Course Project

Calvin Dorsey

2025-03-12

```r
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.4.2

## Warning: package 'ggplot2' was built under R version 4.4.2

## Warning: package 'tibble' was built under R version 4.4.2

## Warning: package 'tidyr' was built under R version 4.4.2

## Warning: package 'readr' was built under R version 4.4.2

## Warning: package 'purrr' was built under R version 4.4.2

## Warning: package 'dplyr' was built under R version 4.4.2

## Warning: package 'forcats' was built under R version 4.4.2

## Warning: package 'lubridate' was built under R version 4.4.2

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- 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 become error
```

```r
library(dplyr)
library(knitr)
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.4.3

## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Abstract

In this study, neural activity of mice was investigated during a series of decision-making tasks based their response to visual stimuli. Correct and incorrect choices were met with correspondingly positive or negative feedback based on the stimuli presented. Stimuli consisted of differing contrasts to the right and left of the mouse. Neuropixels probes were used to record approximately 30,000 neurons in 42 different brain regions across 18 different test sessions. Activation in individual brain areas proved to be inconsistent when presented with differing levels of stimuli presented along making the associated response hard to determine based solely on this. Instead, neural activity as a whole was used to determine their decision making along with difference in left and right visual contrast was examined to better determine the success rate of their decision. Both total neural activity and contrast difference proved to be effective in predicting positive feedback.

Introduction

This report, based on a study conducted by Steinmetz et al. (2019), was conducted to examine decision-making through the neural mechanisms and sensory information in order to find a predictive model for determining the positive and negative feedback associated with each decision. The data originally recorded in the study was done on 4 separate mice across 18 different sessions. Number of trials, number of neurons and number of brain areas being tested varied from session to session. Mice in each of these trials were presented with a visual stimuli consisting of two screens to the left and right. Each screen would display 1 of 4 contrast levels ranging from 0 to 1. After viewing the stimuli, the mouse would turn a wheel based on said stimulus. This decision would be met with either positive or negative feedback. Feedback was determined by the following criteria:

- When left contrast > right contrast, success (1) if turning the wheel to the right and failure (-1) otherwise.
- When right contrast > left contrast, success (1) if turning the wheel to the left and failure (-1) otherwise.
- When both left and right contrasts are zero, success (1) if holding the wheel still and failure (-1) otherwise.
- When left and right contrasts are equal but non-zero, left or right will be randomly chosen (50%) as the correct choice. In order to create and present an effective model for predicting this outcome, the report is divided into 5 different sections, exploratory data analysis, data integration, predictive modeling, predictive performance testing and discussion.

```r
session=list()
for(i in 1:18){
  session[[i]]=readRDS(paste('C:/Users/calvin/Desktop/STA141AProject/Data/session',i,'.rds',sep=''))
}
```

Exploratory Data Analysis

When looking at the data as a whole across all sessions, some important information we can extract include the name of the mouse in each trial, the date each session was conducted, the number of brain areas examined, the number of neurons examined and the success rate in each session.

```r
n.session = length(session)

# creates empty tibble with placeholders
```

```r
meta = tibble(mouse_name = rep('name',n.session),
  date_exp = rep('dt',n.session),
  n_brain_area = rep(0,n.session),
  n_neurons = rep(0,n.session),
  n_trials = rep(0,n.session),
  success_rate = rep(0,n.session))

# fills table with corresponding extracted data
for(i in 1:n.session){
  curr = session[[i]];
  meta[i,1] = curr$mouse_name;
  meta[i,2] = curr$date_exp;
  meta[i,3] = length(unique(curr$brain_area));
  meta[i,4] = dim(curr$spks[[1]])[1];
  meta[i,5] = length(curr$feedback_type);
  meta[i,6] = length(which(curr$feedback_type>0))/length(curr$feedback_type)
}

kable(meta, format = "html", table.attr = "class='table table-striped'",digits=2)
```

mouse_name

date_exp

n_brain_area

n_neurons

n_trials

success_rate

Cori

2016-12-14

8

734

114

0.61

Cori

2016-12-17

5

1070

251

0.63

Cori

2016-12-18

11

619

228

0.66

Forssmann

2017-11-01

11

1769

249

0.67

Forssmann

2017-11-02

10

1077

254

0.66

Forssmann

2017-11-04

5

1169

290

0.74

Forssmann

2017-11-05

8

584

252

0.67

Hench

2017-06-15

15

1157

250

0.64

Hench

2017-06-16

12

788

372

0.69

Hench

2017-06-17

13

1172

447

0.62

Hench

2017-06-18

6

857

342

0.80

Lederberg

2017-12-05

12

698

340

0.74

Lederberg

2017-12-06

15

983

300

0.80

Lederberg

2017-12-07

10

756

268

0.69

Lederberg

2017-12-08

8

743

404

0.76

Lederberg

2017-12-09

6

474

280

0.72

Lederberg

2017-12-10

6

565

224

0.83

Lederberg

2017-12-11

10

1090

216

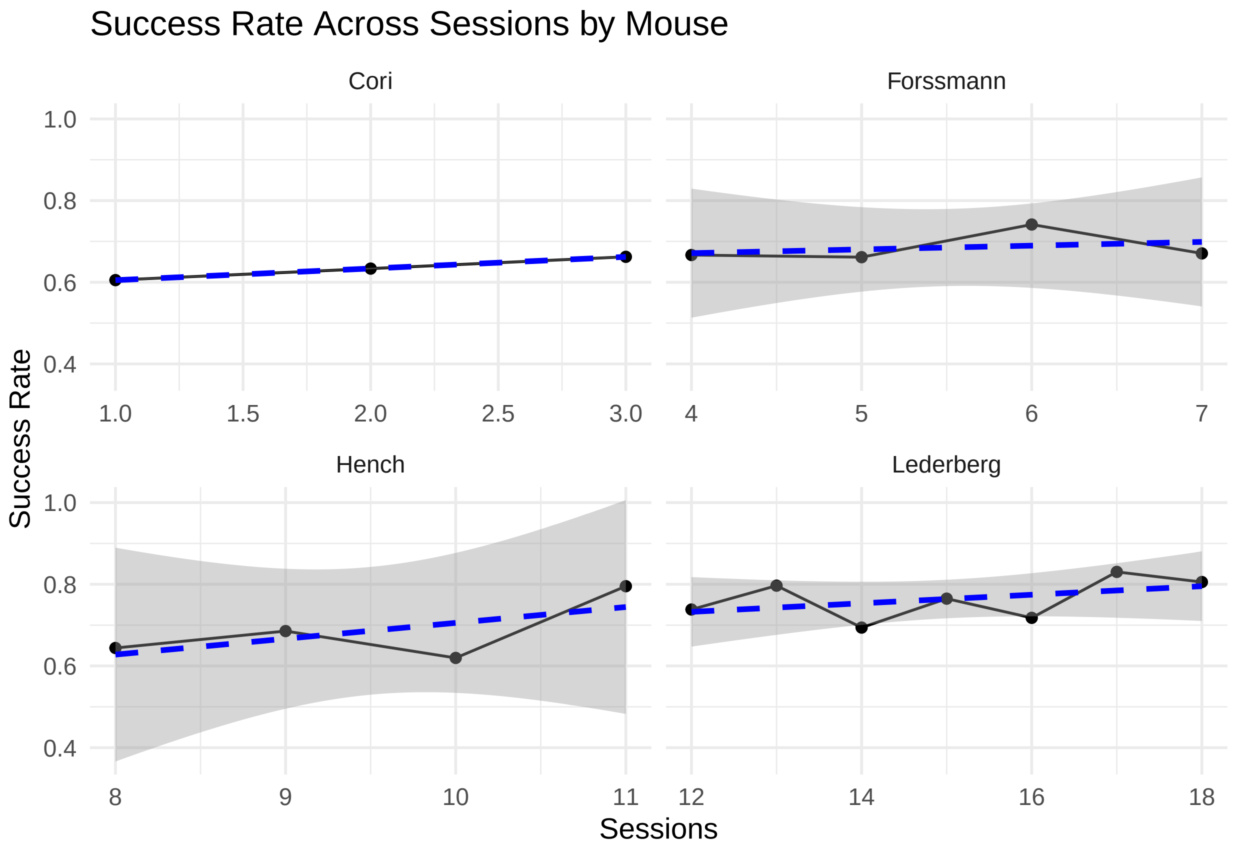
0.81

This table is helpful in allowing us to understand the large amount of variability from session to session of variables including brain areas, number of neurons, number of trials and success rate. This would imply that looking into specific attributes of the data such as specific neurons or brain areas will be hard to use in order to find a predictive model for feedback outcome. General trends and patterns across data will have to be examined and utilized instead. To view how the mice have adapted to the testing from session to session we can use a plot of the changes of success rates for each mouse in each session.

```
#filtering data from meta into each mouse
cori = meta %>% filter(mouse_name == 'Cori')
forssmann = meta %>% filter(mouse_name == 'Forssmann')
hench = meta %>% filter(mouse_name == 'Hench')
lederberg = meta %>% filter(mouse_name == 'Lederberg')
mouse_names = list(cori, forssmann, hench, lederberg)

#creates a plot of trends in success rate for each mouse
for (i in mouse_names) {
  plot = ggplot(i, aes(x = 1:nrow(i), y = success_rate)) +
  geom_point() +
  geom_line() +
  geom_smooth(method = 'lm', color = 'blue', linetype = 'dashed') +
  labs(x = 'Sessions', y = 'Success Rate', title = i$mouse_name)
}

#creates a visualization containing all four plots side by side for later use
```

```
combined_mice = bind_rows(cori, forssmann, hench, lederberg)

grouped_plot = ggplot(combined_mice, aes(x = 1:nrow(combined_mice), y = success_rate)) +
  geom_point() +
  geom_line() +
  geom_smooth(method = 'lm', color = 'blue', linetype = 'dashed') +
  labs(x = 'Sessions', y = 'Success Rate', title = 'Success Rate Across Sessions by Mouse') +
  facet_wrap(~ mouse_name, scales = "free_x") +
  theme_minimal()

grouped_plot
```

```
## 'geom_smooth()' using formula = 'y ~ x'
```



We can see from the success rate plots the extent to which experience in previous sessions impacts success rate positively for each mouse.

Next we'd like to analyze feedback results and neural data in an individual session. We can first start by looking at the average neural spikes for each area of the brain in session 2 trial 1.

```
# Values assigned for a single session and trial to be examined
s = 2
t = 1

spk.trial = session[[s]]$spks[[t]]
```

```
area=session[[s]]$brain_area
spk.count=apply(spk.trial,1,sum)

# function to calculate average spikes per brain area given a specific session and trial
spk.avg = function(s, t){
  spk.trial = session[[s]]$spks[[t]]
  area = session[[s]]$brain_area

  for(i in 1:dim(spk.trial)[1]){
    spk.count[i]=sum(spk.trial[i,])
  }

  return(tapply(spk.count, area, mean))
}

#generates spike averages for the current session and trial
spk.avg(s,t)
```

```
##      CA1     POST     root     VISl    VISpm
## 1.121053 1.816754 1.538462 1.398268 2.000000
```

We can see from average spikes in the first trial that spikes across the 5 brain areas included are similar ranging from one to two spikes. A useful way to use this data in other areas can be done through making a matrix for each individual trial consisting of average neural spikes for each brain area along with the corresponding feedback, left contrast and right contrast.

```
# creates values for the number of brain areas and trials for the given session
n.trial = length(session[[s]]$feedback_type)
n.area = length(unique(session[[s]]$brain_area))

# creates a matrix for the average spike counts for every trial in a session
trial.summary = matrix(nrow=n.trial,ncol= n.area+1+2+1)
for(t in 1:n.trial){
  trial.summary[t,] = c(spk.avg(s, t), session[[s]]$feedback_type[t], session[[s]]$contrast_left[t], ses
}

# assigns column names
colnames(trial.summary) = c(names(spk.avg(s, t)), 'feedback', 'left_contr','right_contr','id' )

# converts matrix to tibble for better viewing
trial.summary = as_tibble(trial.summary)
trial.summary
```

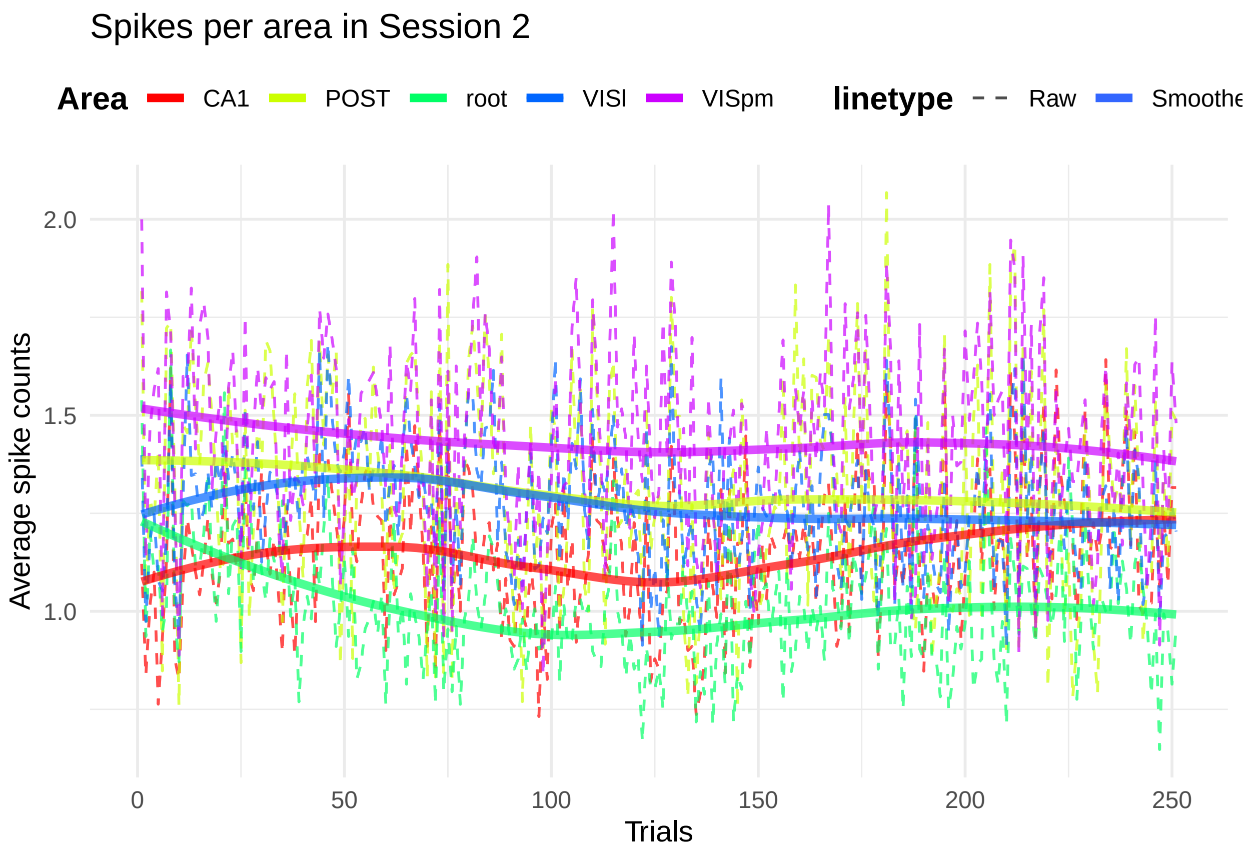
```
## # A tibble: 251 x 9
##      CA1  POST  root  VISl VISpm feedback left_contr right_contr    id
##    <dbl> <dbl> <dbl> <dbl> <dbl>    <dbl>      <dbl>       <dbl> <dbl>
##  1 1.12  1.82  1.54  1.40  2           -1       1            1       1
##  2 0.837 1.04  0.936 0.974 1.14         1       0.25         0       2
##  3 1.02  1.29  1.21  1.18  1.47        -1       0.5          0.5     3
##  4 1.1   1.24  1.10  1.25  1.57         1       0.25         0       4
##  5 0.763 1.41  1.12  1.08  1.62        -1       0            0       5
##  6 0.932 0.848 0.949 1.05  0.990        1       0            0       6
##  7 1.13  1.72  1.01  1.19  1.81         1       1             0       7
##  8 1.62  1.74  1.67  1.58  1.71         1       0             1       8
##  9 0.879 1.32  0.974 1.07  1.27        -1       0             0       9
## 10 0.826 0.759 0.872 0.939 0.887        1       0             0      10
## # i 241 more rows
```

Looking at the provided table from the matrix can help us look at how differing spike averages correspond with different feedback outcomes and contrasts from trial to trial. To better analyze spike averages across the whole session we can use the newly provided matrix to create a line graph depicting brain activity across all trials in the session.

```r
# Organizes tibble columns in order to create plot
trial_summary_long = trial.summary %>% select(!c(feedback, right_contr, left_contr)) %>%
  pivot_longer(cols = -id, names_to = "Area", values_to = "Spike_Count")

# number of brain areas
n.area = length(unique(trial_summary_long$Area))

# line graph for average neural spikes for each trial in a session
ggplot(trial_summary_long, aes(x = id, y = Spike_Count, color = Area)) +
  geom_line(aes(linetype = "Raw"), size = 0.5, alpha = 0.7) +
  geom_smooth(aes(linetype = "Smoothed"), method = "loess", se = FALSE, linewidth = 1.5) +
  scale_color_manual(values = rainbow(n = n.area, alpha = 0.7)) +
  scale_linetype_manual(values = c("Raw" = 2, "Smoothed" = 1)) +
  labs(title = paste("Spikes per area in Session", s), x = "Trials", y = "Average spike counts") +
  theme_minimal() +
  theme(legend.position = "top", legend.title = element_text(size = 12, face = "bold"))
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

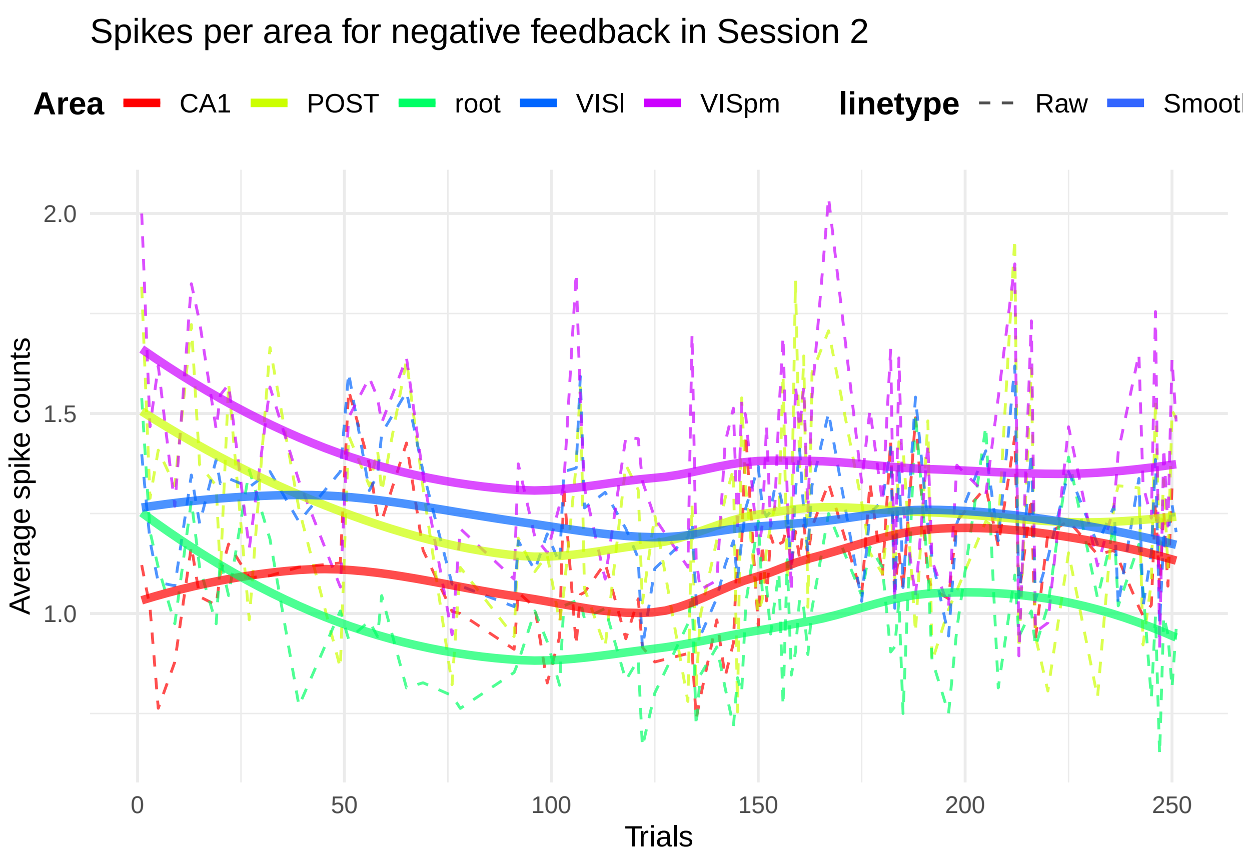
```
## ‘geom_smooth()‘ using formula = ’y ~ x’
```



This visualization can be used to better understand the trends in average neuron activation across all trials in session 2. Total neural activity can be seen gradually decreasing across the session while certain individual brain areas fluctuate more than others. Another possible important distinction that can be made is by splitting this plot into two separate plots. The first will correspond to the brain activity when the feedback outcome is negative and the second will correspond to the brain activity when the feedback outcome is positive.

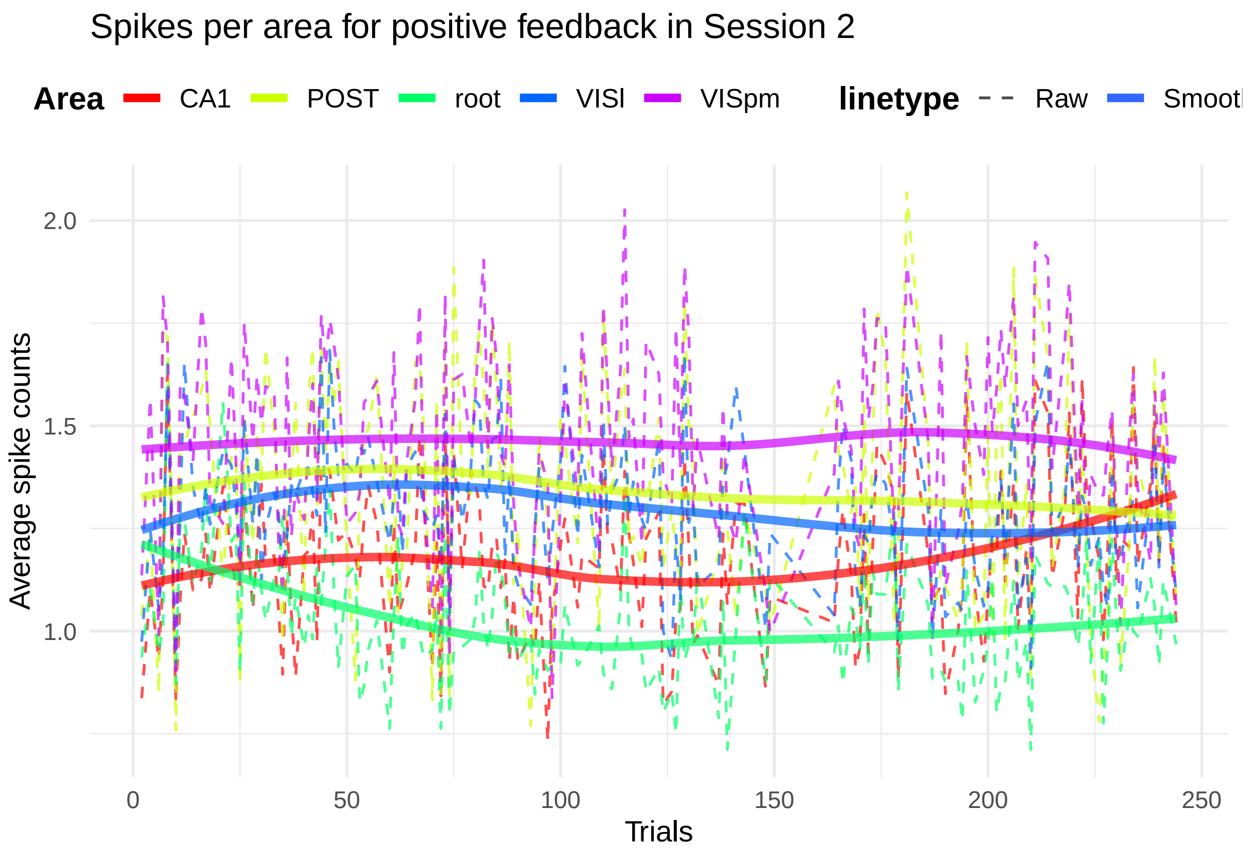
```
# filters trials by feedback outcome
# negative feedback trials
trial_summary_f = trial.summary %>% filter(feedback == -1) %>% select(!c(feedback, right_contr, left_con
# positive feedback trials
trial_summary_t = trial.summary %>% filter(feedback == 1) %>% select(!c(feedback, right_contr, left_con

# line graph using only negative feedback trials
ggplot(trial_summary_f, aes(x = id, y = Spike_Count, color = Area)) +
  geom_line(aes(linetype = "Raw"), size = 0.5, alpha = 0.7) +
  geom_smooth(aes(linetype = "Smoothed"), method = "loess", se = FALSE, linewidth = 1.5) +
  scale_color_manual(values = rainbow(n = n.area, alpha = 0.7)) +
  scale_linetype_manual(values = c("Raw" = 2, "Smoothed" = 1)) +
  labs(title = paste("Spikes per area for negative feedback in Session", s), x = "Trials", y = "Average
  theme_minimal() +
  theme(legend.position = "top", legend.title = element_text(size = 12, face = "bold"), legend.text = el
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



```r
#line graph using only positive feedback trials
ggplot(trial_summary_t, aes(x = id, y = Spike_Count, color = Area)) +
  geom_line(aes(linetype = "Raw"), size = 0.5, alpha = 0.7) +
  geom_smooth(aes(linetype = "Smoothed"), method = "loess", se = FALSE, linewidth = 1.5) +
  scale_color_manual(values = rainbow(n = n.area, alpha = 0.7)) +
  scale_linetype_manual(values = c("Raw" = 2, "Smoothed" = 1)) +
  labs(title = paste("Spikes per area for positive feedback in Session", s), x = "Trials", y = "Average
  theme_minimal() +
  theme(legend.position = "top", legend.title = element_text(size = 12, face = "bold"), legend.text = el
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



When separating the average spike count for the session into negative and positive feedback, comparing the two shows that the negative feedback plot seems to accentuate the slight downward trend in the original line graph while the positive feedback plot seems to flatten out. The negative feedback line graph also shows a large spike at the start followed by much more fluctuation across all trials. While this could help indicate that negative feedback is associated with lower neural activity, many of these results could simply be a result of its relatively smaller sample size.

```
# average spike count and standard deviation of spike counts for each brain area in a session
brain_area_summary = trial_summary_long %>% group_by(Area) %>%
  summarise(avg_spike_count = mean(Spike_Count), sd_spike_count = sd(Spike_Count))

# values for negative feedback
brain_area_summary_f = trial_summary_f %>% group_by(Area) %>%
  summarise(avg_spike_count = mean(Spike_Count), sd_spike_count = sd(Spike_Count))

# values for positive feedback
brain_area_summary_t = trial_summary_t %>% group_by(Area) %>%
  summarise(avg_spike_count = mean(Spike_Count), sd_spike_count = sd(Spike_Count))

# merge all values together
merged_summary <- brain_area_summary %>% full_join(brain_area_summary_f, by = "Area", suffix = c("", "_

#display table containing all values
kable(merged_summary, format = "html", table.attr = "class='table table-striped'", digits = 2)
```

Area

avg_spike_count

sd_spike_count

avg_spike_count_f

sd_spike_count_f

avg_spike_count_t

sd_spike_count_t

CA1

1.15

0.18

1.11

0.16

1.18

0.19

POST

1.31

0.28

1.24

0.26

1.35

0.28

VISl

1.27

0.17

1.24

0.15

1.30

0.17

VISpm

1.43

0.25

1.38

0.25

1.46

0.24

root

1.01

0.16

0.99

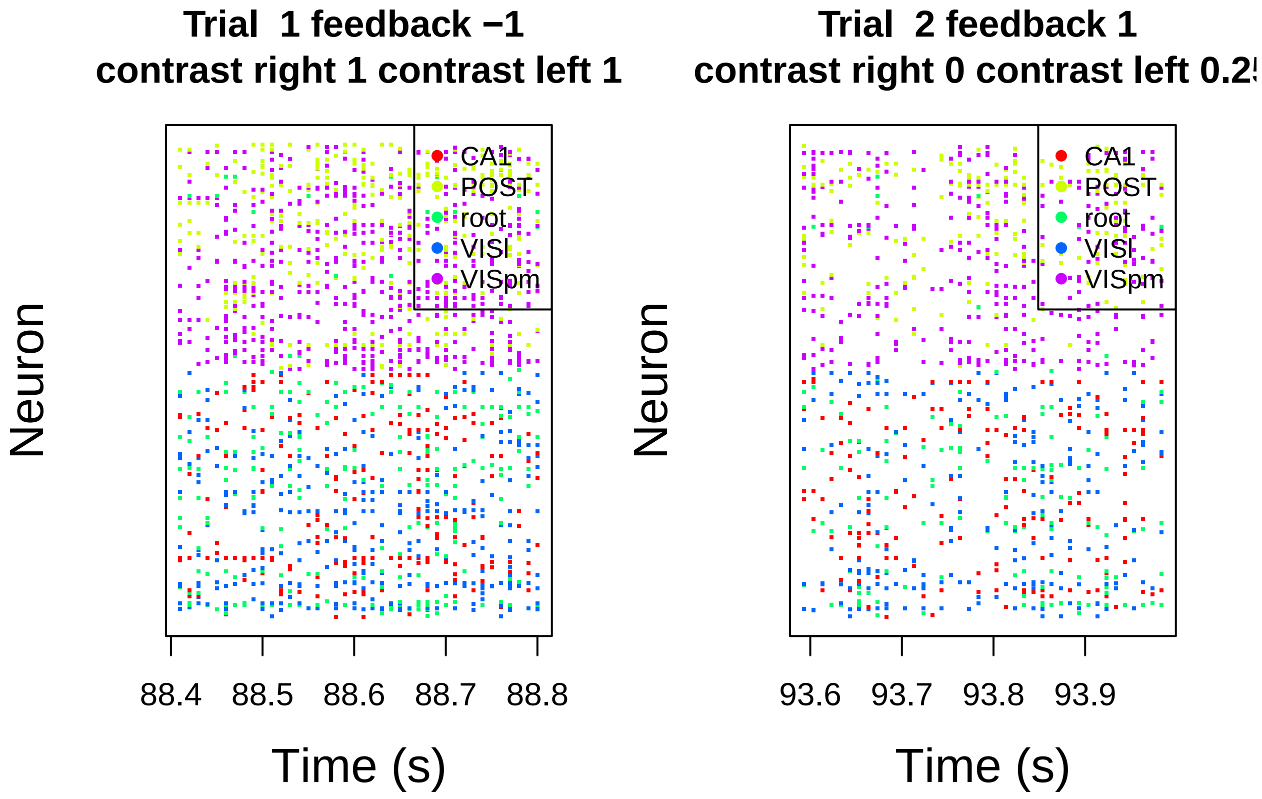
0.18

1.03

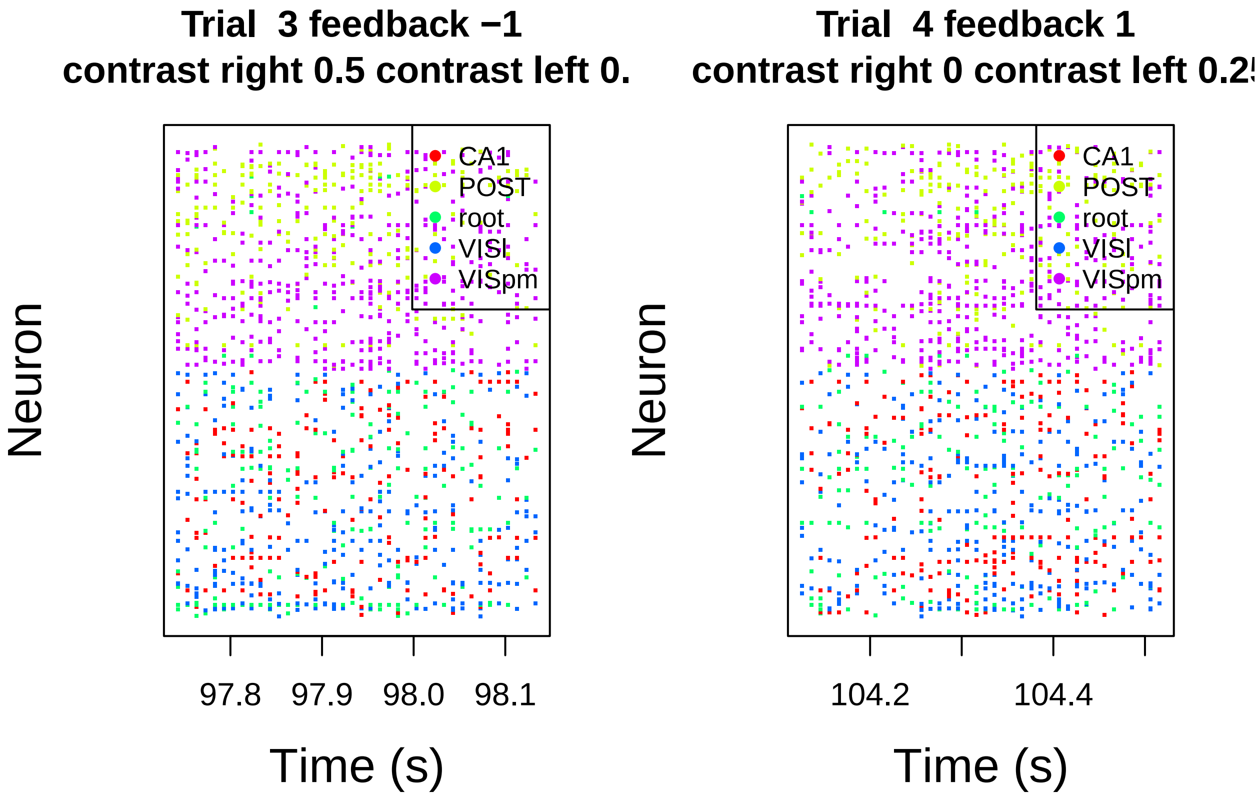
0.16

Looking at the average and standard deviations for spike counts for the session and comparing them according to feedback outcome shows that negative feedback corresponded with a lower neural spike count on average. Standard deviations on the other hand proved to be relatively similar whether or not the feedback was positive.

```
# function to create scatterplots for activated neurons vs time
plot.trial = function(t,area, area.col,this_session){
    spks = this_session$spks[[t]];
    n.neuron = dim(spks)[1]
    time.points = this_session$time[[t]]
    plot(0,0,xlim = c(min(time.points),max(time.points)),ylim = c(0,n.neuron+1),col = 'white', xlab = 'T
    # loop through each neuron
    for(i in 1:n.neuron){
        i.a = which(area == this_session$brain_area[i]);
        col.this = area.col[i.a]
        ids.spike = which(spks[i,] >0)
        if( length(ids.spike) >0 ){
            points(x = time.points[ids.spike],y = rep(i, length(ids.spike) ),pch = '.',cex = 2, col = c
        }
    }
legend("topright",
  legend = area,
  col = area.col,
  pch = 16,
  cex = 0.8)
}

varname = names(trial.summary);
area = varname[1:(length(varname) -4)]
area.col = rainbow(length(area))

# creates side by side plots of the first two trials
varname = names(trial.summary);
area = varname[1:(length(varname) -4)]
par(mfrow = c(1,2))
plot.trial(1,area, area.col,session[[s]])
plot.trial(2,area, area.col,session[[s]])
```



```
# creates side by side plots of two other trials
plot.trial(3,area, area.col,session[[s]])
plot.trial(4,area, area.col,session[[s]])
```



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

Looking at these scatterplots of neuron activation vs time gives a better insight into both the relationship between neural activation with feedback outcome and the relationship between contrast and neural activation. While these few scatterplots are a very limited sample for this comparison, they do highlight the importance that contrast has on neural activation, making contrast an imporant variable to investigate further, later on. For now, we'd like to look further into the difference in neural activity and how it relates to feedback outcome.

```
avg_spikes_positive = list()
avg_spikes_negative = list()

# loop through each session
for (s in 1:n.session) {
  areas = unique(session[[s]]$brain_area)
  avg_spikes_pos = numeric(length(areas))
  avg_spikes_neg = numeric(length(areas))
  names(avg_spikes_pos) = areas
  names(avg_spikes_neg) = areas
  # loop through each brain area in each session
  for (area in areas) {
    neuron_indices = which(session[[s]]$brain_area == area)
    total_spikes_pos = 0
    total_spikes_neg = 0
    count_pos = 0
    count_neg = 0
    # loop through each trial in each brain area to find the amount of spikes depending on feedback
    for (i in 1:length(session[[s]]$feedback_type)) {
      if (session[[s]]$feedback_type[i] == 1) {
        total_spikes_pos = total_spikes_pos + sum(session[[s]]$spks[[i]][neuron_indices, ])
        count_pos = count_pos + 1
      } else if (session[[s]]$feedback_type[i] == -1) {
        total_spikes_neg = total_spikes_neg + sum(session[[s]]$spks[[i]][neuron_indices, ])
        count_neg = count_neg + 1
      }
    }
    # calculates average positive and negative feedback spikes for each brain area
    avg_spikes_pos[area] = ifelse(count_pos > 0, total_spikes_pos / (length(neuron_indices) * count_pos
    avg_spikes_neg[area] = ifelse(count_neg > 0, total_spikes_neg / (length(neuron_indices) * count_neg
  }
  # assigns average positive and negative feedback spikes for each session
  avg_spikes_positive[[s]] = avg_spikes_pos
  avg_spikes_negative[[s]] = avg_spikes_neg
}

# combines results into a data frame
avg_spikes_combined = do.call(rbind, lapply(1:n.session, function(s) {
  data.frame(Session = s,
    BrainArea = names(avg_spikes_positive[[s]]),
    AvgSpikes_Positive = avg_spikes_positive[[s]],
    AvgSpikes_Negative = avg_spikes_negative[[s]])
}))

# gets all unique brain areas across all sessions
brain_areas_all = unique(unlist(lapply(session, function(s) unique(s$brain_area))))

# filters for brain areas that are present in at least one session
avg_spikes_combined = avg_spikes_combined %>% filter(BrainArea %in% brain_areas_all)

# summarizes across sessions
avg_spikes_summary = avg_spikes_combined %>%
  group_by(BrainArea) %>%
  summarise(AvgSpikes_Positive = mean(AvgSpikes_Positive, na.rm = TRUE), AvgSpikes_Negative = mean(AvgS
  # adds column for difference in average positive and negative feedback spikes
  mutate(Difference = AvgSpikes_Positive - AvgSpikes_Negative)

# displays the newly created table
kable(avg_spikes_summary, format = "html", table.attr = "class='table table-striped'", digits = 2)
```

BrainArea

AvgSpikes_Positive

AvgSpikes_Negative

Difference

ACA

0.71

0.67

0.03

ACB

0.81

0.74

0.07

AUD

0.41

0.41

0.00

BLA

1.21

1.17

0.03

CA1

1.20

1.10

0.10

CA3

2.02

1.86

0.16

CP

0.97

0.84

0.13

DG

1.79

1.64

0.15

EPd

0.51

0.51

0.00

GPe

0.94

0.87

0.07

ILA

1.46

1.32

0.14

LD

2.47

2.41

0.06

LGd

1.78

1.65

0.13

LH

4.46

4.52

-0.06

LP

2.82

2.61

0.20

LS

1.57

1.34

0.23

LSc

1.14

0.90

0.23

LSr

1.43

1.20

0.23

MB

2.35

2.21

0.14

MD

2.44

2.24

0.19

MEA

0.95

1.01

-0.06

MG

2.78

2.64

0.14

MOp

1.37

1.09

0.28

MOs

0.76

0.71

0.04

MRN

3.67

3.51

0.16

MS

4.49

4.81

-0.32

NB

2.08

2.06

0.02

OLF

0.31

0.30

0.01

ORB

0.59

0.57

0.02

ORBm

1.05

0.98

0.07

OT

0.86

0.82

0.04

PAG

0.67

0.65

0.02

PIR

1.64

1.58

0.06

PL

1.01

0.95

0.06

PO

2.37

1.96

0.40

POL

1.41

1.39

0.02

POST

1.08

0.99

0.09

PT

1.28

1.18

0.10

RN

8.45

7.81

0.64

RSP

0.51

0.44

0.07

RT

1.30

1.26

0.03

SCm

1.96

1.93

0.03

SCs

1.80

1.65

0.15

SCsg

1.26

1.01

0.26

SI

1.56

1.59

-0.03

SNr

1.10

0.99

0.11

SPF

4.49

4.49

0.00

SSp

0.60

0.56

0.04

SSs

0.72

0.63

0.09

SUB

1.57

1.41

0.16

TH

0.97

0.90

0.07

TT

1.12

1.00

0.12

VISa

0.99

0.84

0.15

VISam

1.76

1.56

0.20

VISl

1.28

1.22

0.07

VISp

1.22

1.09

0.13

VISpm

1.46

1.38

0.08

VISrl

0.43

0.43

0.01

VPL

2.38

2.17

0.20

VPM

1.45

1.33

0.12

ZI

1.90

1.83

0.07

root

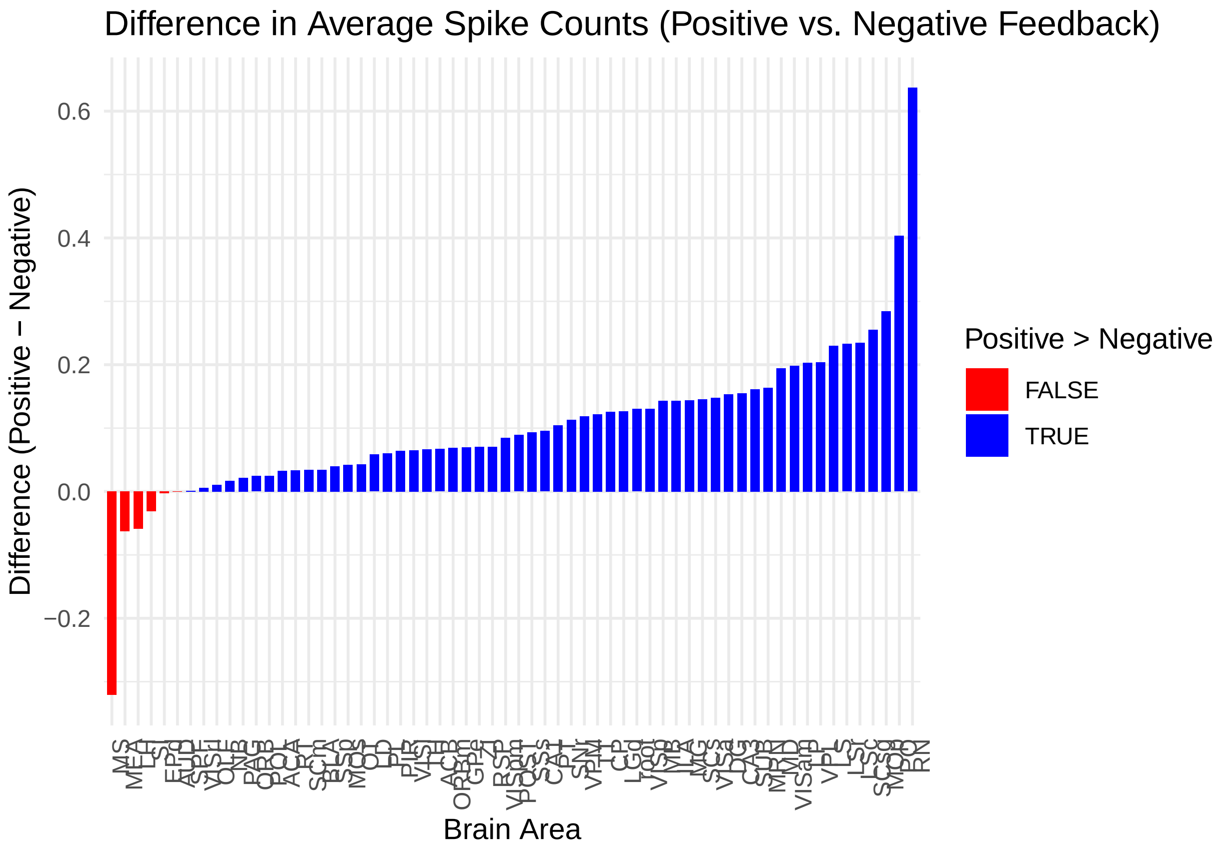
1.27

1.14

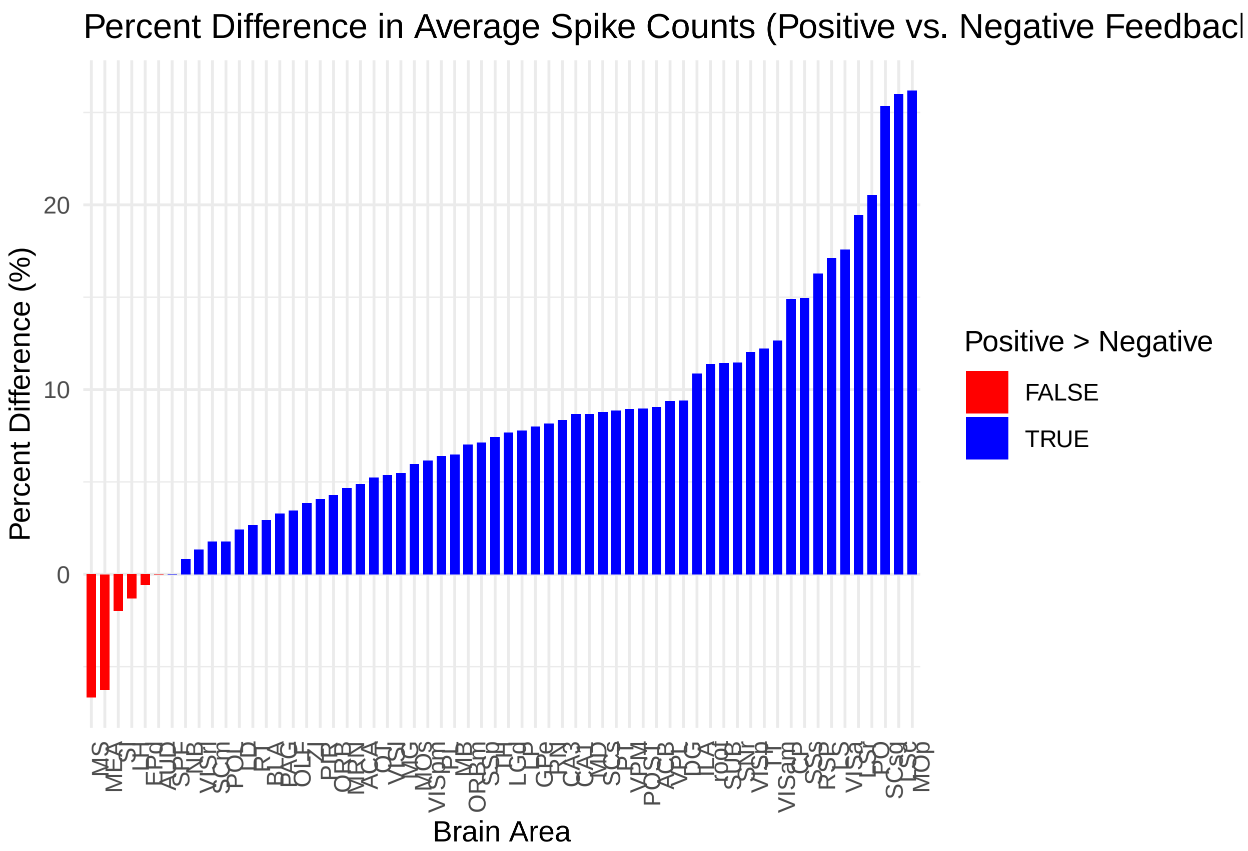
0.13

This table displays the difference between neural activity in all brain regions extending across all sessions. When looking at the data as a whole, the difference between neural activity associated with negative feedback vs positive feedback shows that almost every brain region contains higher neural activation when the feedback is positive. This gives greater confidence in the idea of higher neural spiking relating to increased rate of positive feedback on a larger scale which in turn makes it reasonable idea to use in the final prediction model.

```r
# bar plot showing differences in average neural spikes across all brain areas
ggplot(avg_spikes_summary, aes(x = reorder(BrainArea, Difference), y = Difference, fill = Difference > 0
  geom_bar(stat = "identity", width = 0.7) +
  scale_fill_manual(values = c("TRUE" = "blue", "FALSE" = "red")) +
  labs(title = "Difference in Average Spike Counts (Positive vs. Negative Feedback)", x = "Brain Area",
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



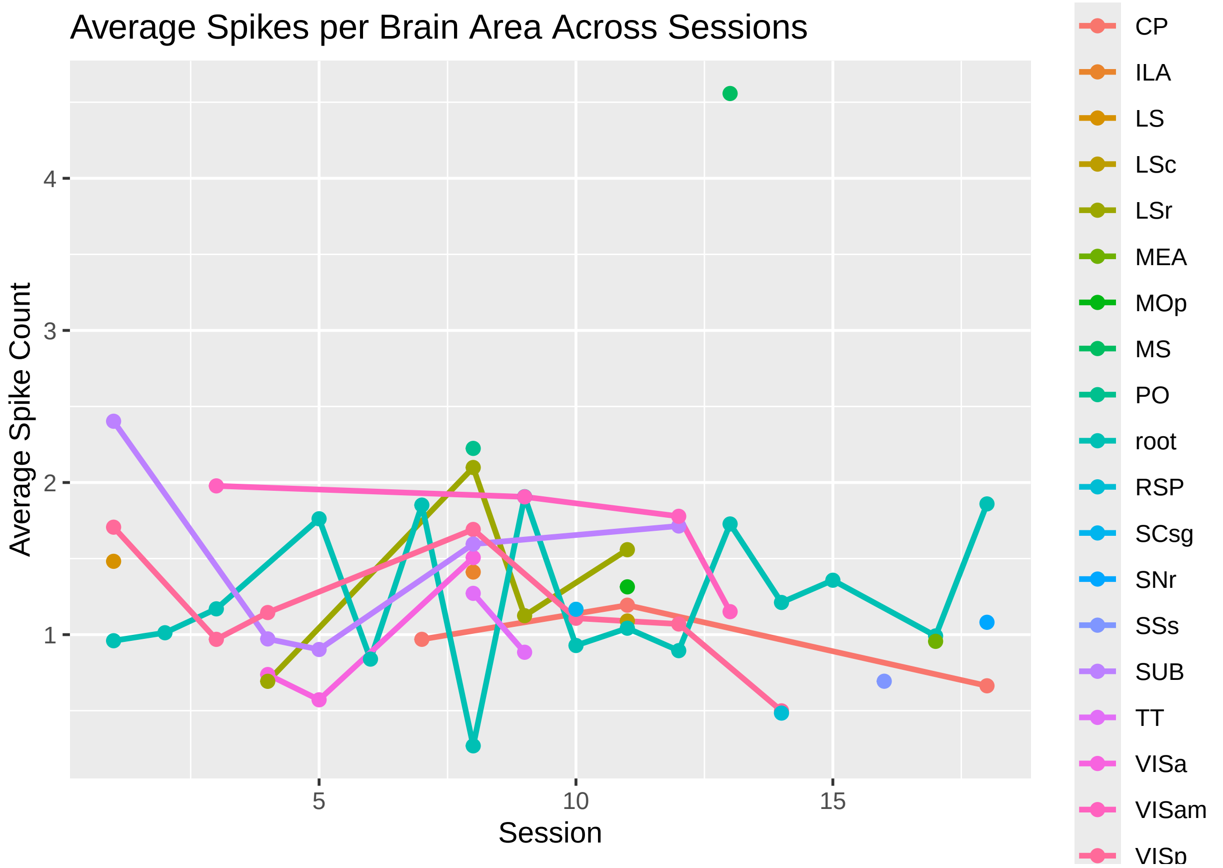
```
# calculates percent difference
avg_spikes_summary = avg_spikes_summary %>%
  mutate(Percent_Difference = ((AvgSpikes_Positive - AvgSpikes_Negative) / AvgSpikes_Negative) * 100)

# bar plot for percent difference in average neural spikes per brain area
ggplot(avg_spikes_summary, aes(x = reorder(BrainArea, Percent_Difference), y = Percent_Difference, fill
  geom_bar(stat = "identity", width = 0.7) +
  scale_fill_manual(values = c("TRUE" = "blue", "FALSE" = "red")) +
  labs(title = "Percent Difference in Average Spike Counts (Positive vs. Negative Feedback)", x = "Brai
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```



```
# takes brain areas with above a 10% increase for positive feedback and brain areas below 5% decrease i
notable_percent_diff = avg_spikes_summary %>% filter(Percent_Difference > 10 | Percent_Difference < -5)
```

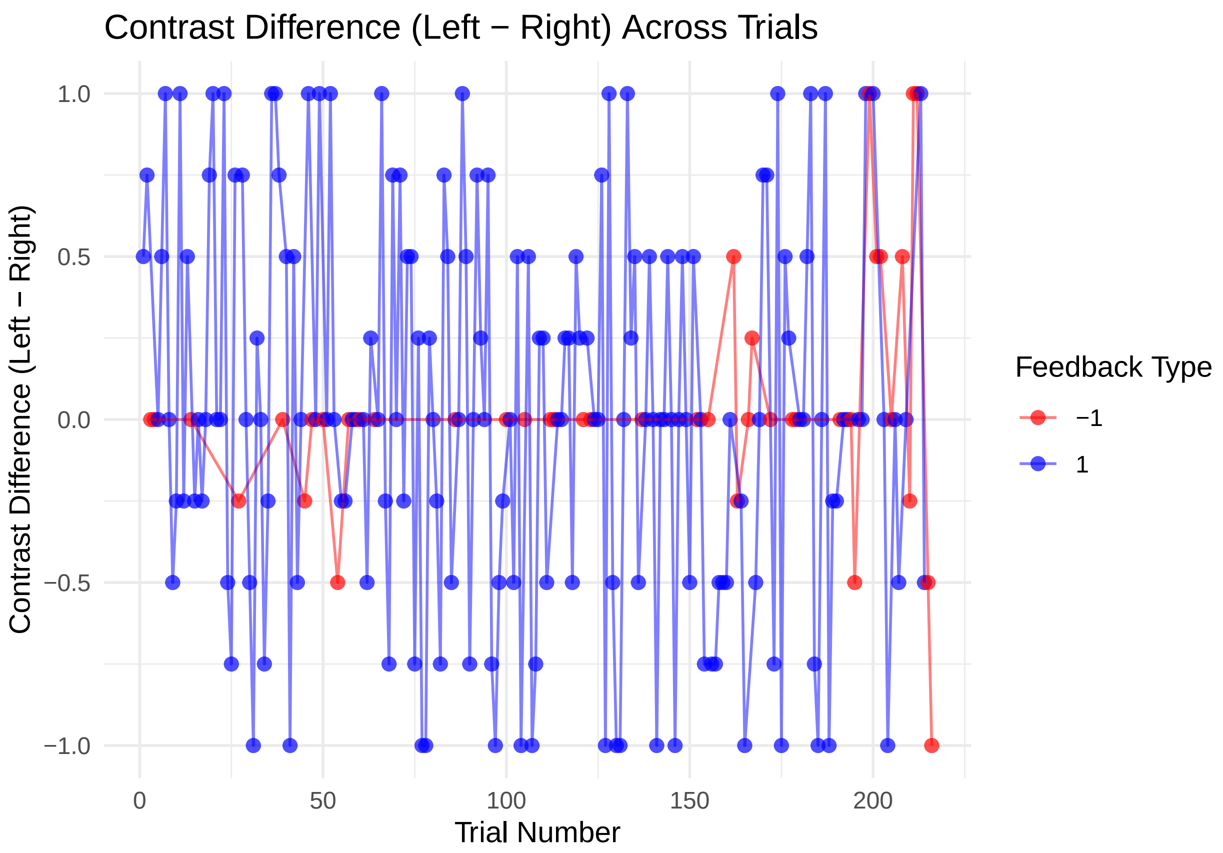
One possible area of interest when it comes to neural activity for this data is the relative increases and decreases in neural activity across brain areas. Since certain brain areas seem to have larger neuron counts, its possible that looking into percent increases could be a better metric for predicting feedbck outcome since it would allow other brain areas to have equal impact on the data despite having less neurons tested. The two bar plots shown above highlight the differences between brain areas that appear to have a larger positive association with feedback outcome depending solely on number of neuron spikes difference vs relative difference in number of spikes depending on the given brain region.

```
avg_spikes_per_session = list()

# loop through each session for average spikes
for (s in 1:n.session) {
  areas = unique(session[[s]]$brain_area)
  avg_spikes = numeric(length(areas))
  names(avg_spikes) = areas
  # loop through each brain area for average spikes per brain area
  for (area in areas) {
    neuron_indices = which(session[[s]]$brain_area == area)
    total_spikes = 0
    # loops through each trial to sum total spikes
    for (i in 1:length(session[[s]]$feedback_type)) {
      total_spikes = total_spikes + sum(session[[s]]$spks[[i]][neuron_indices, ])
    }
    # calculates the average spikes for the current brain area
    avg_spikes[area] = total_spikes / (length(neuron_indices) * length(session[[s]]$feedback_type))
  }
  avg_spikes_per_session[[s]] = avg_spikes
}

# converts the list to a data frame for plotting
avg_spikes_df = do.call(rbind, lapply(1:n.session, function(s) {
  data.frame(Session = s,
    BrainArea = names(avg_spikes_per_session[[s]]),
    AvgSpikes = avg_spikes_per_session[[s]],
    MouseName = session[[s]]$mouse_name)
}))

# filters notable brain areas found prevbiously
avg_spikes_df = avg_spikes_df %>% filter(BrainArea %in% notable_percent_diff$BrainArea)

# plot the average spikes per brain area for each session, split by mouse
ggplot(avg_spikes_df, aes(x = Session, y = AvgSpikes, color = BrainArea)) +
  geom_line(size = 1) +
  geom_point(size = 2) +
  labs(title = "Average Spikes per Brain Area Across Sessions", x = "Session", y = "Average Spike Count
```

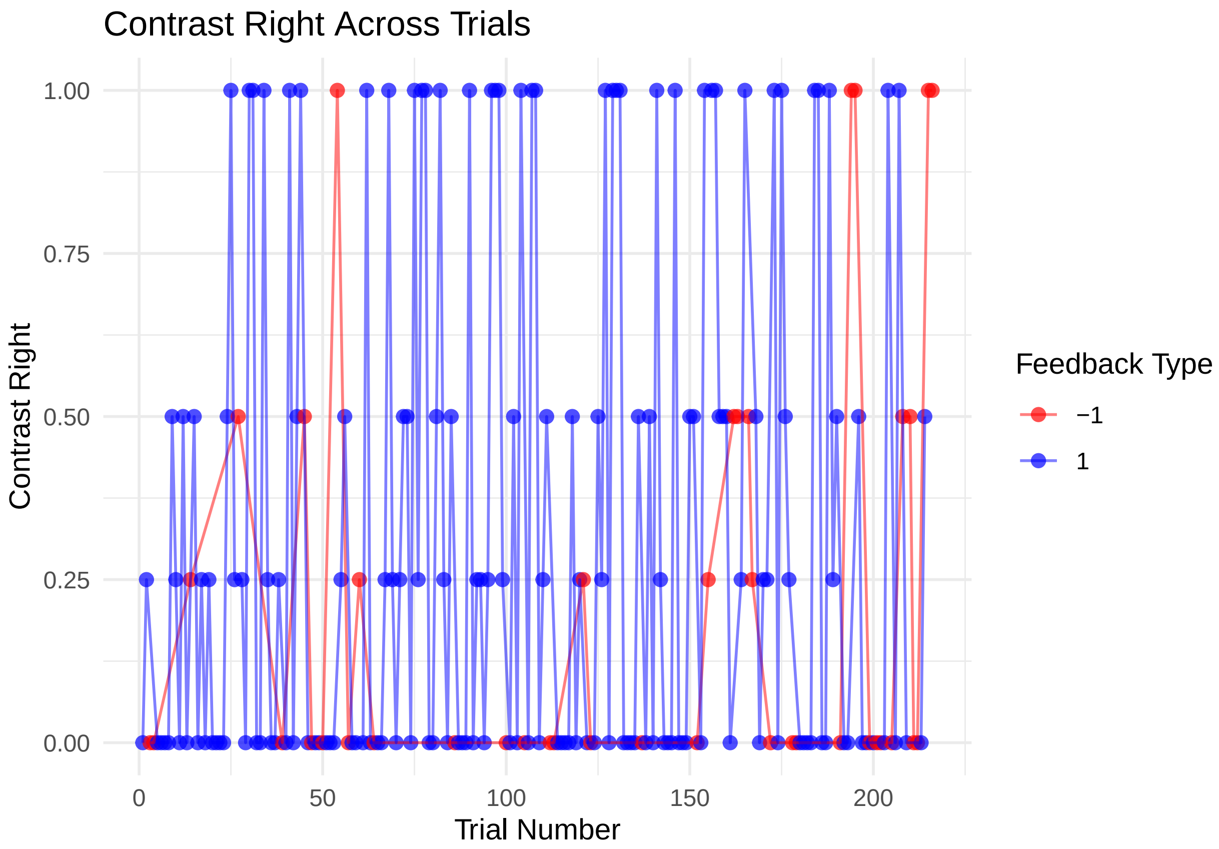


Taking the most prominent brain areas found in the previous bar plot and viewing them on a session by session basis, it becomes clear that attempting to use specific brain areas for the predicitive model will prove difficult since none of the brain areas tested are shared across all trials along with many of them only appearing in a single trial. Because of this, looking into brain activation as a whole may prove to be a safer and more reliable way to predict feedback outcome going forward. Now we'd like to shift focus from neural data to contrast data, more specifically on how the difference in contrast affects feedback outcome.

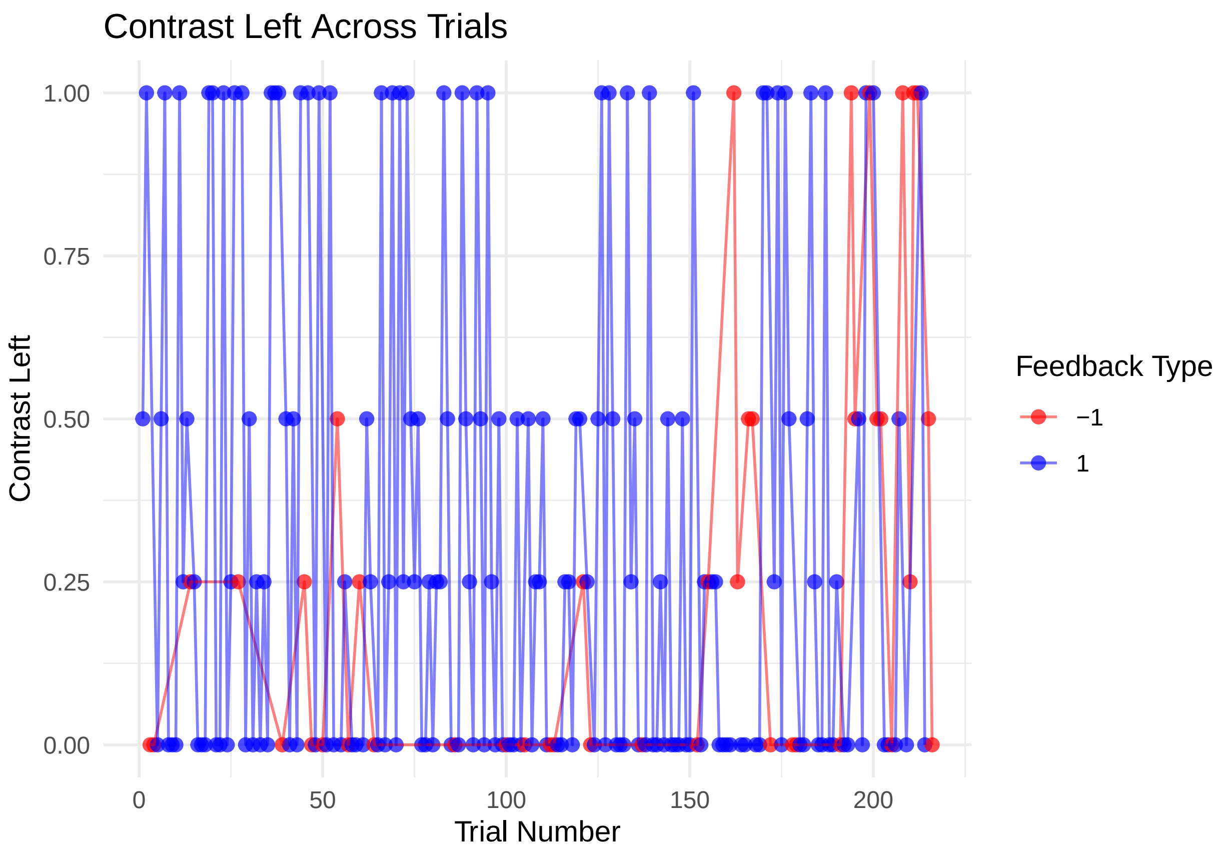
```
# creates tibble for contrast and feedback data
session_df = tibble(trial = 1:length(session[[s]]$contrast_left),
  contrast_left = session[[s]]$contrast_left,
  contrast_right = session[[s]]$contrast_right,
  feedback_type = session[[s]]$feedback_type,
  contrast_diff = session[[s]]$contrast_right - session[[s]]$contrast_left) %>%
  mutate(contrast_diff = contrast_left - contrast_right)

# plots contrast difference across trials in a single session
ggplot(session_df, aes(x = trial, y = contrast_diff, color = factor(feedback_type))) +
  geom_point(size = 2, alpha = 0.7) +
  geom_line(alpha = 0.5) +
  labs(title = "Contrast Difference (Left - Right) Across Trials", x = "Trial Number", y = "Contrast Di
  theme_minimal() +
  scale_color_manual(values = c("-1" = "red", "1" = "blue"))
```



```
#plots only right contrast across trials in a single session
ggplot(session_df, aes(x = trial, y = contrast_right, color = factor(feedback_type))) +
  geom_point(size = 2, alpha = 0.7) +
  geom_line(alpha = 0.5) +
  labs(title = "Contrast Right Across Trials", x = "Trial Number", y = "Contrast Right", color = "Feedba
  theme_minimal() +
  scale_color_manual(values = c("-1" = "red", "1" = "blue"))
```



```
# plots only left contrast across trials
ggplot(session_df, aes(x = trial, y = contrast_left, color = factor(feedback_type))) +
  geom_point(size = 2, alpha = 0.7) +
  geom_line(alpha = 0.5) +
  labs(title = "Contrast Left Across Trials", x = "Trial Number", y = "Contrast Left", color = "Feedbac
  theme_minimal() +
  scale_color_manual(values = c("-1" = "red", "1" = "blue"))
```



Looking at the plots for contrast difference, it is clear that the vast majority negative feedback can be found when the difference in contrast is zero. This makes sense when thinking logically, since non-zero, equal left and right contrasts result in what is essentially a coin flip in feedback result. Another interesting feature of the plot which can be seen in the rapid increase in negative feedback towards the later trials despite larger contrasts difference. This may imply that session duration could also impact feedback outcome.

```r
# counts the number of positive and negative feedback outcomes for each contrast_diff level and calcula
feedback_counts = session_df %>%
  group_by(contrast_diff) %>%
  summarise(Count_Feedback_Negative = sum(feedback_type == -1),
    Count_Feedback_Positive = sum(feedback_type == 1),
    Total_Trials = n(),
    Success_Rate = Count_Feedback_Positive / Total_Trials)

# displays the table containing success rates for every contrast difference
kable(feedback_counts, format = "html", table.attr = "class='table table-striped'", digits = 2)
```

contrast_diff

Count_Feedback_Negative

Count_Feedback_Positive

Total_Trials

Success_Rate

-1.00

1

17

18

0.94

-0.75

0

13

13

1.00

-0.50

3

19

22

0.86

-0.25

4

14

18

0.78

0.00

26

47

73

0.64

0.25

1

13

14

0.93

0.50

4

19

23

0.83

0.75

0

13

13

1.00

1.00

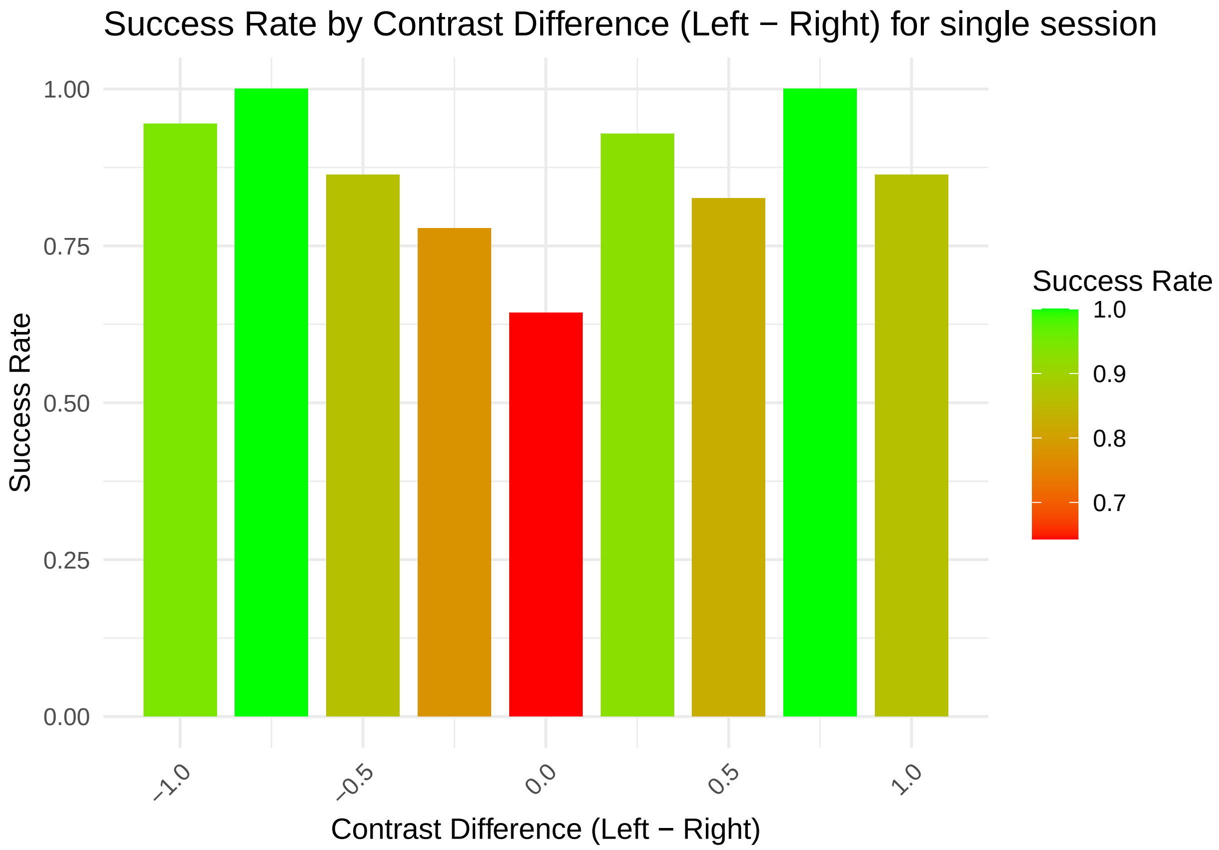
3

19

22

0.86

```r
# bar graph for success rate based on contrast difference
ggplot(feedback_counts, aes(x = contrast_diff, y = Success_Rate, fill = Success_Rate)) +
  geom_bar(stat = "identity", width = 0.2) +
  scale_fill_gradient(low = "red", high = "green") +
  labs(title = "Success Rate by Contrast Difference (Left - Right) for single session", x = "Contrast D
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The table of success rates correlating with contrast differences shows that there were many more trials with a contrast difference of zero which also happened to be the lowest success rate. The bar graph helps visualize the trend showing that success rate increases as contrast difference gets further from zero. Because there are so many trials with a contrast difference equal to zero, it would be useful to examine every combination of contrasts in order to view this data more in depth.

```r
# table for every combination of contrasts
success_rate_table = session_df %>%
  group_by(contrast_left, contrast_right) %>%
  summarise(Total_Trials = n(), Success_Rate = sum(feedback_type == 1) / Total_Trials) %>%
  ungroup()
```

```
## ‘summarise()‘ has grouped output by ’contrast_left’. You can override using the
## ‘.groups‘ argument.
```

```r
kable(success_rate_table, format = "html", table.attr = "class='table table-striped'", digits = 2)
```

contrast_left

contrast_right

Total_Trials

Success_Rate

0.00

0.00

63

0.68

0.00

0.25

8

1.00

0.00

0.50

14

1.00

0.00

1.00

18

0.94

0.25

0.00

8

1.00

0.25

0.25

5

0.20

0.25

0.50

10

0.60

0.25

1.00

13

1.00

0.50

0.00

17

0.88

0.50

0.25

6

0.83

0.50

0.50

3

0.67

0.50

1.00

8

0.62

1.00

0.00

22

0.86

1.00

0.25

13

1.00

1.00

0.50

6

0.67

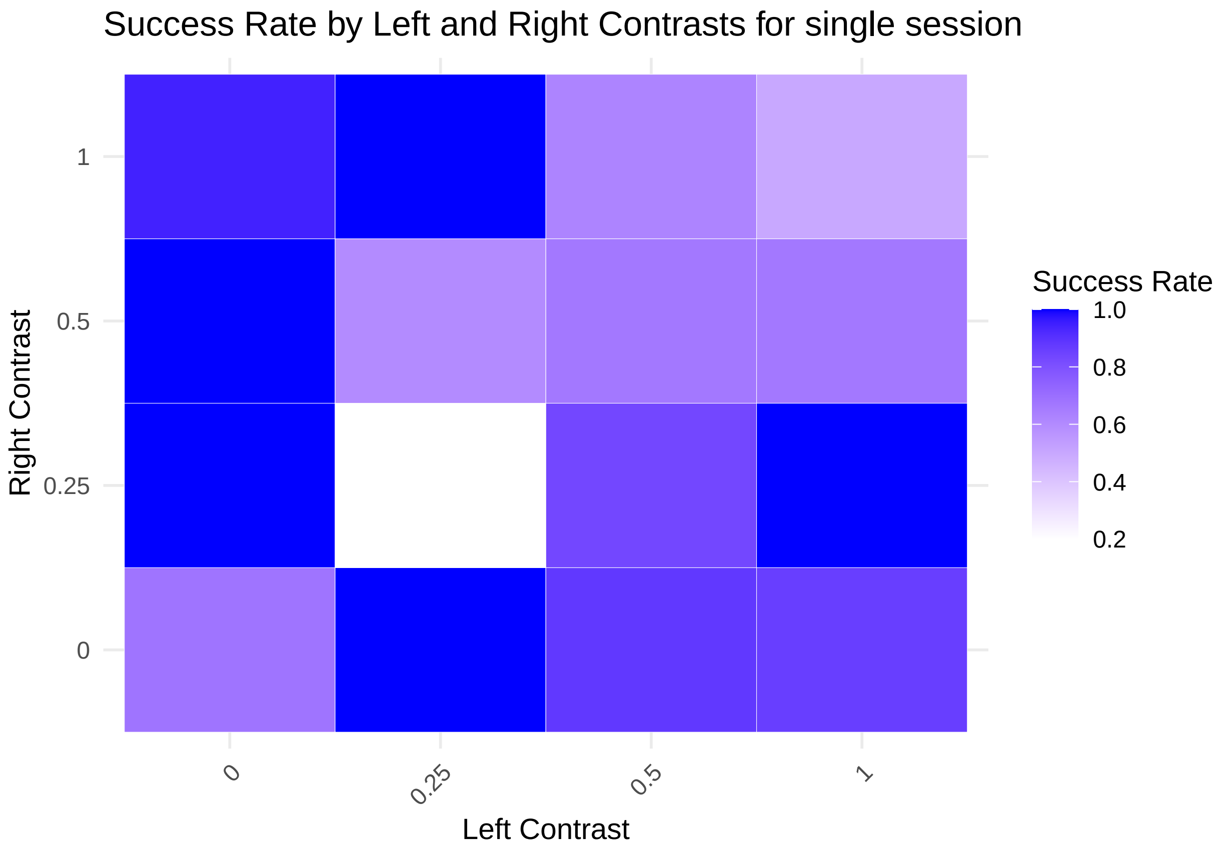
1.00

1.00

2

0.50

```
# plots a heatmap showing success rates for contrasts combinations
ggplot(success_rate_table, aes(x = factor(contrast_left), y = factor(contrast_right), fill = Success_Ra
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Success Rate by Left and Right Contrasts for single session", x = "Left Contrast", y =
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The table displaying the success rates of every combination of contrasts shows that a large majority of the contrast differences of zero come from trials where both both contrasts are both equal to zero as well. Using a heat map to better visualize the many combinations confirms larger contrast differences lead to a greater success rate. It also seems to show that success rates are greatly increased when one of the contrasts is equal to zero even if the difference in contrast is relatively low.

```
all_sessions_df = list()

# loops through each session and combines the data of all sessions
for (s in 1:n.session) {
  session_df = tibble(trial = 1:length(session[[s]]$contrast_left),
```

```r
    contrast_left = session[[s]]$contrast_left,
    contrast_right = session[[s]]$contrast_right,
    feedback_type = session[[s]]$feedback_type,
    mouse_name = session[[s]]$mouse_name,
    contrast_diff = session[[s]]$contrast_right - session[[s]]$contrast_left)
  all_sessions_df[[s]] = session_df
}

# combines data from all sessions into a single data frame
all_sessions_combined = bind_rows(all_sessions_df)

feedback_counts_all = all_sessions_combined %>%
  group_by(contrast_diff) %>%
  summarise(Count_Feedback_Negative = sum(feedback_type == -1), Count_Feedback_Positive = sum(feedback_

# table of success rates for contrast differences across all sessions
kable(feedback_counts_all, format = "html", table.attr = "class='table table-striped'", digits = 2)
```

contrast_diff

Count_Feedback_Negative

Count_Feedback_Positive

Total_Trials

Success_Rate

-1.00

68

370

438

0.84

-0.75

122

301

423

0.71

-0.50

118

438

556

0.79

-0.25

123

222

345

0.64

0.00

626

1060

1686

0.63

0.25

116

257

373

0.69

0.50

115

374

489

0.76

0.75

70

247

317

0.78

1.00

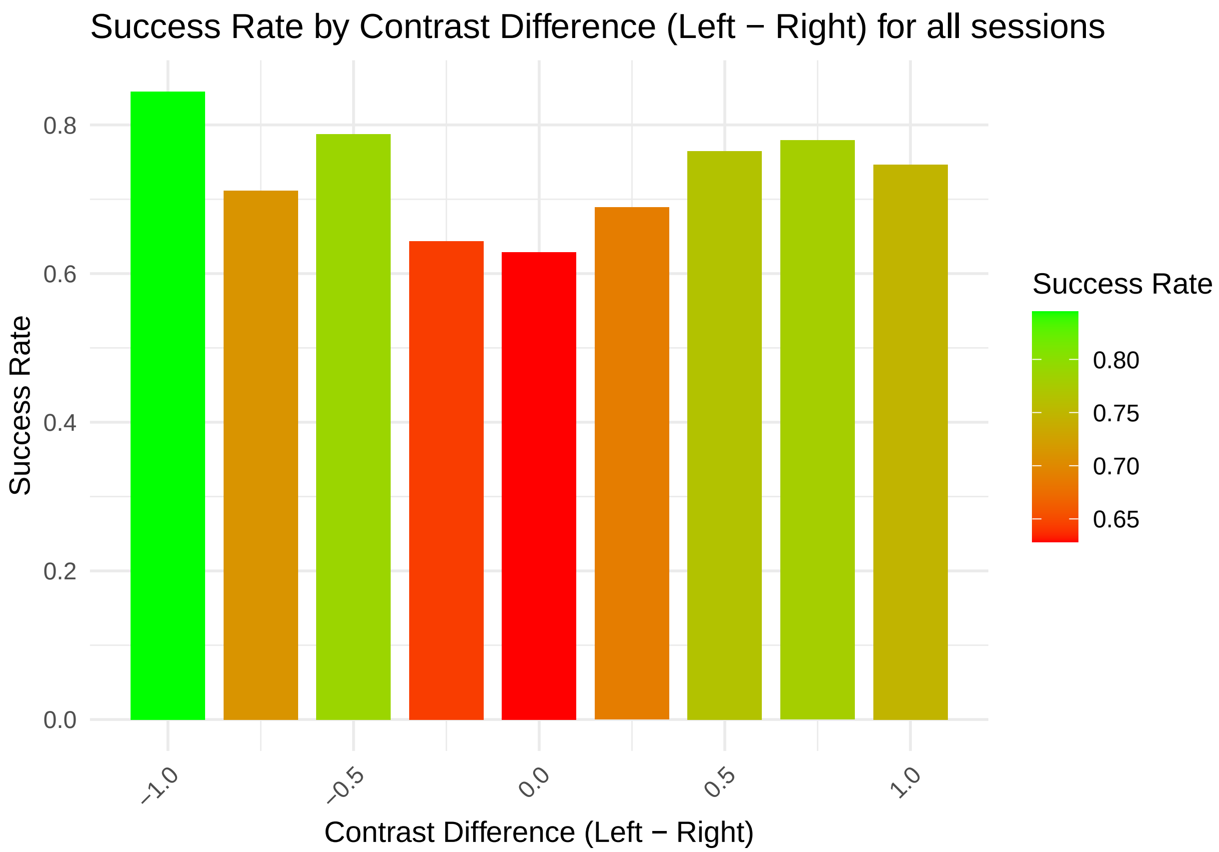
115

339

454

0.75

```r
# plots bar graph of success rates corresponding with contrast differences
ggplot(feedback_counts_all, aes(x = contrast_diff, y = Success_Rate, fill = Success_Rate)) +
  geom_bar(stat = "identity", width = 0.2) +
  scale_fill_gradient(low = "red", high = "green") +
  labs(title = "Success Rate by Contrast Difference (Left - Right) for all sessions", x = "Contrast Dif
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Looking at the success rates for contrast differences across all sessions confirms that increases in contrast difference is positively correlated with success rate for the data as a whole.

```
# calculates success rates for every combination of contrasts across all sessions
success_rate_table_all_sessions <- all_sessions_combined %>%
  group_by(contrast_left, contrast_right) %>%
  summarise(Total_Trials = n(),
    Success_Rate = sum(feedback_type == 1) / Total_Trials) %>%
  ungroup()
```

```
## ‘summarise()‘ has grouped output by ’contrast_left’. You can override using the
## ‘.groups‘ argument.
```

```
# table for success rates for all combinations of contrasts across all sessions
kable(success_rate_table_all_sessions, format = "html", table.attr = "class='table table-striped'", dig
```

contrast_left

contrast_right

Total_Trials

Success_Rate

0.00

0.00

1371

0.66

0.00

0.25

194

0.73

0.00

0.50

326

0.84

0.00

1.00

454

0.75

0.25

0.00

179

0.63

0.25

0.25

99

0.45

0.25

0.50

179

0.64

0.25

1.00

317

0.78

0.50

0.00

397

0.85

0.50

0.25

166

0.66

0.50

0.50

111

0.49

0.50

1.00

163

0.62

1.00

0.00

438

0.84

1.00

0.25

423

0.71

1.00

0.50

159

0.64

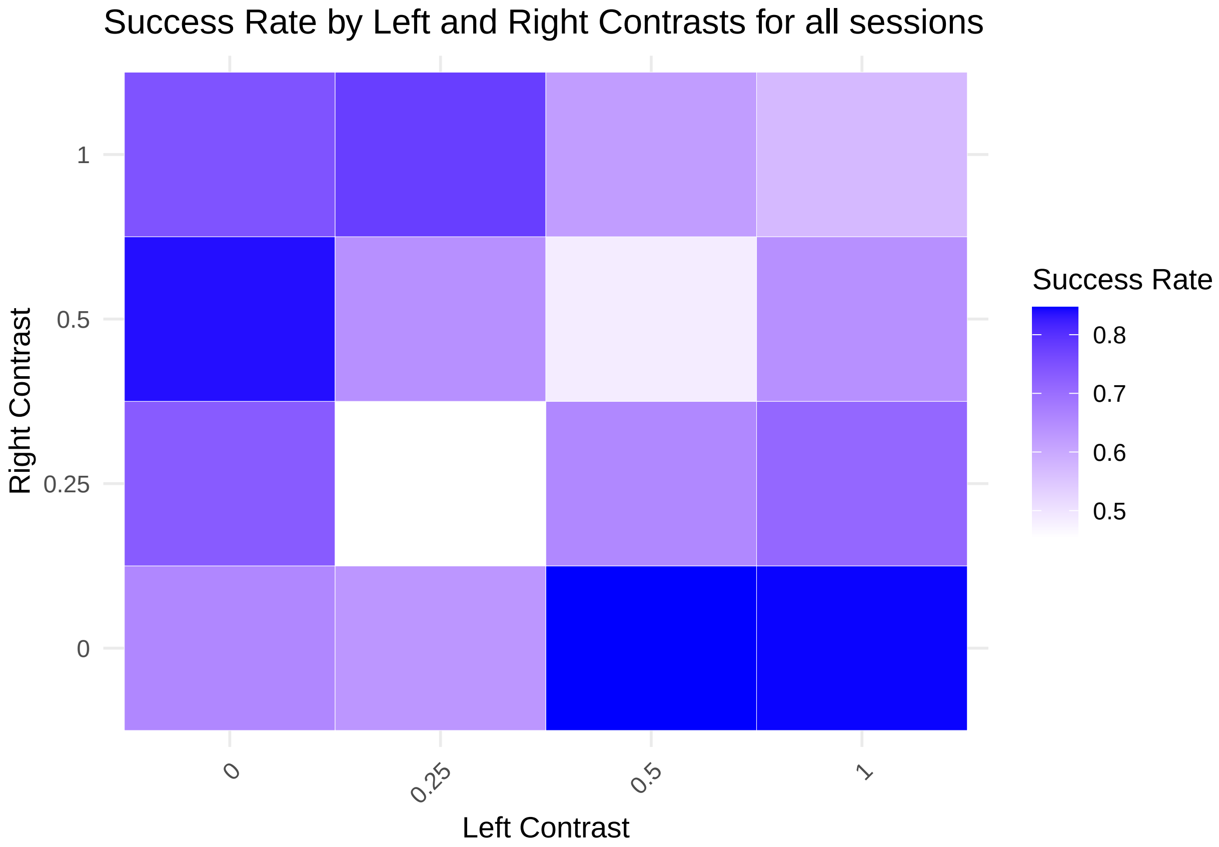
1.00

1.00

105

0.57

```r
# plots a heatmap showing success rates for contrasts combinations across all sessions
ggplot(success_rate_table_all_sessions, aes(x = factor(contrast_left), y = factor(contrast_right), fill
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "blue") +
  labs(title = "Success Rate by Left and Right Contrasts for all sessions", x = "Left Contrast", y = "Ri
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The heat map for contrast combinations across all sessions also shows larger differences in contrast show higher success rate while success rates closer together have a much lower success rate especially when they are non-zero. This allows us to apply our assumptions from earlier to the data as a whole.

Data Integration

When integrating data based on our extensive exploratory analysis, an important place to start is by determining average neural spikes per trial across all sessions.

```r
# loops through sessions to add average spikes for each trial to our session data
for (i in 1:n.session) {
  n.trials = length(session[[i]]$feedback_type)
  avg_spikes_all = numeric(n.trials)
  # loops through each trial finding average neural spikes
  for (j in 1:n.trials) {
    spks.trial = session[[i]]$spks[[j]]
    total.spikes = apply(spks.trial, 1, sum)
    avg_spikes_all[j] = mean(total.spikes)
  }
  # Stores average neural spikes for every trial in a given session together
  session[[i]]$avg_spks = avg_spikes_all
}
```

Another area that was focused on extensively during analysis was the extent to which contrast corresponds with success rate. For this, we will integrate contrast difference into the model as well. Lastly, using data based on contrast combinations we can see that even when both contrasts are high, we may have conflicting outcomes with our two predictors previously mentioned. Because of this, combining these two variables together could prove to be significantly more useful then using each individually. This means that our combined variable measures whether both neural activity and contrast difference are high as a form of predicting feedback outcome.

```r
trials_all = tibble()

# loops through all sessions to combine data
for (i in 1:n.session) {
  curr = session[[i]]
  n.trials = length(curr$feedback_type)

  # creates a tibble of chosen variables for the current session
  trials = tibble(
    mouse_name = rep(curr$mouse_name, n.trials),
    avg_spks = curr$avg_spks,
    contrast_left = as.vector(curr$contrast_left),
    contrast_right = as.vector(curr$contrast_right),
    abs_contrast_diff = as.vector(abs(curr$contrast_right - curr$contrast_left)),
    feedback_type = as.vector(curr$feedback_type),
    session_ID = rep(i, n.trials),
    combined = as.vector((curr$avg_spks * abs(curr$contrast_right - curr$contrast_left)))
  )

  # combines all data across sessions together
  trials_all = bind_rows(trials_all, trials)
}
str(trials_all)
```

```
## tibble [5,081 x 8] (S3: tbl_df/tbl/data.frame)
##  $ mouse_name       : chr [1:5081] "Cori" "Cori" "Cori" "Cori" ...
##  $ avg_spks         : num [1:5081] 1.58 1.31 1.84 1.38 1.43 ...
##  $ contrast_left    : num [1:5081] 0 0 0.5 0 0 0 1 0.5 0 0.5 ...
##  $ contrast_right   : num [1:5081] 0.5 0 1 0 0 0 0.5 0 0 0.25 ...
##  $ abs_contrast_diff: num [1:5081] 0.5 0 0.5 0 0 0 0.5 0.5 0 0.25 ...
##  $ feedback_type    : num [1:5081] 1 1 -1 -1 -1 1 1 1 1 1 ...
##  $ session_ID       : int [1:5081] 1 1 1 1 1 1 1 1 1 1 ...
##  $ combined         : num [1:5081] 0.791 0 0.922 0 0 ...
```

Predictive Modeling

Set up an 80% training data set that of randomly selected trials with the remaining 20% being the test data.

```r
set.seed(123)
# creates training and test data
train_index = sample(1:nrow(trials_all), 0.8 * nrow(trials_all))
train_data = trials_all[train_index, ]
test_data = trials_all[-train_index, ]
```

The final predictors we will use in our model include average neural spikes per trial, contrast right, contrast left, session id, absolute contrast difference, and the combined metric of both average neural spikes and absolute contrast difference.

```
# fits the logistic regression model using training data
fit.mod = glm(
  as.factor(feedback_type) ~ combined + abs_contrast_diff + avg_spks + contrast_right + contrast_left +
  data = train_data,
  family = 'binomial'
)

# summarizes the model
summary(fit.mod)
```

```
##
## Call:
## glm(formula = as.factor(feedback_type) ~ combined + abs_contrast_diff +
##     avg_spks + contrast_right + contrast_left + as.factor(session_ID),
##     family = "binomial", data = train_data)
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -2.36877    0.37358  -6.341 2.29e-10 ***
## combined                 1.64041    0.24898   6.588 4.45e-11 ***
## abs_contrast_diff       -0.75470    0.35027  -2.155 0.031194 *
## avg_spks                 1.68104    0.20719   8.114 4.91e-16 ***
## contrast_right          -1.04429    0.11817  -8.837  < 2e-16 ***
## contrast_left           -0.42030    0.11289  -3.723 0.000197 ***
## as.factor(session_ID)2   0.70435    0.28513   2.470 0.013501 *
## as.factor(session_ID)3  -1.07565    0.32541  -3.306 0.000948 ***
## as.factor(session_ID)4   1.99837    0.30829   6.482 9.05e-11 ***
## as.factor(session_ID)5   1.34913    0.29384   4.591 4.40e-06 ***
## as.factor(session_ID)6   2.69046    0.32235   8.346  < 2e-16 ***
## as.factor(session_ID)7   0.66473    0.28835   2.305 0.021151 *
## as.factor(session_ID)8  -0.06058    0.28842  -0.210 0.833621
## as.factor(session_ID)9   0.12348    0.27430   0.450 0.652585
## as.factor(session_ID)10  0.87358    0.26983   3.237 0.001206 **
## as.factor(session_ID)11  1.80668    0.29876   6.047 1.47e-09 ***
## as.factor(session_ID)12  0.40662    0.28137   1.445 0.148420
## as.factor(session_ID)13 -0.98454    0.34438  -2.859 0.004251 **
## as.factor(session_ID)14  1.64698    0.29700   5.545 2.93e-08 ***
## as.factor(session_ID)15  0.93240    0.27292   3.416 0.000634 ***
## as.factor(session_ID)16  1.55011    0.29472   5.260 1.44e-07 ***
## as.factor(session_ID)17  1.99058    0.32028   6.215 5.13e-10 ***
## as.factor(session_ID)18  1.96914    0.31607   6.230 4.66e-10 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4889.7  on 4063  degrees of freedom
## Residual deviance: 4436.6  on 4041  degrees of freedom
## AIC: 4482.6
##
## Number of Fisher Scoring iterations: 5
```

Testing our newly created model using a confusion matrix and an ROC curve that our accuracy in predicting the feedback outcome from the test data set is approximately 72% and our area under the curve is 0.698.

```r
# generate predicted probabilities
predicted_probs = predict(fit.mod, newdata = test_data, type = "response")

# converts probabilities to binary predictions
predicted_classes = ifelse(predicted_probs >= 0.5, 1, -1)

# creates a confusion matrix
confusion_matrix = table(Predicted = predicted_classes, Actual = test_data$feedback_type)
confusion_matrix
```

```
##          Actual
## Predicted  -1   1
##        -1  50  40
##        1  247 680
```

```r
# calculates accuracy
sum(diag(confusion_matrix)) / sum(confusion_matrix)
```

```
## [1] 0.7177974
```

```r
# calculates ROC and area under the curve
roc_curve = roc(response = test_data$feedback_type, predictor = predicted_probs)
```

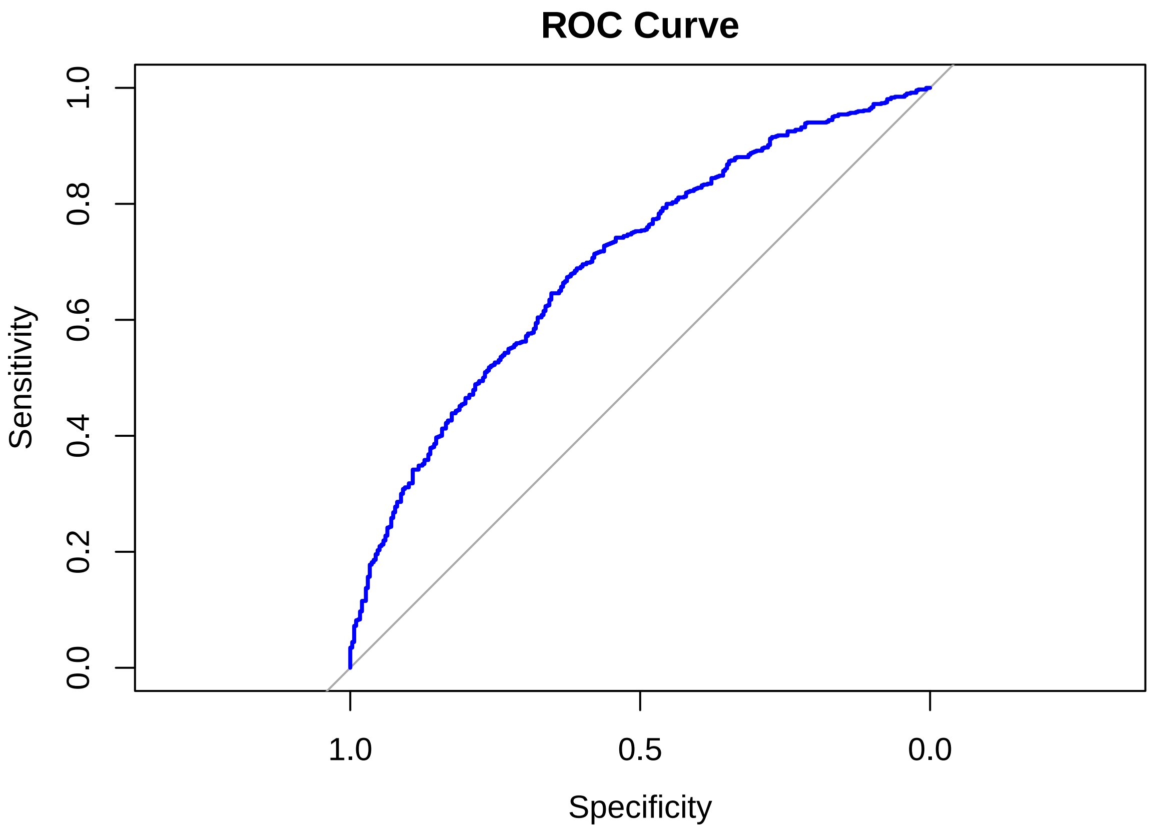
```
## Setting levels: control = -1, case = 1
```

```
## Setting direction: controls < cases
```

```r
auc(roc_curve)
```

```
## Area under the curve: 0.698
```

```r
# plots ROC curve
plot(roc_curve, main = "ROC Curve", col = "blue", lwd = 2)
```



This means that the model is significantly better than simply guessing the feedback result in each trial. Overall, the model seems to do a decent job in predicting feedback outcome but still could have a lot of potential improvement.

Predictive Performance Testing

Now after being provided with test sessions for 100 randomly selected trials from session 1 and session 18, we can truly confirm its effectiveness. First we need to calculate the variables from our model and integrate them into the test sessions provided.

```
# loads the test sets
test_session1 = readRDS("C:/Users/calvin/Desktop/STA141AProject/test/test1.rds")
test_session18 = readRDS("C:/Users/calvin/Desktop/STA141AProject/test/test2.rds")

# calculates variables from model created for test data in section 1
n.trials = length(test_session1$feedback_type)
avg_spikes_all = numeric(n.trials)

for (j in 1:n.trials) {
    spks.trial = test_session1$spks[[j]]
    total.spikes = apply(spks.trial, 1, sum)
    avg_spikes_all[j] = mean(total.spikes)
}
test_session1$avg_spks = avg_spikes_all
```

```
test_data_1 = tibble(
    mouse_name = rep(test_session1$mouse_name, n.trials),
    avg_spks = test_session1$avg_spks,
    contrast_left = test_session1$contrast_left,
    contrast_right = test_session1$contrast_right,
    abs_contrast_diff = abs(test_session1$contrast_right - test_session1$contrast_left),
    feedback_type = test_session1$feedback_type,
    session_ID = as.integer(rep(1, n.trials)),
    combined = (test_session1$avg_spks * abs(test_session1$contrast_right - test_session1$contrast_left
  )

# calculates variables from model created for test data in section 1
n.trials = length(test_session18$feedback_type)
avg_spikes_all = numeric(n.trials)

for (j in 1:n.trials) {
    spks.trial = test_session18$spks[[j]]
    total.spikes = apply(spks.trial, 1, sum)
    avg_spikes_all[j] = mean(total.spikes)
}
test_session18$avg_spks = avg_spikes_all

test_data_18 = tibble(
    mouse_name = rep(test_session18$mouse_name, n.trials),
    avg_spks = test_session18$avg_spks,
    contrast_left = test_session18$contrast_left,
    contrast_right = test_session18$contrast_right,
    abs_contrast_diff = abs(test_session18$contrast_right - test_session18$contrast_left),
    feedback_type = test_session18$feedback_type,
    session_ID = as.integer(rep(18, n.trials)),
    combined = (test_session18$avg_spks * abs(test_session18$contrast_right - test_session18$contrast_l
  )
```

Now we will refit our model on the entire original data set and use it as a training set for the two test sessions provided.

```
# fits the logistic regression model using data from all trials
fit.mod.performance = glm(
  as.factor(feedback_type) ~ combined + abs_contrast_diff + avg_spks + contrast_right + contrast_left +
  data = trials_all,
  family = 'binomial'
)

# summarizes the model
summary(fit.mod.performance)
```

```
##
## Call:
## glm(formula = as.factor(feedback_type) ~ combined + abs_contrast_diff +
##     avg_spks + contrast_right + contrast_left + as.factor(session_ID),
##     family = "binomial", data = trials_all)
##
## Coefficients:
```

```
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -2.4620     0.3276  -7.516 5.65e-14 ***
## combined                   1.4826     0.2168   6.839 8.00e-12 ***
## abs_contrast_diff         -0.6529     0.3084  -2.117 0.034260 *
## avg_spks                   1.8046     0.1847   9.768  < 2e-16 ***
## contrast_right            -1.0406     0.1053  -9.881  < 2e-16 ***
## contrast_left             -0.4243     0.1007  -4.212 2.53e-05 ***
## as.factor(session_ID)2     0.6957     0.2505   2.777 0.005486 **
## as.factor(session_ID)3    -1.3037     0.2860  -4.559 5.15e-06 ***
## as.factor(session_ID)4     1.9192     0.2699   7.111 1.15e-12 ***
## as.factor(session_ID)5     1.2877     0.2561   5.028 4.96e-07 ***
## as.factor(session_ID)6     2.7161     0.2825   9.614  < 2e-16 ***
## as.factor(session_ID)7     0.6604     0.2512   2.628 0.008577 **
## as.factor(session_ID)8    -0.1411     0.2553  -0.552 0.580622
## as.factor(session_ID)9     0.1016     0.2405   0.422 0.672674
## as.factor(session_ID)10    0.7819     0.2358   3.316 0.000912 ***
## as.factor(session_ID)11    1.8069     0.2631   6.867 6.54e-12 ***
## as.factor(session_ID)12    0.3153     0.2459   1.282 0.199771
## as.factor(session_ID)13   -1.1170     0.3043  -3.670 0.000242 ***
## as.factor(session_ID)14    1.6030     0.2597   6.172 6.75e-10 ***
## as.factor(session_ID)15    0.9278     0.2406   3.857 0.000115 ***
## as.factor(session_ID)16    1.6154     0.2590   6.237 4.47e-10 ***
## as.factor(session_ID)17    2.0521     0.2818   7.283 3.26e-13 ***
## as.factor(session_ID)18    1.9873     0.2791   7.120 1.08e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6118.2  on 5080  degrees of freedom
## Residual deviance: 5554.9  on 5058  degrees of freedom
## AIC: 5600.9
##
## Number of Fisher Scoring iterations: 5
```

Lastly, we will use our model to attempt to predict the feedback outcome of the test sessions. We will also create a confusion matrix and ROC curves to determine the models overall accuracy

```r
# evaluates the model on the test sets
test_session1_pred = predict(fit.mod.performance, newdata = test_data_1, type = "response")
test_session18_pred = predict(fit.mod.performance, newdata = test_data_18, type = "response")

# converts probabilities to binary predictions
predicted_classes_1 = ifelse(test_session1_pred >= 0.5, 1, -1)
predicted_classes_18 = ifelse(test_session18_pred >= 0.5, 1, -1)

# creates a confusion matrices
confusion_matrix_1 = table(Predicted = predicted_classes_1, Actual = test_data_1$feedback_type)
confusion_matrix_1
```

```
##          Actual
## Predicted -1  1
##        -1 20 18
##        1   8 54
```

```
confusion_matrix_18 = table(Predicted = predicted_classes_18, Actual = test_data_18$feedback_type)
confusion_matrix_18
```

```
##           Actual
## Predicted -1  1
##          1 27 73
```

```
# calculates accuracies
sum(diag(confusion_matrix_1)) / sum(confusion_matrix_1)
```

```
## [1] 0.74
```

```
sum(confusion_matrix_18[2]) / sum(confusion_matrix_18)
```

```
## [1] 0.73
```

```
# calculates performance metrics for each test set
roc_session1 = roc(response = test_session1$feedback_type, predictor = test_session1_pred)
```

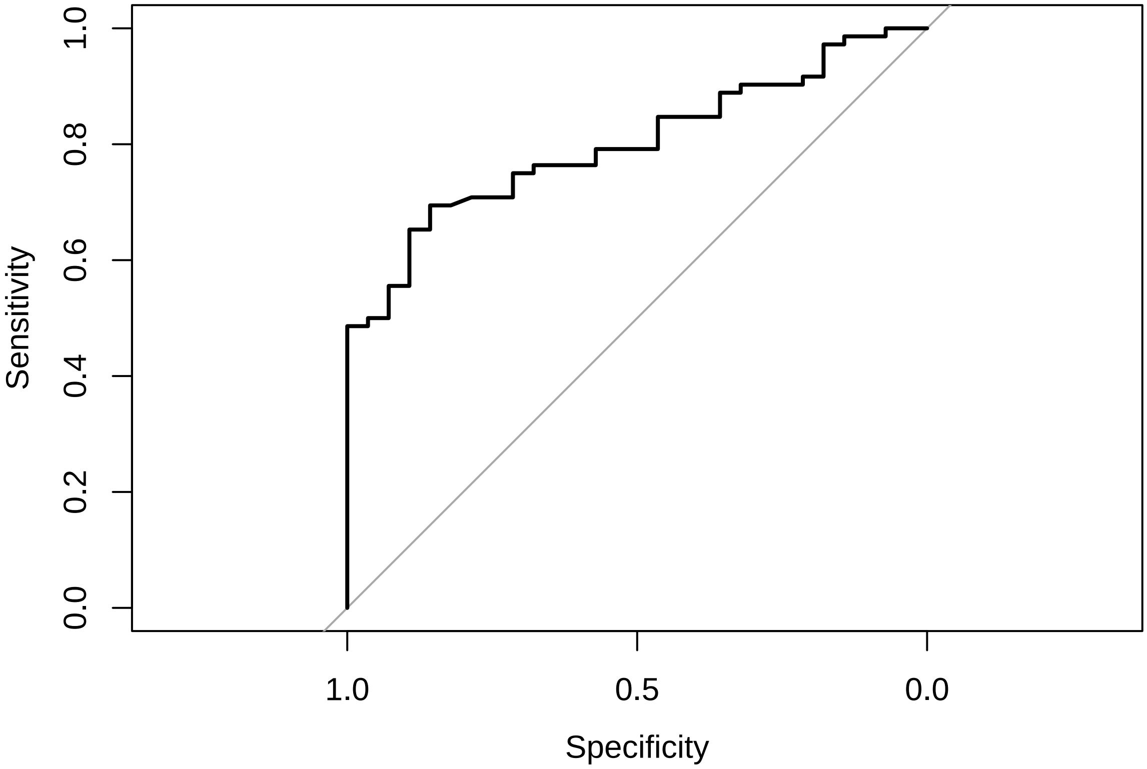
```
## Setting levels: control = -1, case = 1
```

```
## Setting direction: controls < cases
```

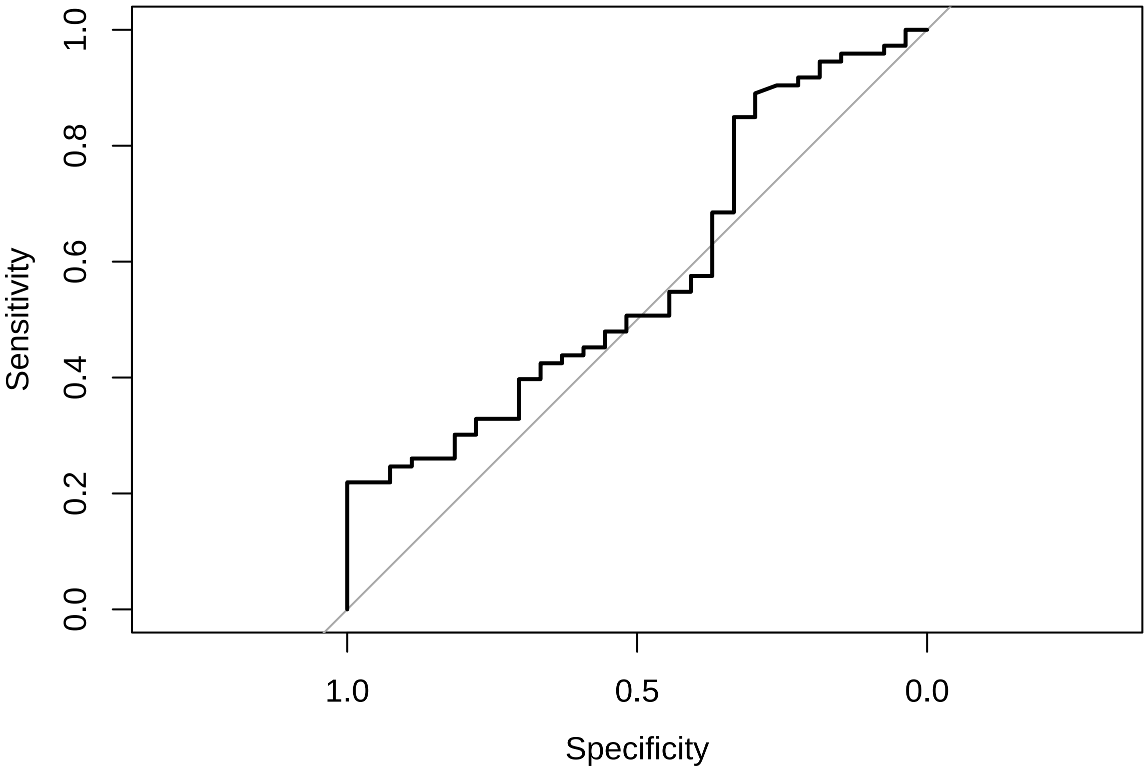
```
roc_session18 = roc(response = test_session18$feedback_type, predictor = test_session18_pred)
```

```
## Setting levels: control = -1, case = 1
## Setting direction: controls < cases
```

```
# plots roc curves of the two test sessions
plot(roc_session1)
```



```
plot(roc_session18)
```



```
# calculates area under the curve for the two roc curves
auc(roc_session1)
```

```
## Area under the curve: 0.8008
```

```
auc(roc_session18)
```

```
## Area under the curve: 0.5771
```

Our final results for the test sessions show that the models accuracy in predicting feedback outcome was 74% accurate with an area under the curve of 0.8008 for test session 1 and 73% accurate with an area under the curve of 0.5771. Despite the similarity in accuracy, the large discrepancy in AUC between the two test sessions shows that the model is an inconsistent form of predicting feedback outcome from session to session.

Discussion

Using neural data in across many different brain areas along with visual stimuli, we attempted to predict feedback outcomes based on correct or incorrect decision-making. Our analysis of these features revealed many key insights into their relationship with this decision-making process. In our exploratory analysis, it was determined that higher neural activity across the vast majority brain areas was generally associated with an increased number in positive feedback outcomes. This was concluded when comparing average neural spike counts in trials with a positive feedback vs trials with a negative feedback. One major challenge and potential area of further interest was the specific role each brain area played in determining these outcomes.

Because certain brain areas were used so inconsistently and number of neurons measured as a whole varied greatly from session to session, it proved extremely difficult to investigate each area individually across the whole data set. The other essential area of analysis presented was the difference in contrasts. We were able to determine that a greater difference between the left and right contrasts presented to each subject was correlated with a higher likelihood of correct decision-making being associated. A useful metric in our final model was the combined predictor of both average neural spikes and contrast difference that took both of these factors into account when predicting feedback outcome. Our final model performance proved to be somewhat effective on the data as a whole with an accuracy of 72% and an area under the curve of 0.7. However, when using the test sessions, the area under the curve showed a large inconsistency from test session 1 compared to test session 18. This showed that while the model was an effective predictor in session 1 it was not an effective predictor in session 18, revealing its potential inconsistency from session to session. In summary, using a combination of average neural spike counts and difference in contrasts proved to be somewhat effective in determining successful decision-making despite having some inconsistency across different sessions in the data. Looking into separating the individual roles of brain areas in further investigation could prove to be a more consistent and effective measure of feedback outcome through avoiding the reliance and oversimplification of using neural activity as a whole.

Reference

Steinmetz, N.A., Zatka-Haas, P., Carandini, M. et al. Distributed coding of choice, action and engagement across the mouse brain. Nature 576, 266–273 (2019). https://doi.org/10.1038/s41586-019-1787-x