 1 | 0.92 |
| falcon | 1 | 0.90 |
| flamingo | 1 | 0.78 |
| gazelle | 1 | 0.78 |
| giraffe | 0 | 0.70 |
| hedgehog | 1 | 0.86 |
| jaguar | 0 | 0.76 |
| kangaroo | 1 | 0.82 |
| leopard | 1 | 0.94 |
| lion | 1 | 0.94 |
| manatee | 1 | 0.82 |
| moose | 1 | 0.96 |
| panda | 0 | 0.52 |
| seal | 1 | 0.74 |
| tardigrade666 | 1 | 0.92 |
| trex | 1 | 0.88 |
| turtle | 1 | 0.84 |

link text

```
library(forcats)

# Reordering participants' accuracies to be ascending
proc_nback_data$participant_id <- fct_reorder(proc_nback_data$participant_id, pro


proc_nback_data$accuracy <- as.numeric(proc_nback_data$accuracy)


ggplot(proc_nback_data, aes(x=proc_nback_data$accuracy)) +
  geom_density()
```



```
library(forcats)

# Reordering participants' accuracies to be ascending
proc_nback_data$participant_id <- fct_reorder(proc_nback_data$participant_id, pro

ggplot(proc_nback_data, aes(x = participant_id, y = accuracy, fill = nback_partic
  geom_bar(stat="identity") +
  labs(title = "Participant Accuracy (N-Back Participation)",
       x = "Participant",
       y = "Accuracy") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



Panda, giraffe, and jaguar have the lowest, second lowest, and fourth lowest rated accuracies respectively.

```
updated_proc <- proc_data_stat |>
  filter(!participant_id %in% c("panda", "giraffe", "jaguar")) |>
  group_by(trial_kind) |>
  summarise(
      mean_accuracy = mean(accuracy),
      sd_accuracy = sd(accuracy),
      n_obs = n(),
      sem = sd(accuracy) / sqrt(n_obs),
      ci_lower = mean_accuracy - 1.96 * sem,
      ci_upper = mean_accuracy + 1.96 * sem)
updated_proc
```

A tibble: 2 × 7

| trial_kind | mean_accuracy | sd_accuracy | n_obs | sem | ci_lower | ci_upper |
|---:|---:|---:|---:|---:|---:|---:|
| <chr> | <dbl> | <dbl> | <int> | <dbl> | <dbl> | <dbl> |
| exemplar | 0.9058147 | 0.06780385 | 16 | 0.01695096 | 0.8725908 | 0.9390386 |
| state | 0.8295937 | 0.08045465 | 16 | 0.02011366 | 0.7901709 | 0.8690165 |

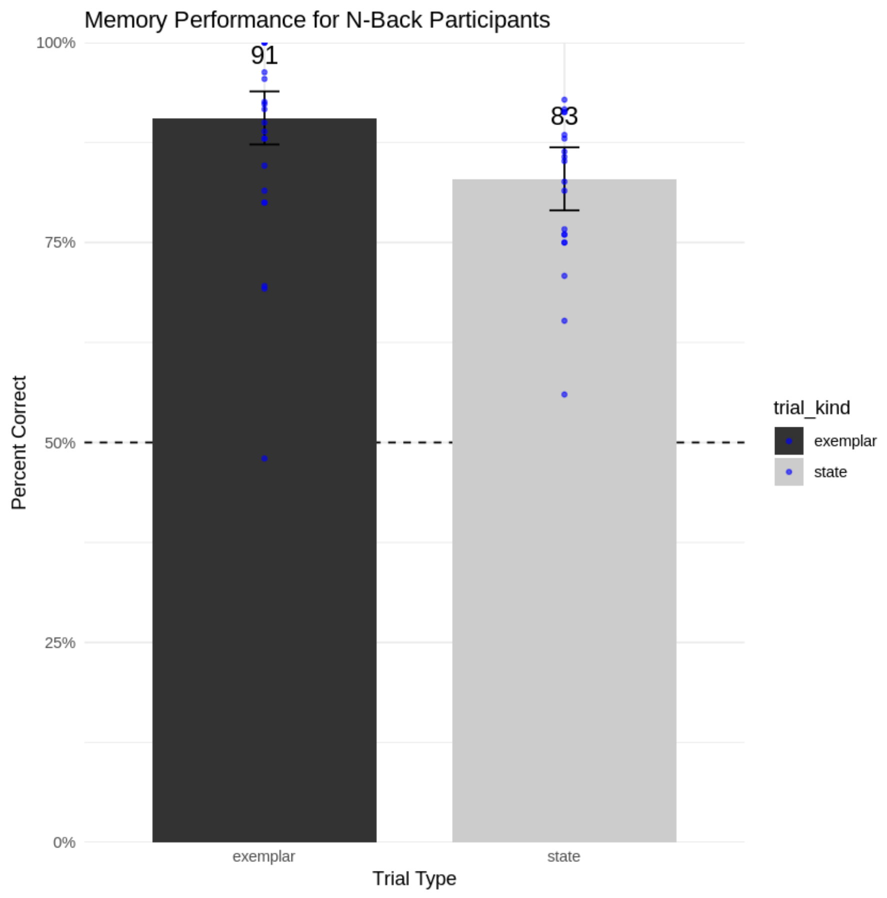
```
desc_stat <- updated_proc |>
  mutate(mean_accuracy = mean_accuracy * 100, sd_accuracy = sd_accuracy * 100, se
colnames(desc_stat) = c("Trial Kind", "Percent Accurate", "Standard Deviation", "N
desc_stat
```

A tibble: 2 × 7

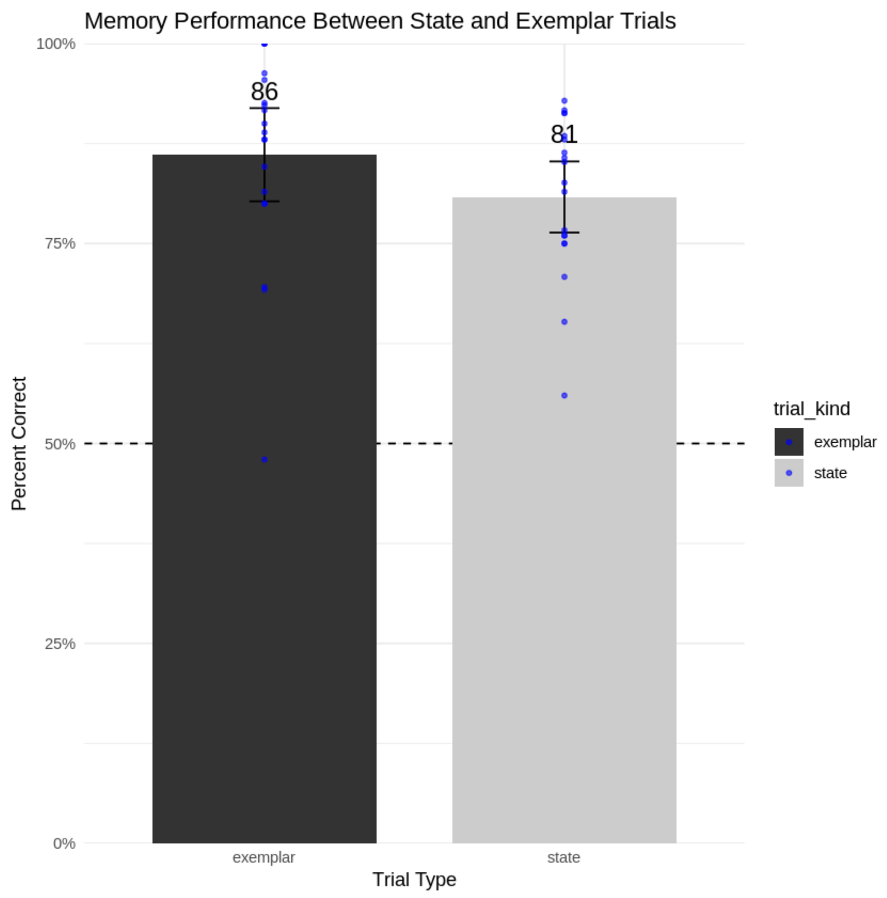
| Trial Kind | Percent Accurate | Standard Deviation | Number of Observations | Standard Error of the Mean | Lower 95th Percentile | Upper 95th Percentile |
|---:|---:|---:|---:|---:|---:|---:|
| <chr> | <dbl> | <dbl> | <int> | <dbl> | <dbl> | <dbl> |
| exemplar | 90.58147 | 6.780385 | 16 | 1.695096 | 87.25908 | 93.90386 |
| state | 82.95937 | 8.045465 | 16 | 2.011366 | 79.01709 | 86.90165 |

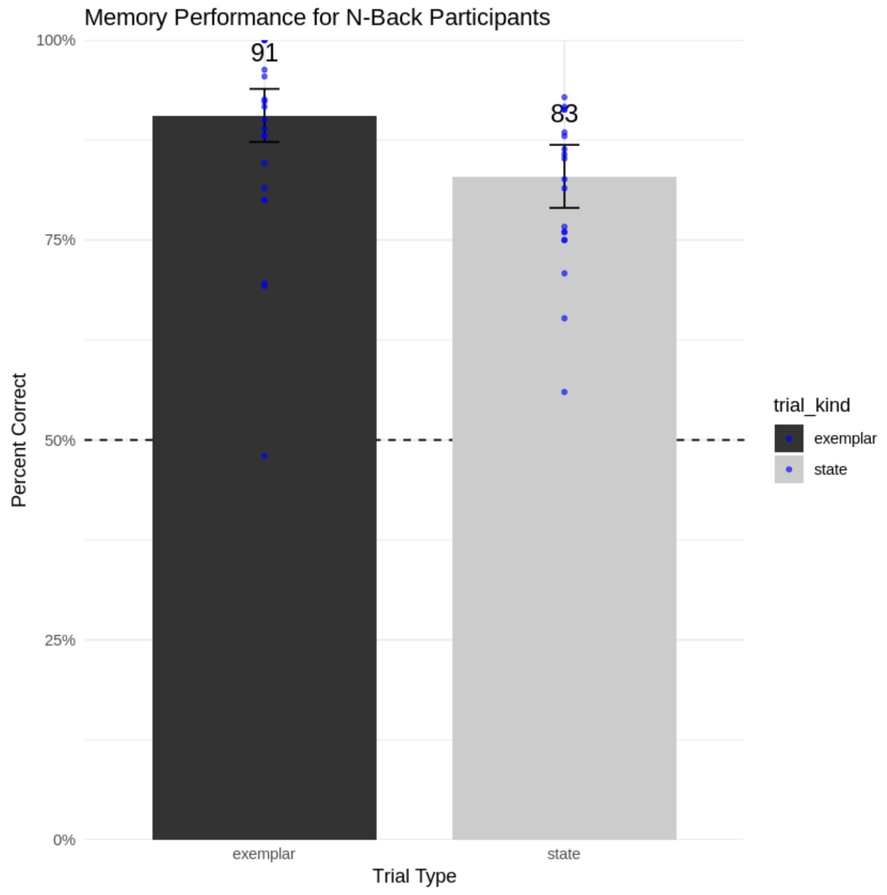
```
nback_mem_plot <- ggplot(updated_proc, aes(x=trial_kind, y=mean_accuracy, fill=tr
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "black", size = 0.5,
  geom_bar(stat="identity", width=0.75) +
  geom_errorbar(aes(ymin=ci_lower, ymax=ci_upper), width=.1,
                position=position_dodge(.9)) +
  geom_text(aes(label = round(mean_accuracy * 100)),
            vjust = -3, color = "black", size = 5) +
  geom_point(data = proc_data_stat, aes(x = trial_kind, y = accuracy),
             width = 0.2, size = 1, alpha = 0.6, color = "blue") +
  scale_y_continuous(labels = percent,
                     limits = c(0, 1),
                     expand = expansion(0,0)) +
  labs(title = "Memory Performance for N-Back Participants",
       x = "Trial Type",
       y = "Percent Correct") +
  scale_fill_grey() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none") +
  theme_minimal()
nback_mem_plot
```

```
Warning message in geom_point(data = proc_data_stat, aes(x = trial_kind, y = a
“Ignoring unknown parameters: `width`”
```



```
par(mfrow = c(1, 2))

mem_performance_plot

nback_mem_plot
```


Memory Performance Between State and Exemplar Trials
100%
75%
50%
25%
0%
Percent Correct
86
81
exemplar
state
Trial Type
trial_kind
exemplar
state

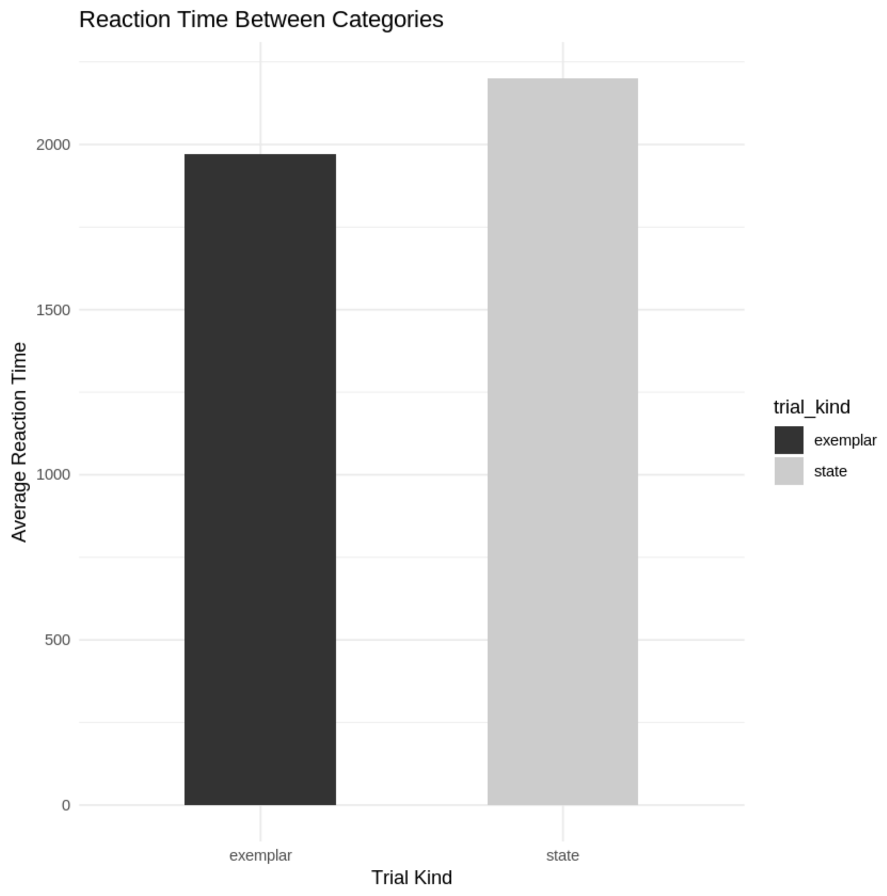

Memory Performance for N-Back Participants
100%
75%
50%
25%
0%
Percent Correct
91
83
exemplar
state
Trial Type
trial_kind
exemplar
state

```
# Reaction Time Comparison State vs. Exemplar
rt_test_trial <- test_trial|>
  group_by(trial_kind) |>
  summarise(rt_acc = mean(rt))
rt_test_trial
```

A tibble: 2 × 2

| trial_kind | rt_acc |
|---|---|
| <chr> | <dbl> |
| exemplar | 1972.226 |
| state | 2200.095 |

```
ggplot(rt_test_trial, aes(x=trial_kind, y=rt_acc, fill=trial_kind)) +
  geom_bar(stat="identity", width=0.5) +
  labs(title = "Reaction Time Between Categories",
       x = "Trial Kind",
       y = "Average Reaction Time") +
  scale_fill_grey() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "none") +
  theme_minimal()
```



## 4. Inference [optional]

Can you derive statistical tests or models that investigate your central question of interest? A good starting point will be to look at the Results section of your replication article.

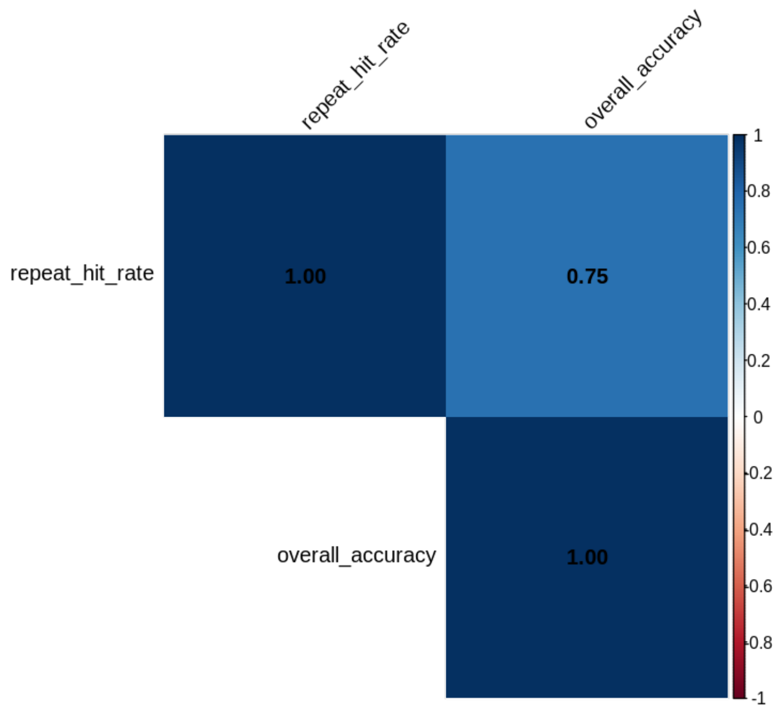
```
install.packages("corrplot")
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

```
#participant-level accuracy
participant_mean_accuracy <- proc_data_stat %>%
  group_by(participant_id) %>%
  summarise(overall_accuracy = mean(accuracy, na.rm = TRUE))

#`repeat_hit_rate` in `processed_data`?
if (!"repeat_hit_rate" %in% colnames(processed_data)) {
  processed_data <- processed_data %>%
    mutate(repeat_hit_rate = repeat_correct / (repeat_correct + repeat_false_aları
}

merged_data <- merge(participant_mean_accuracy, processed_data, by = "participant_
str(merged_data)
```

```
'data.frame':	950 obs. of  21 variables:
 $ participant_id          : Factor w/ 19 levels "panda","giraffe",..: 12 12 1:
 $ overall_accuracy        : num  0.881 0.881 0.881 0.881 0.881 ...
 $ random_id               : chr  "p3zegmnxubt" "p3zegmnxubt" "p3zegmnxubt" "p:
 $ repeat_false_alarm_rate: num  0.00667 0.00667 0.00667 0.00667 0.00667 ...
 $ repeat_hit_rate         : num  0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 ...
 $ trial_index             : num  389 405 391 397 387 411 385 395 367 413 ...
 $ trial_number            : num  21 29 22 25 20 32 19 24 10 33 ...
 $ time_elapsed            : num  547158 586079 553399 565492 543669 ...
 $ trial_phase             : chr  "Test" "Test" "Test" "Test" ...
 $ trial_kind              : chr  "exemplar" "exemplar" "exemplar" "exemplar" .
 $ response                : num  1 0 1 1 1 1 0 1 0 0 ...
 $ rt                      : num  1970 842 1562 2165 3437 ...
 $ confidence_response     : num  4 5 5 2 2 2 5 3 5 2 ...
 $ correct                 : logi  TRUE TRUE TRUE FALSE TRUE FALSE ...
 $ is_right                : Factor w/ 2 levels "0","1": 2 2 2 1 2 1 2 2 2 1 ..
 $ choice_images           : chr  "[\"stimuli\\\\EXEMPLAR\\\\USED\\\\Econe2.jp
 $ correct_image           : chr  "stimuli\\EXEMPLAR\\USED\\Econe1.jpg" "stimu
 $ choice                  : chr  "stimuli\\EXEMPLAR\\USED\\Econe1.jpg" "stimu
 $ experiment_purpose      : chr  "I think items you saw repeated you would pr
 $ Q1                      : chr  "Certain positions, certain facets pop out (:
 $ technical_issues        : chr  "No." "No." "No." "No." ...
```

```
#correlation
cor_data <- merged_data %>%
  select(repeat_hit_rate, overall_accuracy)

cor_matrix <- cor(cor_data, use = "complete.obs")

if (!requireNamespace("corrplot", quietly = TRUE)) {
  install.packages("corrplot")
}
library(corrplot)
corrplot(cor_matrix, method = "shade", type = "upper", tl.col = "black", tl.srt = 4
```



```
#numeric `is_right`
processed_data$is_right <- as.numeric(as.character(processed_data$is_right))

processed_data <- processed_data %>%
  filter(!is.na(is_right), !is.na(trial_kind)) %>%
  mutate(is_right = as.numeric(as.character(is_right)),
         trial_kind = as.factor(trial_kind))
```

```
condition_accuracy <- processed_data %>%
  group_by(trial_kind) %>%
  summarise(mean_accuracy = mean(is_right, na.rm = TRUE), .groups = "drop")
condition_accuracy
```

A tibble: 2 × 2

| trial_kind | mean_accuracy |
|---|---|
| <fct> | <dbl> |
| exemplar | 0.8598326 |
| state | 0.8093220 |

```
#paired t-test: State vs. Exemplar
state_data <- processed_data$is_right[processed_data$trial_kind == "State"]
exemplar_data <- processed_data$is_right[processed_data$trial_kind == "Exemplar"]

if (length(state_data) > 1 & length(exemplar_data) > 1) {
  state_exemplar_ttest <- t.test(state_data, exemplar_data, paired = TRUE)
  print(state_exemplar_ttest)
} else {
  message("Not enough observations for paired t-test.")
}
```

```
Not enough observations for paired t-test.
```

```
length(state_data)
length(exemplar_data)
```

```
0
0
```

```
str(processed_data) #structure
```

```
tibble [950 × 20] (S3: tbl_df/tbl/data.frame)
 $ participant_id         : chr [1:950] "manatee" "manatee" "manatee" "manatee
 $ random_id              : chr [1:950] "p3duj2aujg2" "p3duj2aujg2" "p3duj2au
 $ repeat_false_alarm_rate: num [1:950] 0 0 0 0 0 0 0 0 0 0 ...
 $ repeat_hit_rate        : num [1:950] 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.9 0.
 $ trial_index            : num [1:950] 349 351 353 355 357 359 361 363 365 36
 $ trial_number           : num [1:950] 1 2 3 4 5 6 7 8 9 10 ...
 $ time_elapsed           : num [1:950] 1372371 1375171 1378242 1383217 13866
 $ trial_phase            : chr [1:950] "Test" "Test" "Test" "Test" ...
 $ trial_kind             : Factor w/ 2 levels "exemplar","state": 1 2 2 2 1 1
 $ response               : num [1:950] 1 0 1 0 0 0 0 0 1 0 ...
 $ rt                     : num [1:950] 1599 876 1654 2162 1001 ...
 $ confidence_response    : num [1:950] 5 5 5 5 5 5 5 2 5 5 ...
 $ correct                : logi [1:950] TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ is_right               : num [1:950] 1 1 1 1 1 1 1 0 1 1 ...
 $ choice_images          : chr [1:950] "[\"stimuli\\\\EXEMPLAR\\\\USED\\\\Ewh
 $ correct_image          : chr [1:950] "stimuli\\EXEMPLAR\\USED\\Ewheelchair
 $ choice                 : chr [1:950] "stimuli\\EXEMPLAR\\USED\\Ewheelchair
 $ experiment_purpose     : chr [1:950] "testing subjects on their ability to
 $ Q1                     : chr [1:950] "shape, colour, size, and category of
 $ technical_issues       : chr [1:950] NA NA NA NA ...
```

```
head(processed_data) #sample rows
```

| participant_id | random_id | repeat_false_alarm_rate | repeat_hit_rate | trial_ind |
|---|---|---|---|---|
| <chr> | <chr> | <dbl> | <dbl> | <db |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |
| manatee | p3duj2aujg2 | 0 | 0.9 | 3 |

```
summary(processed_data) #summarizing key columns
```

```
 participant_id      random_id         repeat_false_alarm_rate
repeat_hit_rate
 Length:950         Length:950         Min.   :0.000000        Min.   :0.0000
 Class :character   Class :character   1st Qu.:0.000000        1st Qu.:0.7500
 Mode  :character   Mode  :character   Median :0.006667        Median :0.9000
                                       Mean   :0.011930        Mean   :0.7658
                                       3rd Qu.:0.020000        3rd Qu.:0.9500
                                       Max.   :0.053333        Max.   :1.0000
  trial_index    trial_number    time_elapsed      trial_phase
 Min.   :349   Min.   : 1.0   Min.   : 457741   Length:950
 1st Qu.:373   1st Qu.:13.0   1st Qu.: 556686   Class :character
 Median :398   Median :25.5   Median : 615136   Mode  :character
 Mean   :398   Mean   :25.5   Mean   : 849481
 3rd Qu.:423   3rd Qu.:38.0   3rd Qu.: 691381
 Max.   :447   Max.   :50.0   Max.   :3746911
    trial_kind      response            rt          confidence_response
 exemplar:478   Min.   :0.0000   Min.   :  724   Min.   :1.000
 state   :472   1st Qu.:0.0000   1st Qu.: 1248   1st Qu.:2.000
                Median :0.0000   Median : 1634   Median :4.000
                Mean   :0.4547   Mean   : 2085   Mean   :3.466
                3rd Qu.:1.0000   3rd Qu.: 2364   3rd Qu.:5.000
                Max.   :1.0000   Max.   :19244   Max.   :5.000
  correct           is_right      choice_images      correct_image
 Mode :logical   Min.   :0.0000   Length:950         Length:950
 FALSE:157       1st Qu.:1.0000   Class :character   Class :character
 TRUE :793       Median :1.0000   Mode  :character   Mode  :character
                 Mean   :0.8347
                 3rd Qu.:1.0000
                 Max.   :1.0000
    choice          experiment_purpose      Q1            technical_issues
 Length:950         Length:950         Length:950         Length:950
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character
```

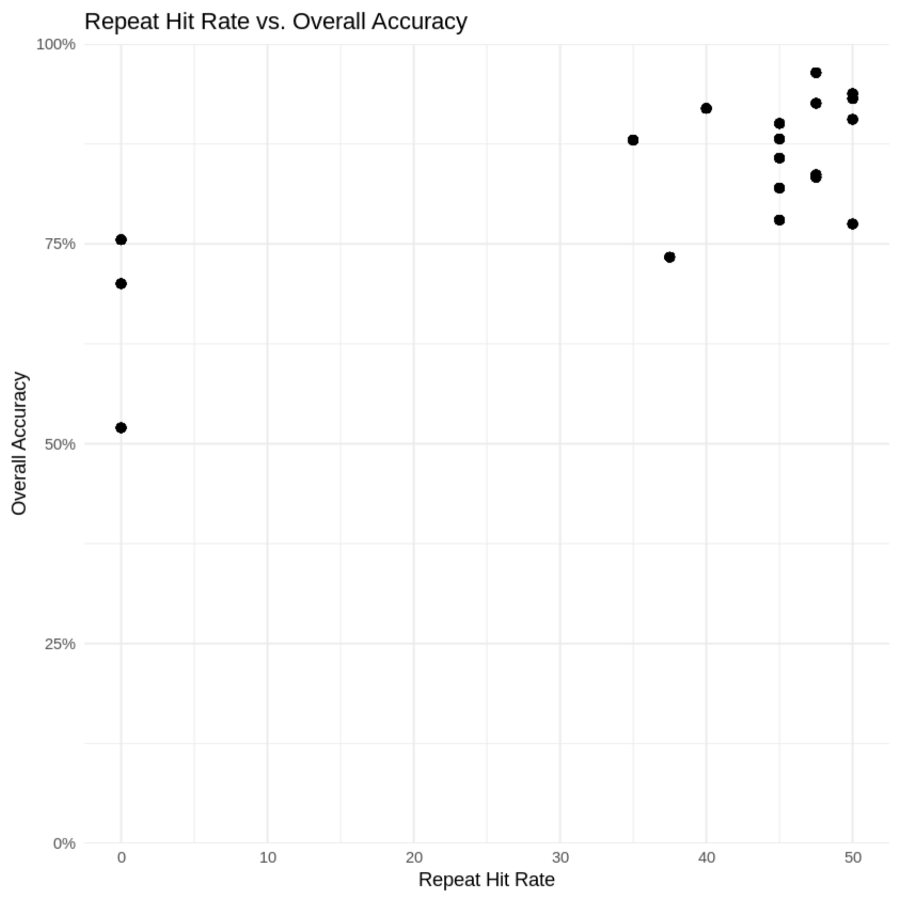
```
table(processed_data$trial_kind)

#switching to unpaired t-test
if (length(state_data) > 1 & length(exemplar_data) > 1) {
  state_exemplar_ttest <- t.test(state_data, exemplar_data, paired = FALSE)
  print(state_exemplar_ttest)
} else {
  message("Not enough observations for t-test (paired or unpaired).")
}
```

```
exemplar    state
     478      472
Not enough observations for t-test (paired or unpaired).
```

```
#ANOVA: Effect of Trial Kind on Accuracy
processed_data <- processed_data %>%
  group_by(participant_id, trial_kind) %>%
  summarise(accuracy = mean(is_right, na.rm = TRUE), .groups = "drop")

anova_results <- aov(accuracy ~ trial_kind, data = processed_data)
summary(anova_results)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
trial_kind   1 0.0265 0.02654   1.999  0.166
Residuals   36 0.4781 0.01328
```

```
# Ensure merged_data is created
ggplot(merged_data, aes(x = repeat_hit_rate, y = overall_accuracy)) +
  geom_point(color = "black", size = 2, alpha = 0.6) +
  scale_y_continuous(labels = percent,
                     limits = c(0, 1),
                     expand = expansion(0,0)) +
  labs(title = "Repeat Hit Rate vs. Overall Accuracy",
       x = "Repeat Hit Rate", y = "Overall Accuracy") +
  theme_minimal()
```



Start coding or generate with AI.