Exploring the Relationship between Neural Activity and Feedback Types in Mouse Decision-Making

Riyaadh Bukhsh 921470997 STA141A

2023-06-10

Abstract:

This project aims to analyze a subset of data from experiments conducted on mice, specifically focusing on the neural activity in their visual cortex during decision-making tasks. The objective is to build a predictive model that can accurately predict the outcome of each trial based on the neural activity and stimuli information. Through exploratory data analysis, data integration, and model training, this project seeks to gain insights into the relationship between neural activity and decision-making in mice, ultimately contributing to a better understanding of the experimental data and potentially providing valuable insights for future studies in this field.

Introduction

This project focuses on the analysis of a subset of data collected by Steinmetz et al. (2019) from experiments conducted on mice. The study involved 10 mice over 39 sessions, where visual stimuli were presented to the mice, and they had to make decisions based on the stimuli. The neural activity in the mice's visual cortex was recorded in the form of spike trains during the trials.

The main objective of this project is to build a predictive model that can predict the outcome of each trial using the neural activity data and stimuli information. The project is divided into three parts.

In Part 1, I perform exploratory data analysis to understand the characteristics of the data set and explore the neural activities during the trials. I also investigate the changes across trials and examine the homogeneity and heterogeneity across sessions and mice.

In Part 2, I propose a data integration approach based on the findings from Part 1. This approach aims to combine data across trials by identifying shared patterns across sessions and addressing any differences between sessions. The goal is to enhance the prediction performance in the subsequent part.

In Part 3, I build a prediction model using the integrated data to predict the outcome of the trials. The model's performance will be evaluated on two test sets randomly selected from Session 1 and Session 18, respectively.

By conducting this analysis and building a predictive model, I aim to gain insights into the relationship between neural activity and decision-making in mice. The results obtained will contribute to a better understanding of the experimental data and potentially provide valuable insights for future studies in this field.

The original subset of data collected by Steinmetz encompasses various essential variables, providing a comprehensive picture of the neural activity within the visual cortex. The data consists of 18 sessions, with each session dedicated to a specific mouse and a distinct brain area within the visual cortex. For instance, session one focuses on neurons in areas ACA, CA3, DG, LS, MOs, SUB, VISp, and the root region.

Within each session, multiple trials are conducted, capturing crucial information related to decision-making. These trials include the following variables:

1. feedback_type: Indicates the type of feedback received, denoted as 1 for success and -1 for failure.
2. contrast_left: Represents the contrast level of the left stimulus presented during the trials.
3. contrast_right: Signifies the contrast level of the right stimulus presented during the trials.
4. time: Corresponds to the center points of the time bins used for organizing the neuron spikes.
5. spks: Reflects the number of spikes recorded for each neuron in the visual cortex, categorized into time bins defined by the 'time' variable.
6. brain_area: Identifies the specific brain area where each neuron is located within the visual cortex.

These variables collectively provide crucial insights into the neural dynamics and decision-making processes in the visual cortex. Analyzing this rich data set opens up avenues for understanding the intricate workings of the brain and its role in perception and decision-making.

Exploratory Analysis

To begin the data analysis process, the first step involves creating a comprehensive data frame that contains all the important information regarding the mouse data. At first, I will import the data into a list fully conscious about the drawbacks following, and although dealing with lists can present some challenges, I will overcome this limitation by transforming the list of sessions into a structured data frame.

By converting the data into a data frame format, I can effectively organize and manipulate the information, enabling us to perform in-depth analyses and extract valuable insights. This approach enhances the overall accessibility and usability of the data, facilitating further exploration and modeling tasks.

Aggregating the 18 sessions into a Comprehensive Data set

```
#Allocating session
session=list()

#Reading the session files into a list of 18 elements
for(i in 1:18){
  session[[i]]=readRDS(paste('../data/mouse_data/session',i,'.rds',sep=''))


}
```

Transforming sessions into a accessible data frame in order to simplify data manipulation

To ensure data integrity and maintain the distinct dimensions of spike data and general mouse information, I will refrain from combining them at this stage.

```
#Allocating mouse data frame
mouseData = data.frame()

#To iterate through the sessions
for(i in 1:18){

  #Temporary variable to store the allocated information for each session
```

```
  x = cbind(session[[i]]$contrast_left,session[[i]]$contrast_right,rep(i,length(session[[i]]$contrast_l

  #Binding the rows to the data frame
   mouseData = rbind(mouseData,x)

}



#Names of the data frame
colnames(mouseData) = c("contrast_left","contrast_right", "session","mouse","number_of_neurons","brain_a


##Checking the data frame
head(mouseData)
```

```
##   contrast_left contrast_right session mouse number_of_neurons brain_area
## 1             0            0.5       1  Cori               734          8
## 2             0              0       1  Cori               734          8
## 3           0.5              1       1  Cori               734          8
## 4             0              0       1  Cori               734          8
## 5             0              0       1  Cori               734          8
## 6             0              0       1  Cori               734          8
##   number_of_trials feedback_type
## 1              114             1
## 2              114             1
## 3              114            -1
## 4              114            -1
## 5              114            -1
## 6              114             1
```

```
# To check the total number of trials
totalTrials = 0

for(i in 1:18){


  num = length(session[[i]]$feedback_type)
  totalTrials = totalTrials + num
}

#Confirming the dimensions (rows) are equivalent to the number of total trials

dim(mouseData)
```

```
## [1] 5081    8
```

```
#Converting some data to factors for easier data analysis and manipulation
mouseData$session = as.factor(mouseData$session)
mouseData$mouse = as.factor(mouseData$mouse)
mouseData$feedback_type = as.factor(mouseData$feedback_type)
mouseData$brain_area = as.numeric(mouseData$brain_area)
```

```
head(mouseData)
```

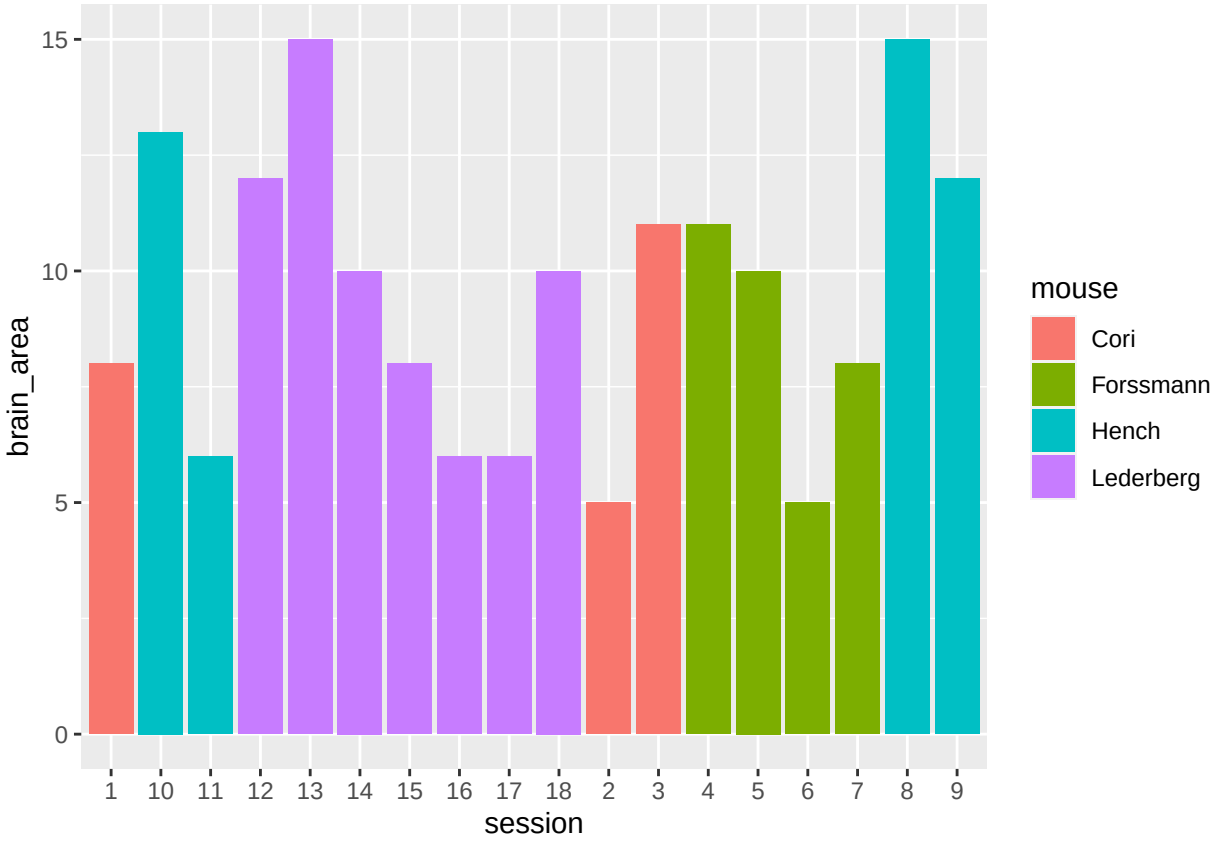
```
##   contrast_left contrast_right session mouse number_of_neurons brain_area
## 1             0            0.5       1  Cori               734          8
## 2             0              0       1  Cori               734          8
## 3           0.5              1       1  Cori               734          8
## 4             0              0       1  Cori               734          8
## 5             0              0       1  Cori               734          8
## 6             0              0       1  Cori               734          8
##   number_of_trials feedback_type
## 1              114             1
## 2              114             1
## 3              114            -1
## 4              114            -1
## 5              114            -1
## 6              114             1
```

Distribution of Measured Brain Areas

To better understand the coverage of brain areas in the data set, I examine the distribution of distinct brain areas that were measured during the experiments. This information is important for referencing the number of spikes per trial/session and provides insights into the spatial extent of the recorded neural activity.

```
mouseData %>% select(session,mouse,brain_area) %>% group_by(session,brain_area,mouse) %>% summarise("Nu
```

```
## `summarise()` has grouped output by 'session', 'brain_area'. You can override
## using the `.groups` argument.
```



Manipulate Data Function

Creating a function that aggregates the spike data in sessions and stores them into a data frame

The "manipulateData" function is designed to aggregate the spike data within each session and store it in a data frame. It takes a list (referred to as session[i]) as input and extracts the spike data while summing across the rows, excluding the time bin factor.

```
##Takes an input of a list (AKA session[[i]]), and extracts out the spike data  and sums across the row
manipulateData<-function(data,sessionNum){


  #Number of trials for each session
  trial_nums = NULL

  #Allocating variables
  brain_area<-data$brain_area
  spks<-cbind(brain_area,as.data.frame(sapply(data$spks,rowSums)))
  spks<-spks %>% group_by(brain_area) %>% summarise(across(everything(), sum))


  #Pivoting the data frame
  proper <- tidyr::pivot_longer(spks, cols = starts_with("V"), names_to = "Trial", values_to = "Spikes"


trial_numbers <- as.numeric(sub("V", "", grep("^V\\d+$", names(spks), value = TRUE)))
```

```
# Get the column names starting with "V" and extract the numeric part
trial_nums<-rep(trial_numbers,dim(proper %>% distinct(brain_area))[1])
proper$Trial<-c(trial_nums)




proper$session<-sessionNum

return(proper)

}
```

Creating the data frame for spikes

```
#Allocating the spike data frame
totalSpikeData<-NULL

for(i in 1:18){

  #Place holder for the spike data
  tempData <-manipulateData(session[[i]],i)


  #binding it to the data frame
  totalSpikeData<- rbind(totalSpikeData,tempData)
}


#Checking dimensions
dim(totalSpikeData)
```

```
## [1] 49173     4
```

```
#Confirming number of rows is correct for the newly created data frame
sum((mouseData %>% distinct(brain_area,number_of_trials)%>% pull(brain_area) %>% as.numeric())* (mouseDa
```

```
## [1] 49173
```

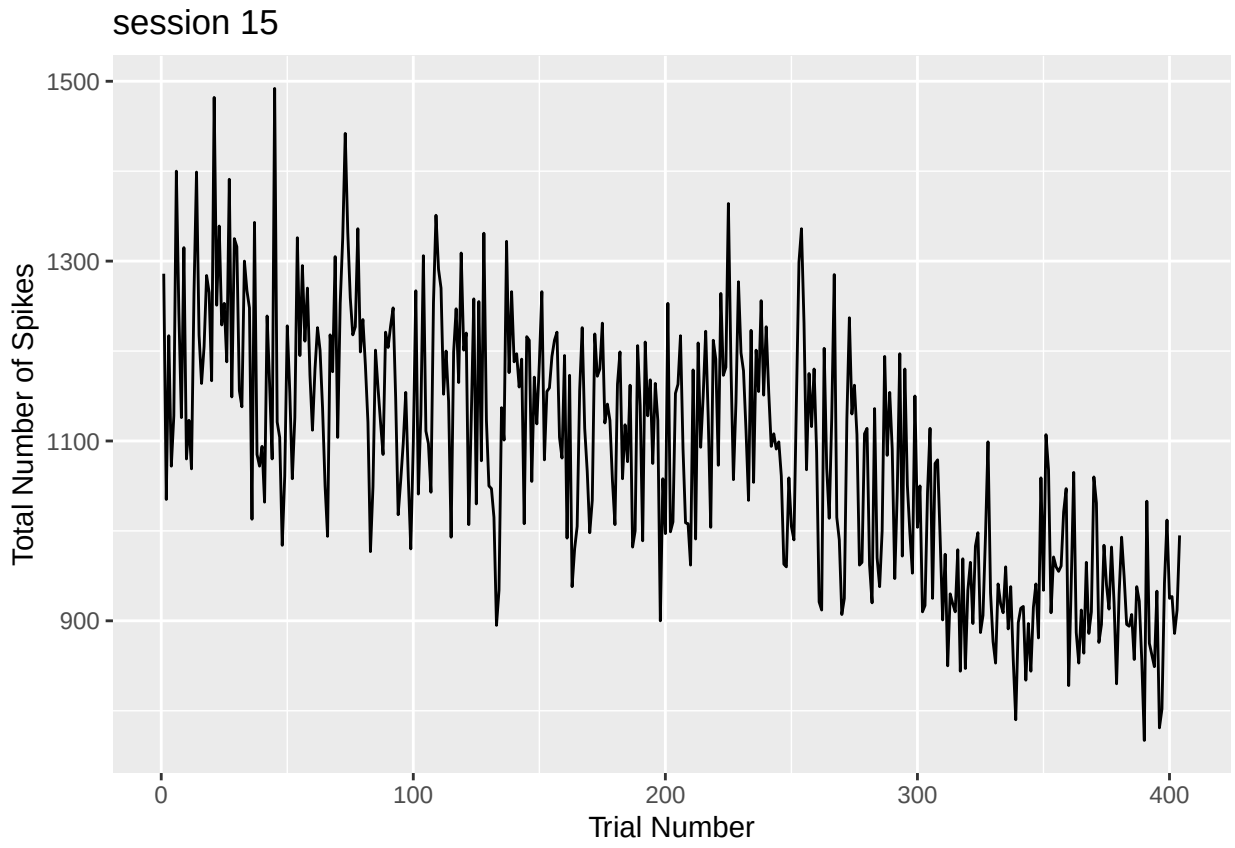
Visualizations for spike activity per session

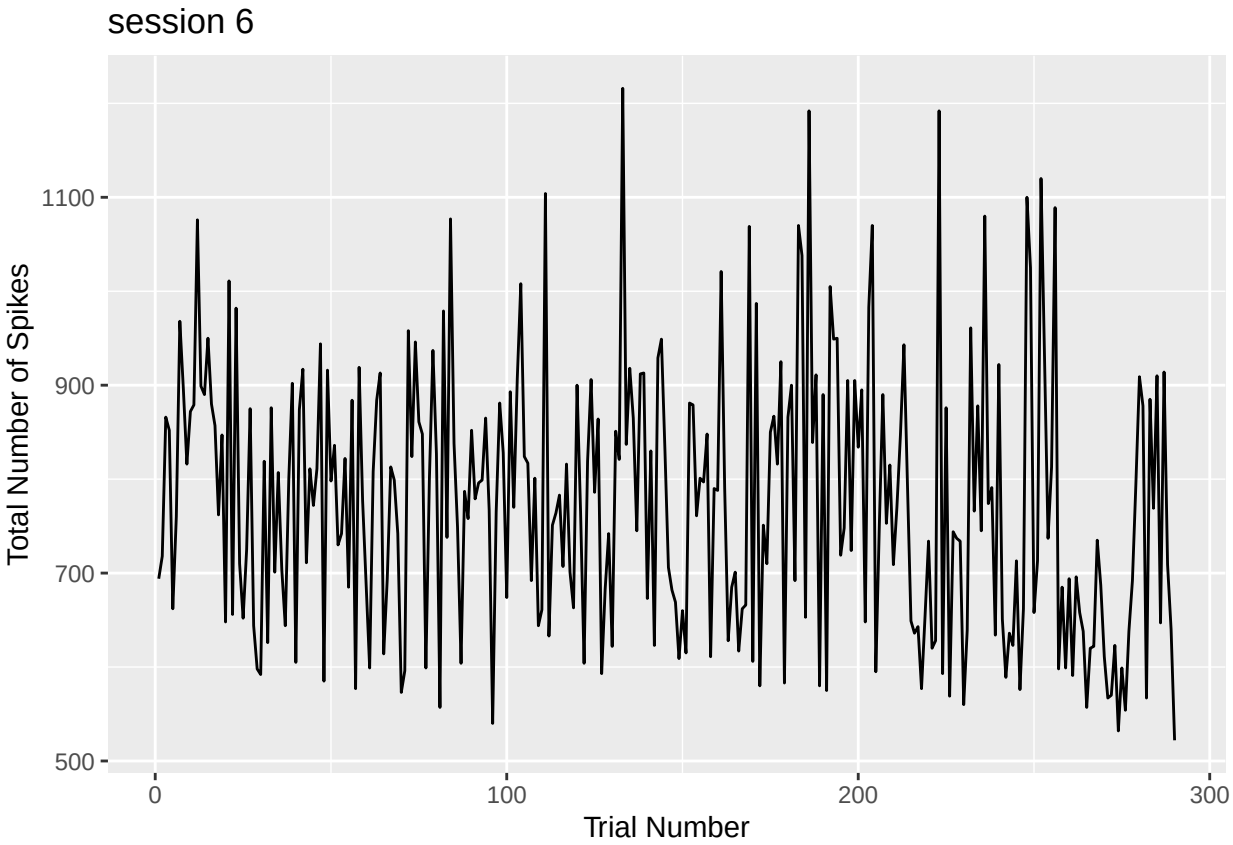
By grouping the total number of spikes in each trial and visualizing the data using a line graph, I can gain insights into the spike activity per session. This approach allows us to understand the ranges and variations in spike counts across the trials.

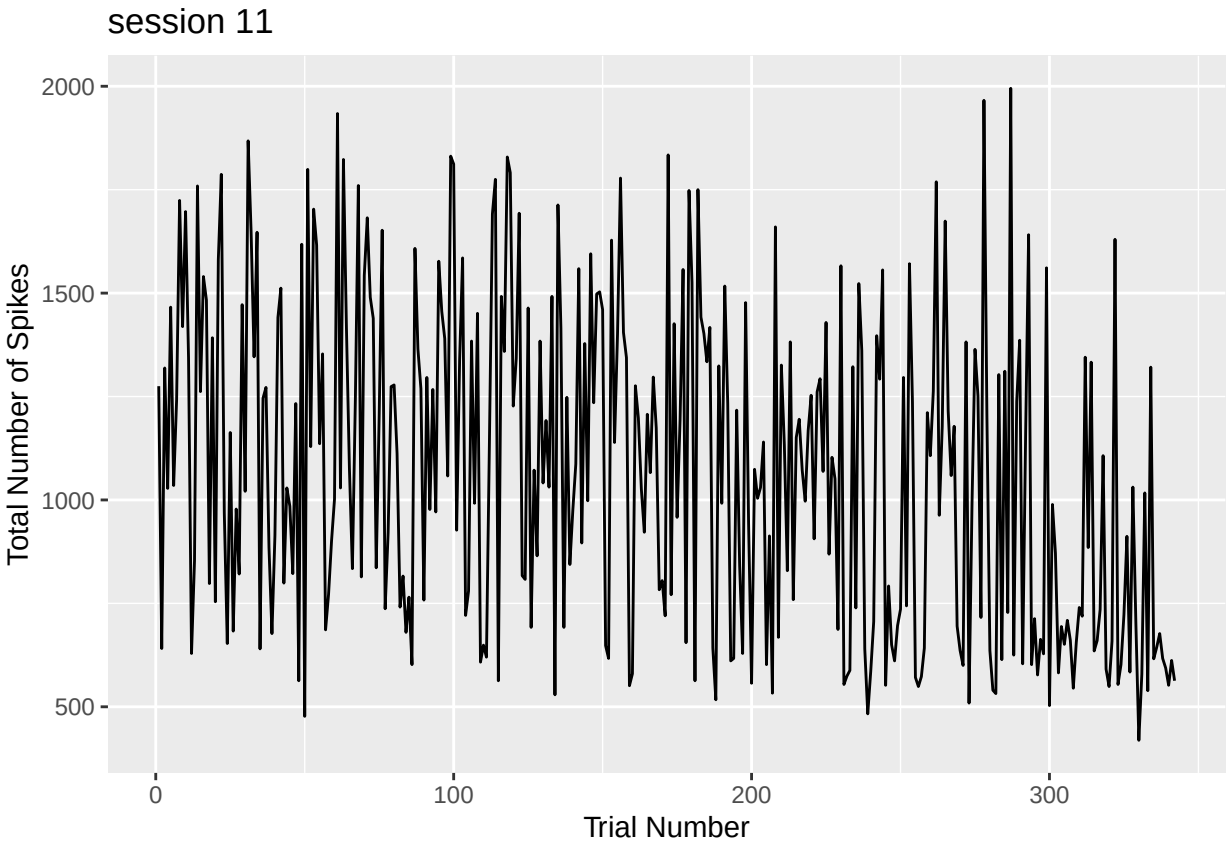
Upon examining the graph, I observe that some sessions have higher total spike counts compared to others. This difference could be attributed to factors such as the involvement of different mice or the application of more neuron readers to specific brain areas within the cortex. However, our main focus is on identifying differences in trends and fluctuations. Upon closer examination, I notice that some mice experience fatigue, leading to fluctuations in their total spike counts, while others maintain a more consistent average.

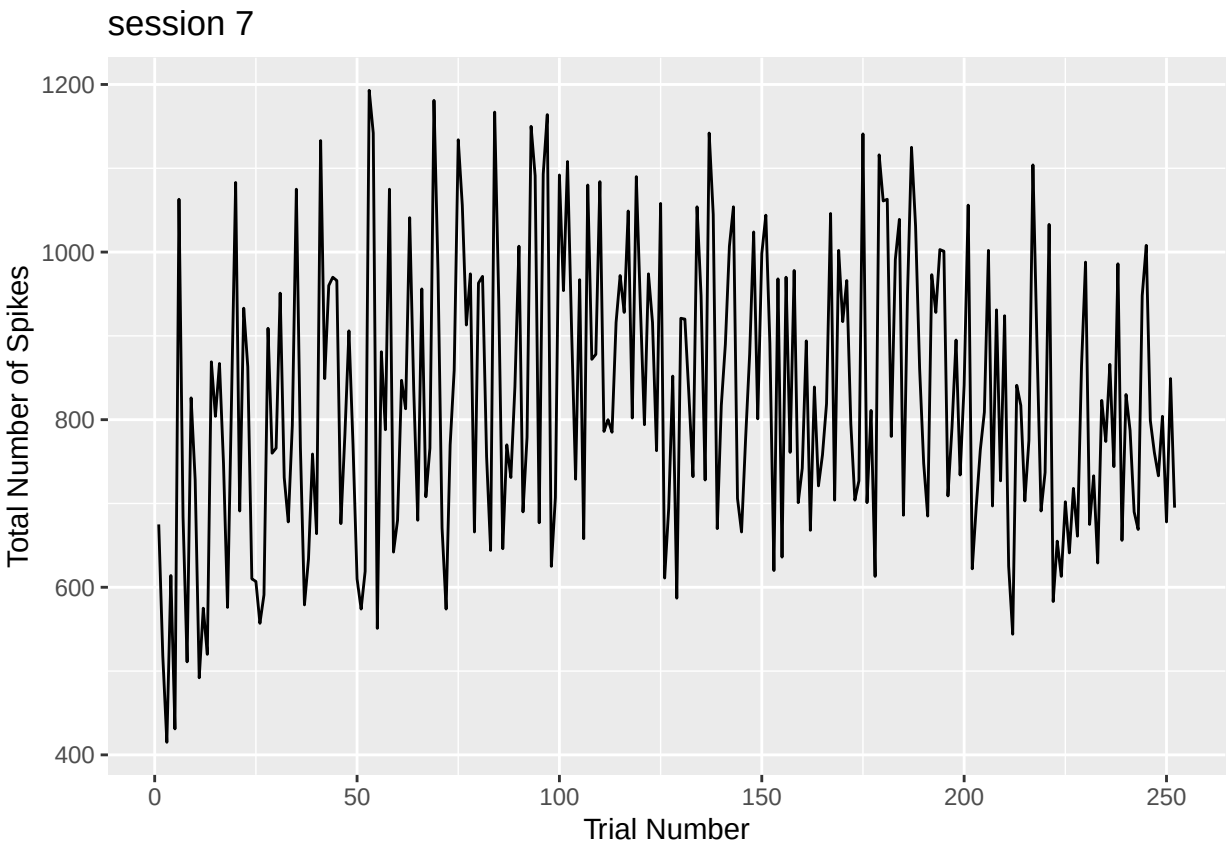
Despite the variations, I can observe that there are similarities in spike trends across sessions, suggesting common underlying patterns or dynamics in the neural activity.

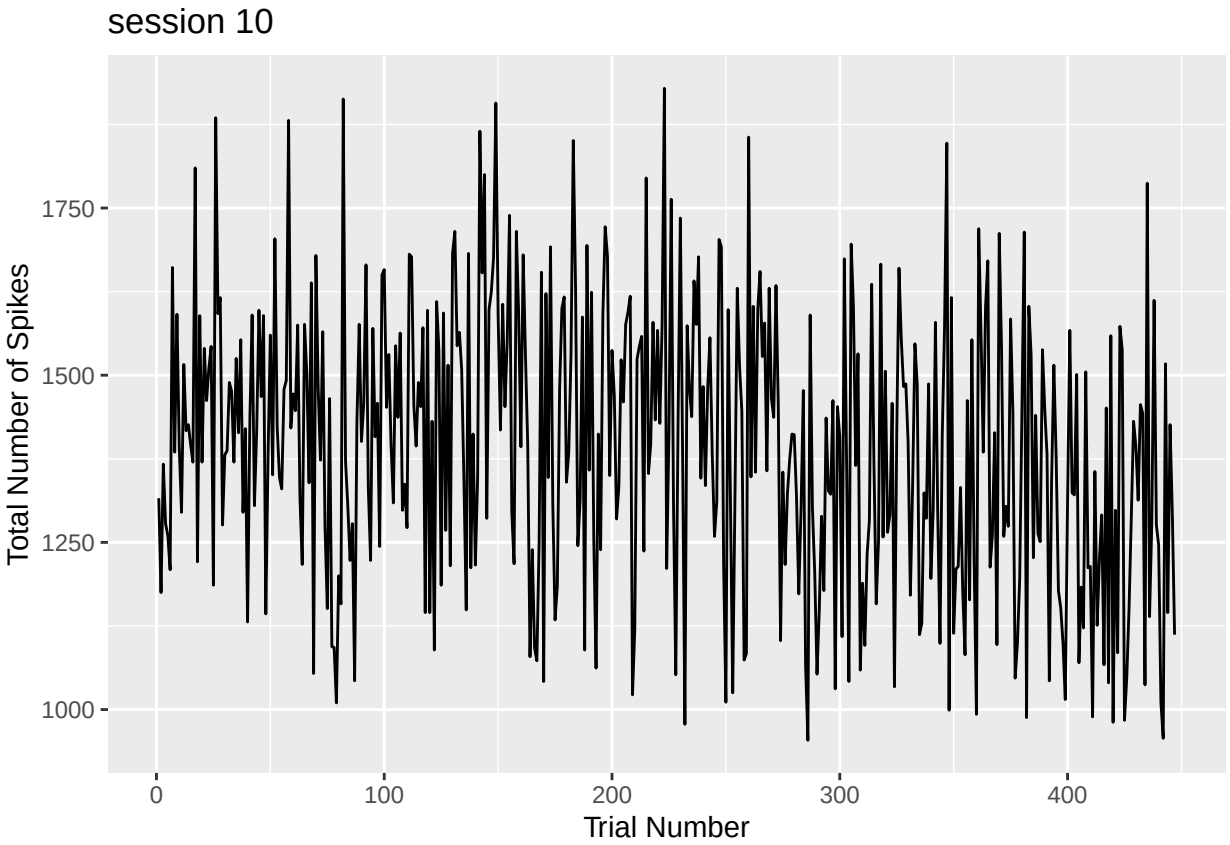
```r
#Session numbers for a random sampling method (removing bias)
sessionNumbers<-1:18


##Selecting Random sessions to plot and see association between number of spikes and Trial number
for(i in sample(sessionNumbers,6,replace = F)){

print(ggplot(totalSpikeData %>% filter(session == i) %>% group_by(Trial) %>% summarise(AverageSpikes = m
}
```

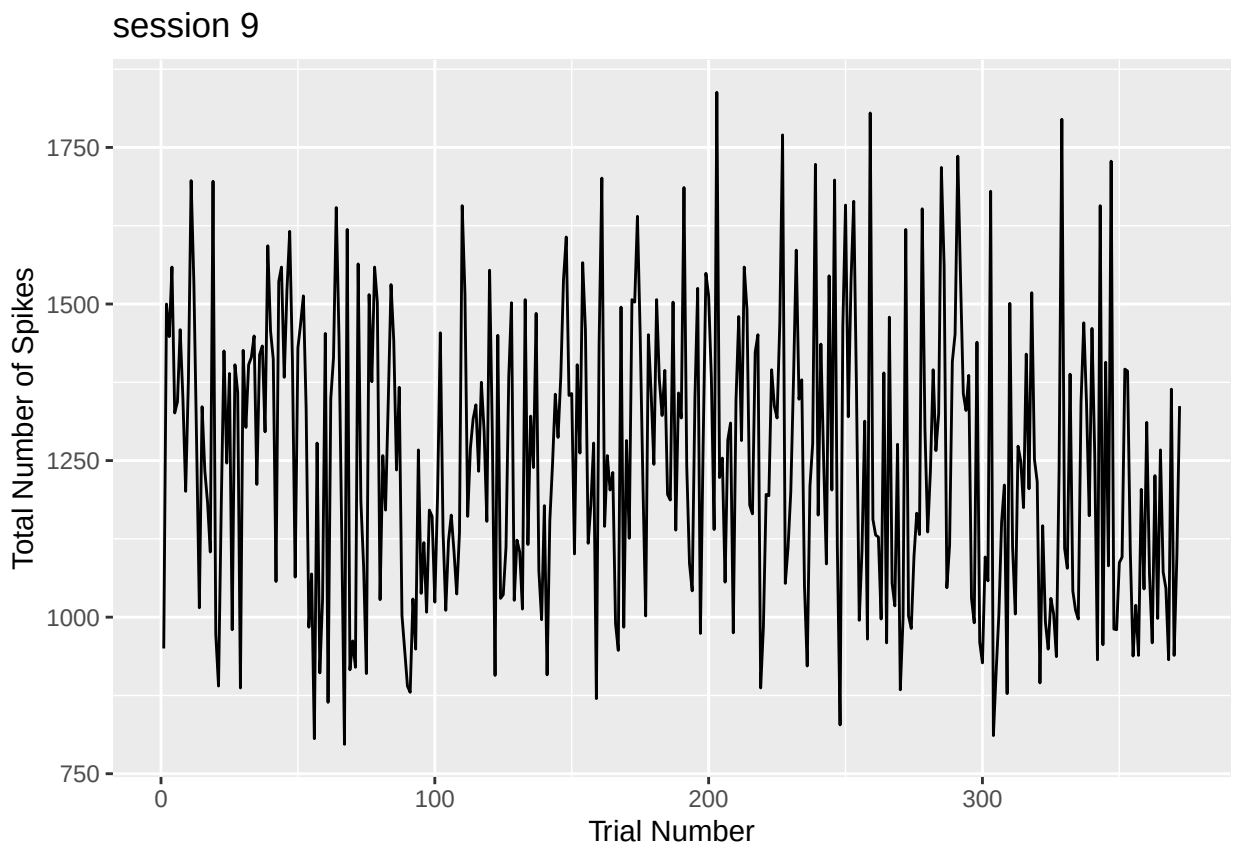



session 6
Total Number of Spikes
1100
900
700
500
0
100
200
300
Trial Number


session 11
2000
1500
1000
500
Total Number of Spikes
0
100
200
300
Trial Number


session 7
Total Number of Spikes
1200
1000
800
600
400
0
50
100
150
200
250
Trial Number


session 10
Total Number of Spikes
1750
1500
1250
1000
0
100
200
300
400
Trial Number



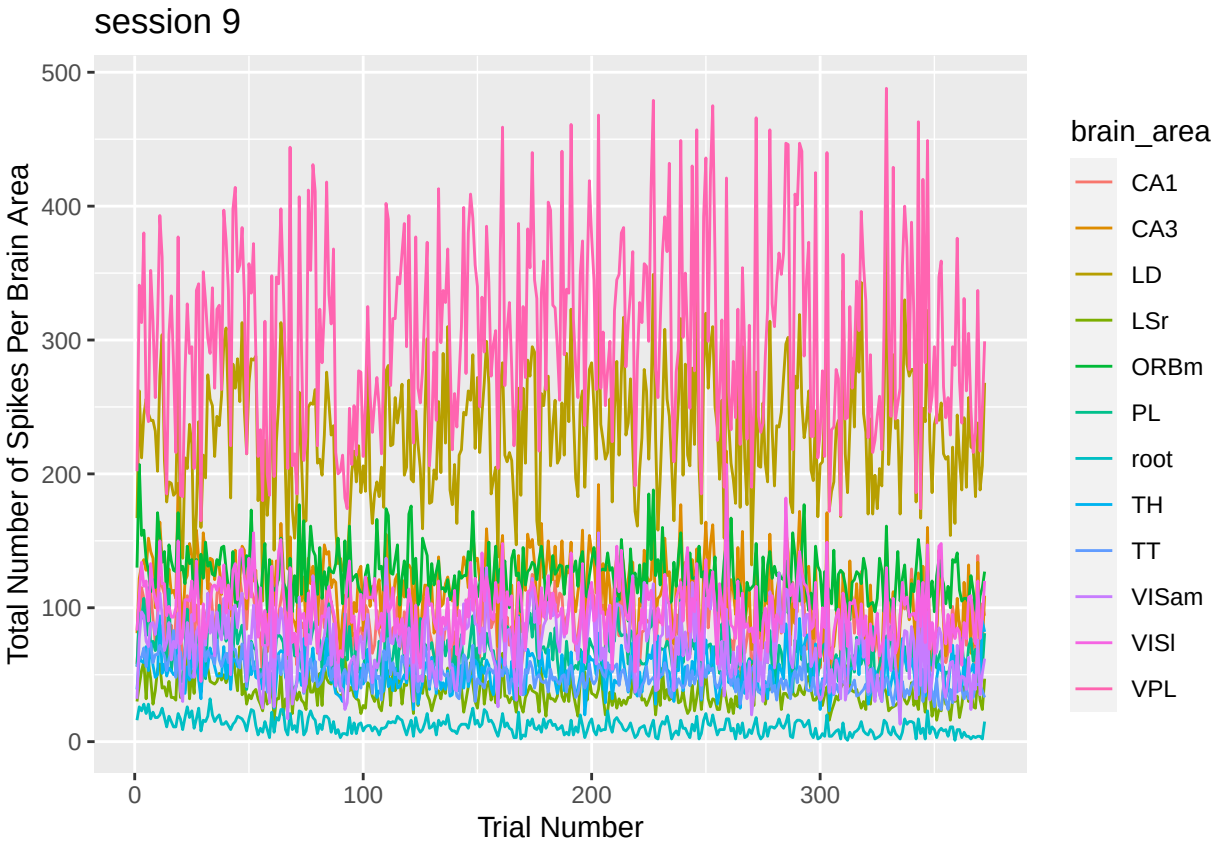
Analyzing Spike Activity Across Different Brain Areas

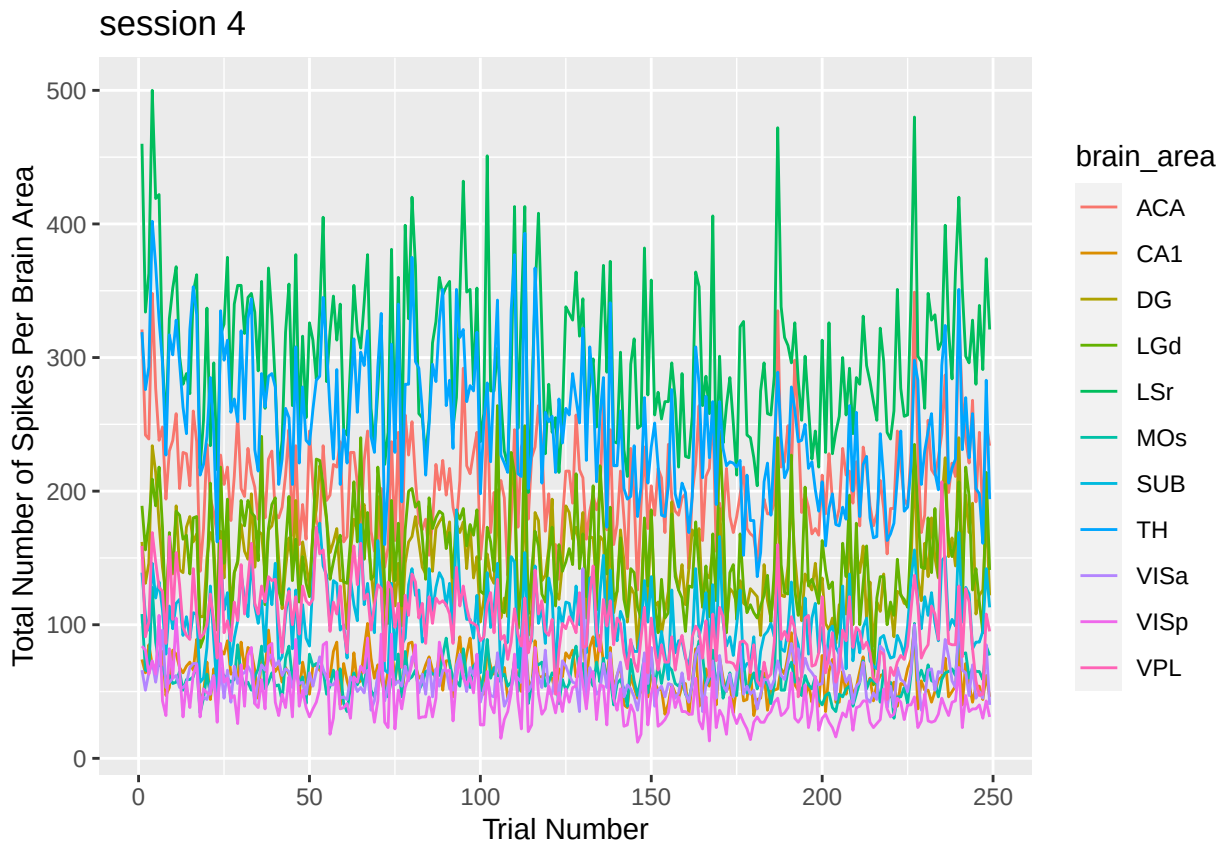
To explore the spike activity per brain area, I select arbitrary sessions to analyze the trends. This approach helps eliminate bias and allows us to observe patterns in neural activity across different brain areas.

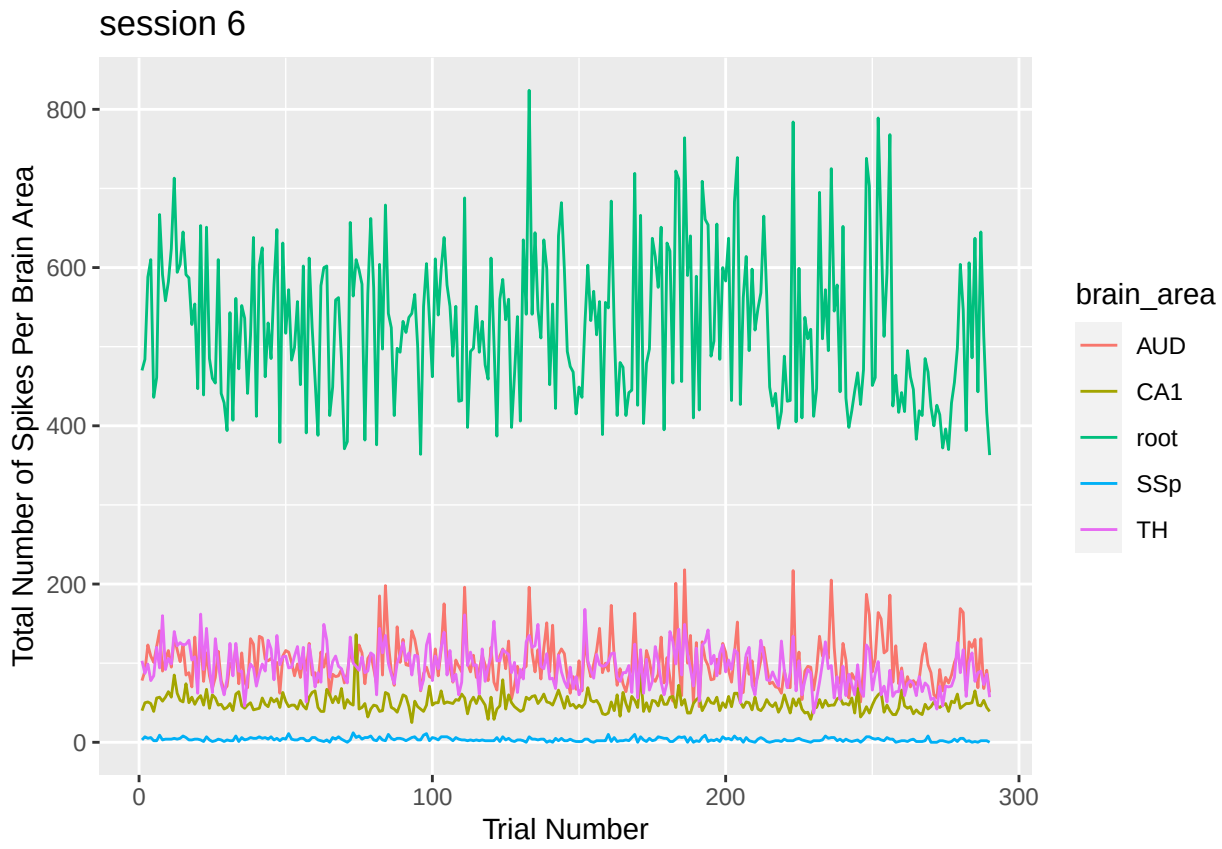
By examining the spike activity in these sessions, I can identify how neurons in specific brain areas respond to the stimuli presented during the trials. This analysis provides insights into the functional properties and information processing capabilities of different regions within the visual cortex.

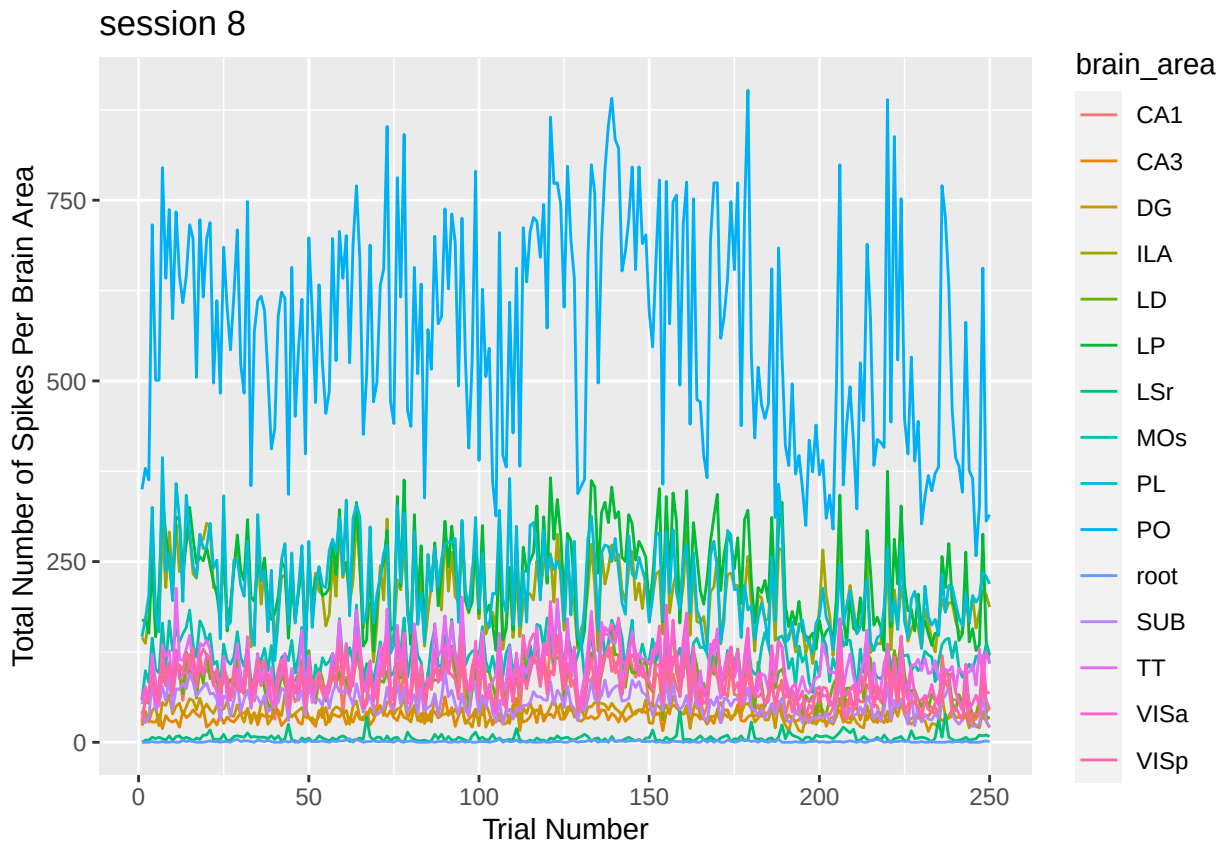
By considering multiple sessions and brain areas, I can gain a more comprehensive understanding of the neural dynamics and potentially uncover common patterns or variations in spike activity across different experimental conditions.

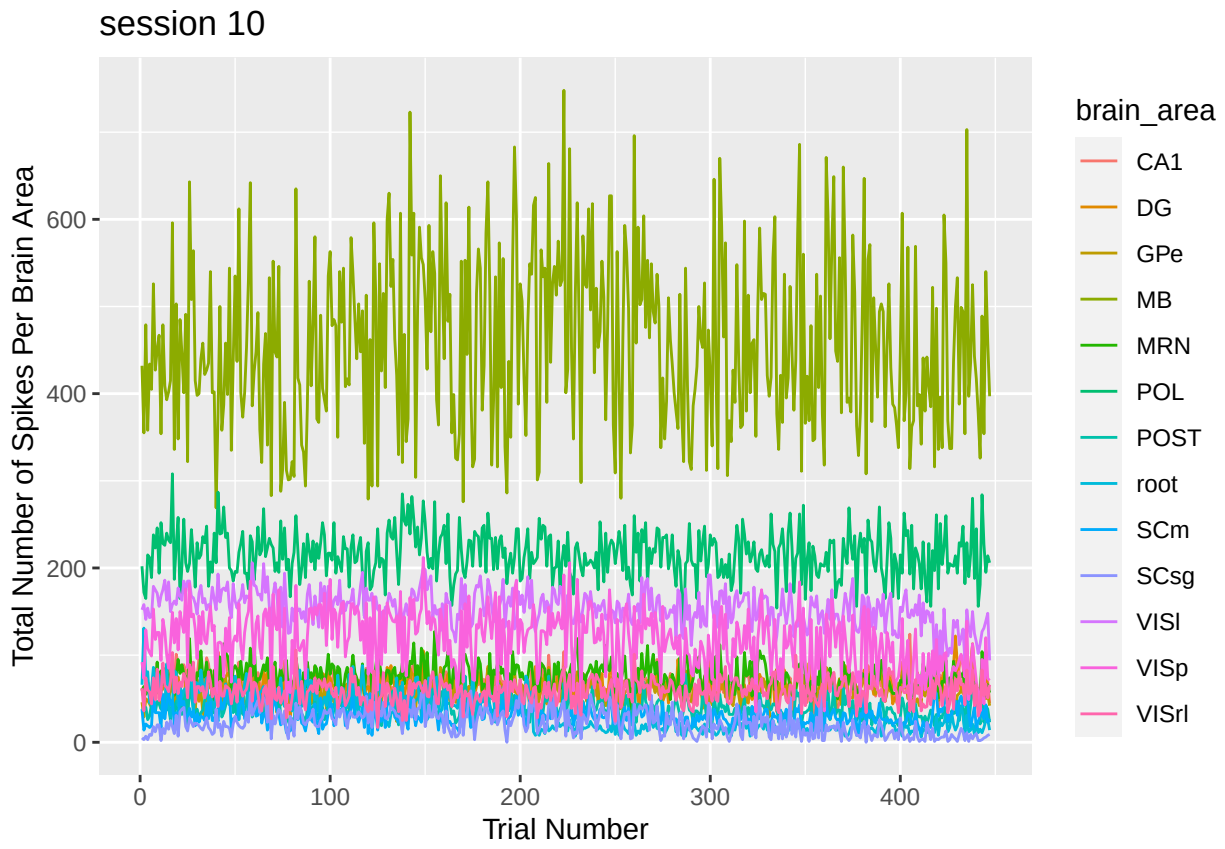
```
for(i in sample(sessionNumbers,5,replace = F)){

  print(ggplot(totalSpikeData %>% filter(session == i),aes(x = Trial,y = Spikes,color = brain_area))+ g

}
```


session 9
Total Number of Spikes Per Brain Area
500
400
300
200
100
0
0
100
200
300
Trial Number
brain_area
CA1
CA3
LD
LSr
ORBm
PL
root
TH
TT
VISam
VISl
VPL


session 4
Total Number of Spikes Per Brain Area
500
400
300
200
100
0
0
50
100
150
200
250
Trial Number
brain_area
ACA
CA1
DG
LGd
LSr
MOs
SUB
TH
VISa
VISp
VPL


session 6
Total Number of Spikes Per Brain Area
800
600
400
200
0
0
100
200
300
Trial Number
brain_area
AUD
CA1
root
SSp
TH


session 8
Total Number of Spikes Per Brain Area
750
500
250
0
0
50
100
150
200
250
Trial Number
brain_area
CA1
CA3
DG
ILA
LD
LP
LSr
MOs
PL
PO
root
SUB
TT
VISa
VISp



Data Integration

In the Data Integration phase, I leverage the insights gained from Part 1 to develop an approach for combining data across trials. The main objective is to extract shared patterns across sessions and address the differences between sessions, enabling us to borrow information and improve the prediction performance in Part 3.

To achieve this, I employ two strategies:

Extracting Shared Patterns: I identify common patterns and trends that exist across multiple sessions. By identifying these shared features, I can capture the underlying patterns that contribute to the neural activity during trials.

Addressing Session Differences: I take into account the variations between sessions, such as differences in brain areas or spike activity, feedback type, etc. By addressing these session-specific factors, I can account for potential biases and ensure a more robust and accurate prediction model.

I will now perform some data manipulation in order to combine the spike data with the mouse data

```
##Summarizing the spike data in order to merge it with the mouse data
summarisedSpikeData<- totalSpikeData %>%  group_by(session,Trial) %>% summarise(spikes = sum(Spikes))
```

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

```r
perfectMouseData<-cbind(mouseData,summarisedSpikeData[-1])



##Creating the perfect mouse data.
head(perfectMouseData,20)
```

```
##    contrast_left contrast_right session mouse number_of_neurons brain_area
## 1              0            0.5       1  Cori                734          8
## 2              0              0       1  Cori                734          8
## 3            0.5              1       1  Cori                734          8
## 4              0              0       1  Cori                734          8
## 5              0              0       1  Cori                734          8
## 6              0              0       1  Cori                734          8
## 7              1            0.5       1  Cori                734          8
## 8            0.5              0       1  Cori                734          8
## 9              0              0       1  Cori                734          8
## 10           0.5           0.25       1  Cori                734          8
## 11           0.5              0       1  Cori                734          8
## 12             0              1       1  Cori                734          8
## 13             1              1       1  Cori                734          8
## 14             0              0       1  Cori                734          8
## 15             0              0       1  Cori                734          8
## 16             0              0       1  Cori                734          8
## 17             0              0       1  Cori                734          8
## 18             0            0.5       1  Cori                734          8
## 19           0.5           0.25       1  Cori                734          8
## 20          0.25              1       1  Cori                734          8
##    number_of_trials feedback_type Trial spikes
## 1               114             1     1   1161
## 2               114             1     2    963
## 3               114            -1     3   1354
## 4               114            -1     4   1014
## 5               114            -1     5   1046
## 6               114             1     6    803
## 7               114             1     7   1543
## 8               114             1     8   1251
## 9               114             1     9   1108
## 10              114             1    10   1271
## 11              114             1    11   1104
## 12              114             1    12   1398
## 13              114            -1    13   1295
## 14              114            -1    14   1071
## 15              114            -1    15   1265
## 16              114            -1    16   1019
## 17              114             1    17   1001
## 18              114             1    18   1508
## 19              114             1    19   1410
## 20              114             1    20   1545
```

To analyze the distribution of feedback types (success or failure) across sessions and identify any consistent patterns or trends, I utilized bar charts. These charts visually represented the distribution of feedback types across sessions, providing insights into the relationship between sessions and feedback outcomes.

I generated two bar charts for this analysis. The first chart displayed the absolute counts of success and failure without any scaling. This allowed me to observe the raw differences in the total success-to-failure ratio across sessions. By examining this chart, I could identify sessions with notable variations in the distribution of feedback types.

To gain a better understanding of the proportional differences between sessions, I created a second bar chart with scaling. This chart allowed me to compare the proportions of success and failure across sessions, normalizing the data and enabling a more direct comparison. By visualizing the scaled proportions, I could identify any consistent patterns or trends in the distribution of feedback types across sessions.

Shared Patterns Across Sessions

```
##Total counts data set
feedback_counts <- perfectMouseData %>%
  group_by(session, feedback_type) %>%
  summarise(count = n()) %>% group_by(session) %>% mutate(totals = sum(count))
```

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

```
#Feedback proportions data set
feedback_proportions <- perfectMouseData %>%
  group_by(session, feedback_type) %>%
  summarise(count = n()) %>% mutate(percentage = count/sum(count))
```

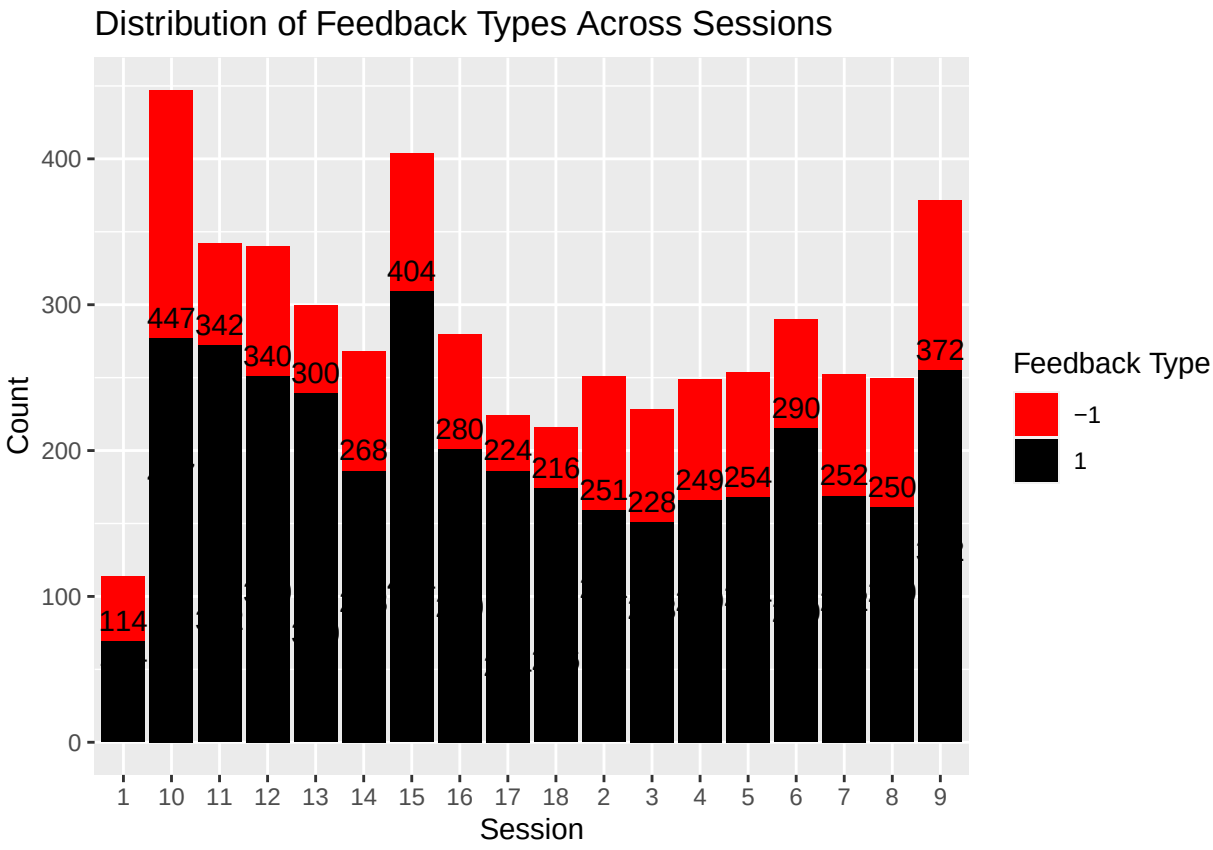
```
## `summarise()` has grouped output by 'session'. You can override using the
## `.groups` argument.
```

```
##Total counts of feedback
total_counts = feedback_counts %>% group_by(session) %>% summarise(total = sum(count))

##Average feedback for success
average_rate <- feedback_proportions %>%filter(feedback_type ==1) %>% pull(percentage) %>% mean()



feedback_counts
```

```
## # A tibble: 36 x 4
## # Groups:   session [18]
##    session feedback_type count totals
##    <fct>   <fct>         <int>  <int>
##  1 1       -1               45    114
##  2 1       1                69    114
##  3 10      -1              170    447
##  4 10      1               277    447
##  5 11      -1               70    342
##  6 11      1               272    342
##  7 12      -1               89    340
##  8 12      1               251    340
##  9 13      -1               61    300
```

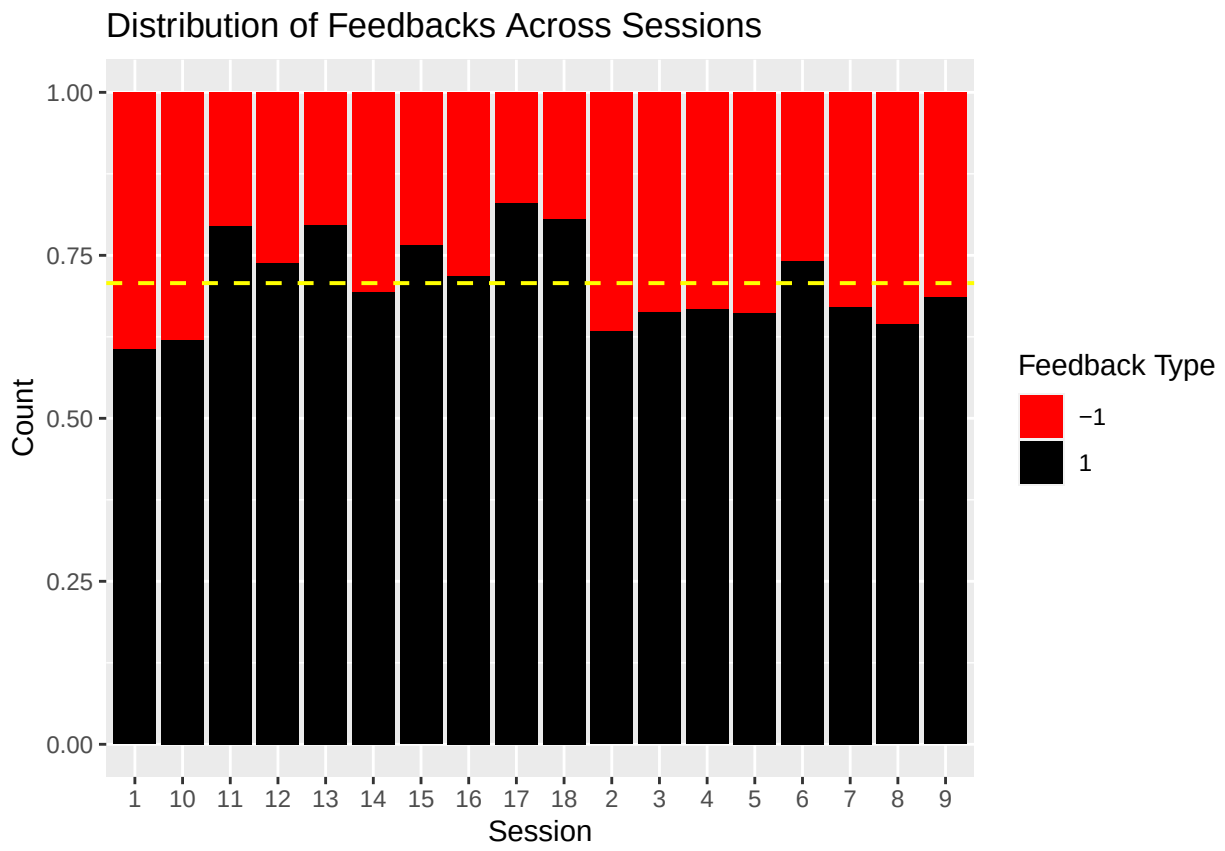
```
## 10 13      1               239     300
## # i 26 more rows
```

```
ggplot(feedback_counts, aes(x = session, y = count, fill = feedback_type)) +
  geom_bar(stat = "identity") + geom_text(aes(label = totals),vjust = -.5,color = "black")+ scale_fill_
```




```
ggplot(feedback_proportions, aes(x = session, y = percentage, fill = feedback_type)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("1" = "black", "-1" = "red")) + geom_hline(yintercept = average_rate, li
  labs(x = "Session", y = "Count", fill = "Feedback Type") +
  ggtitle("Distribution of Feedbacks Across Sessions")
```

```
## 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.
```



Exploring the Relationship Between Contrast Levels and Feedback Type

In this analysis, I delve into the relationship between the contrast levels (contrast_left and contrast_right) and the feedback type across sessions. Our aim is to investigate whether certain combinations of contrast levels consistently result in success or failure.

By examining the data from various sessions, I can identify patterns and trends that shed light on the influence of contrast levels on the feedback outcomes. Specifically, I assess how different combinations of contrast levels are associated with the feedback types, which are categorized as success (1) or failure (-1).

This exploration provides valuable insights into the relationship between visual stimuli (represented by contrast levels) and the resulting feedback. By understanding the consistent associations between specific contrast combinations and success/failure outcomes, I can gain a deeper understanding of the decision-making process of the mice during the trials.

Upon analyzing the data, I observe a significant portion of trials with contrast levels of (0, 0), indicating the absence of visual stimuli. Within this contrast level, I notice a higher occurrence of success feedback compared to failures. This finding suggests that the mice demonstrate an understanding of the absence of stimuli and refrain from moving the wheel, resulting in successful outcomes. Other contrast levels seem to be distributed fairly.

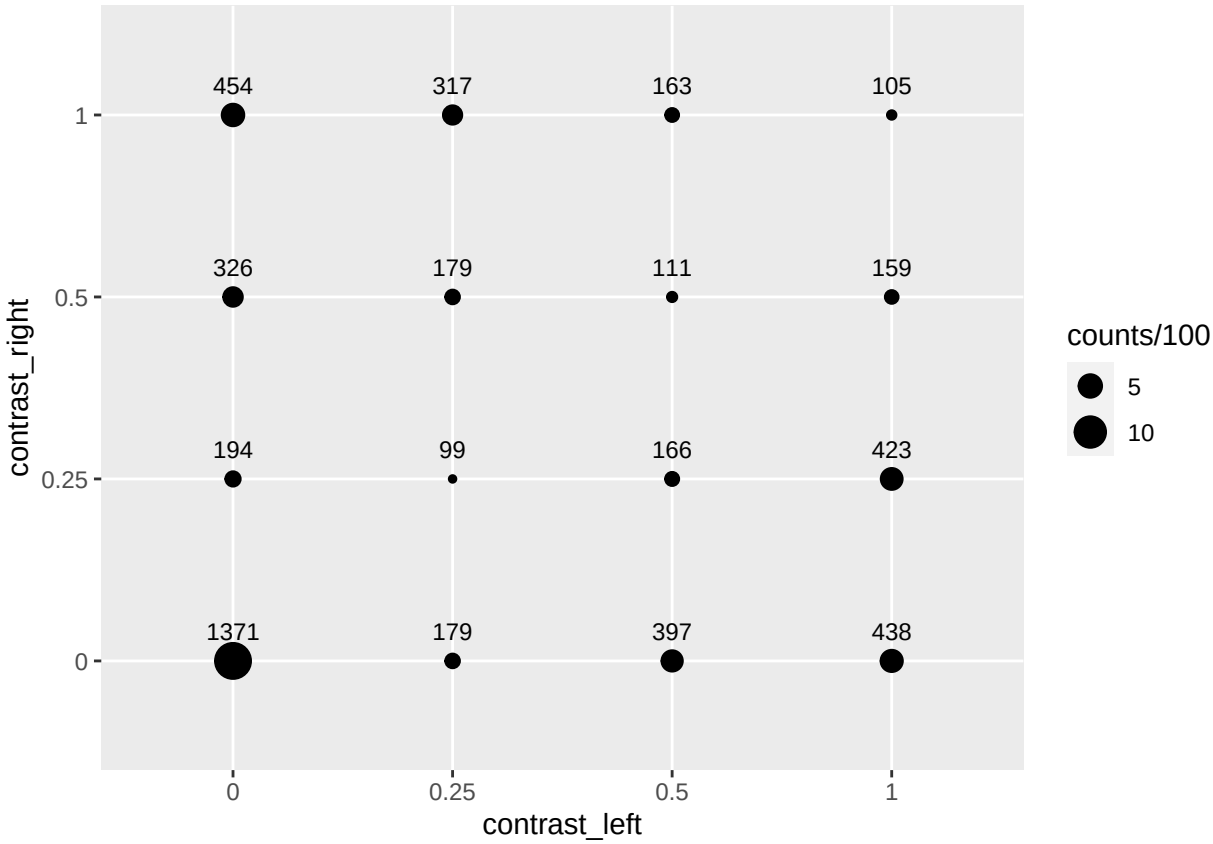
```
##Distribution of contrast varieties across all sessions
contrastLevels <-  perfectMouseData %>% group_by(contrast_left,contrast_right) %>% summarize(counts = n
```

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

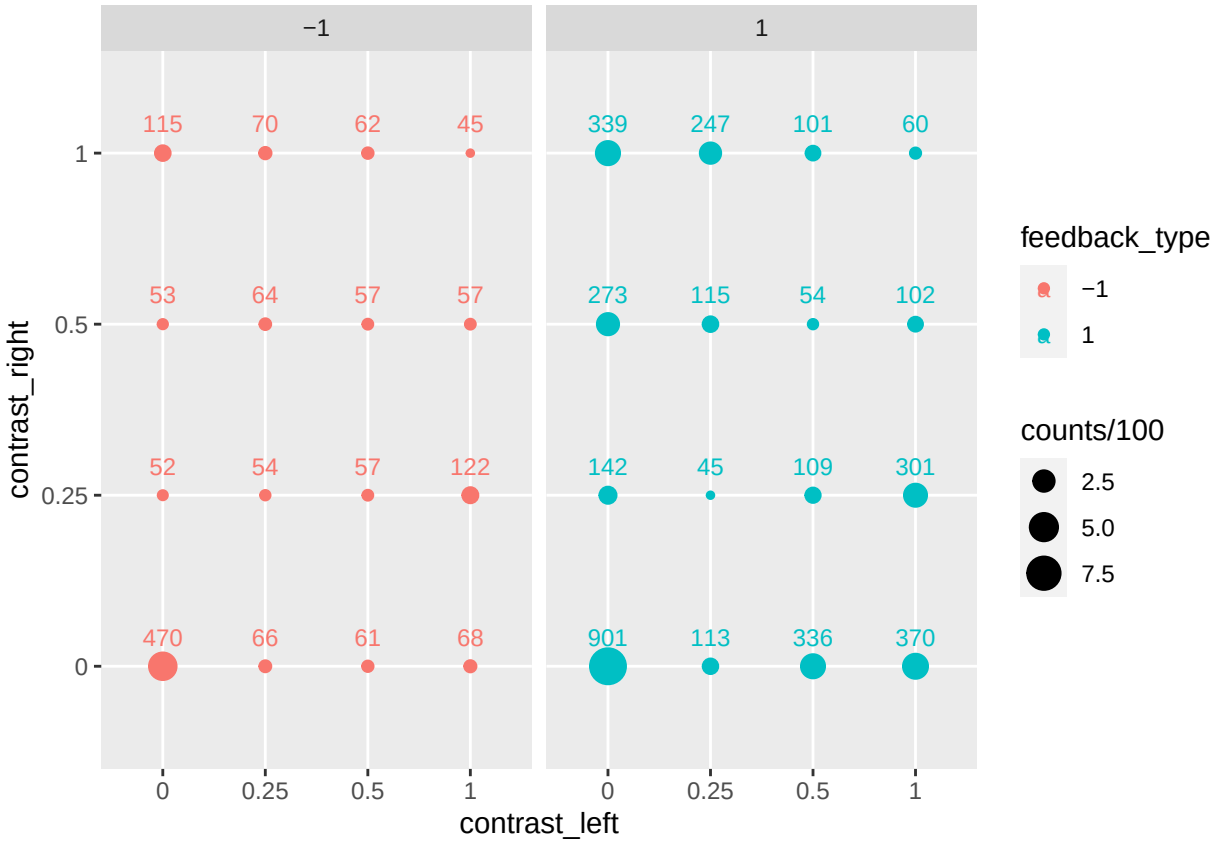
```
##Distribution of contrast varieties across all sessions with feedback
contrastLevelsByFeedback = perfectMouseData %>% group_by(contrast_left,contrast_right,feedback_type) %>%
```

```
## `summarise()` has grouped output by 'contrast_left', 'contrast_right'. You can
## override using the `.groups` argument.
```

```
##Plots

##Counts of various Contrast Levels for trials
ggplot(contrastLevels, aes(x = contrast_left, y = contrast_right,size = counts/100,label = counts)) +
  geom_point()+geom_text(size = 3, vjust = -1.2)
```




```
## Counts of trials depending on Contrast Levels faceted by feedback type
ggplot(contrastLevelsByFeedback, aes(x = contrast_left, y = contrast_right,size = counts/100,label = cou
  geom_point()+geom_text(size = 3, vjust = -1.2)+facet_grid(~feedback_type)
```



Feedback Success Efficiency Between Mice and Sessions

In this analysis, I examine the feedback success efficiency across different mice and sessions. By investigating the variations in success rates, I can identify potential differences in performance among mice and sessions.

Upon examining the data, I observe that Lederberg exhibits a higher success rates compared to others. This variation suggests that individual mice may have different cognitive abilities or learning strategies, leading to variations in their decision-making and overall success in the trials.

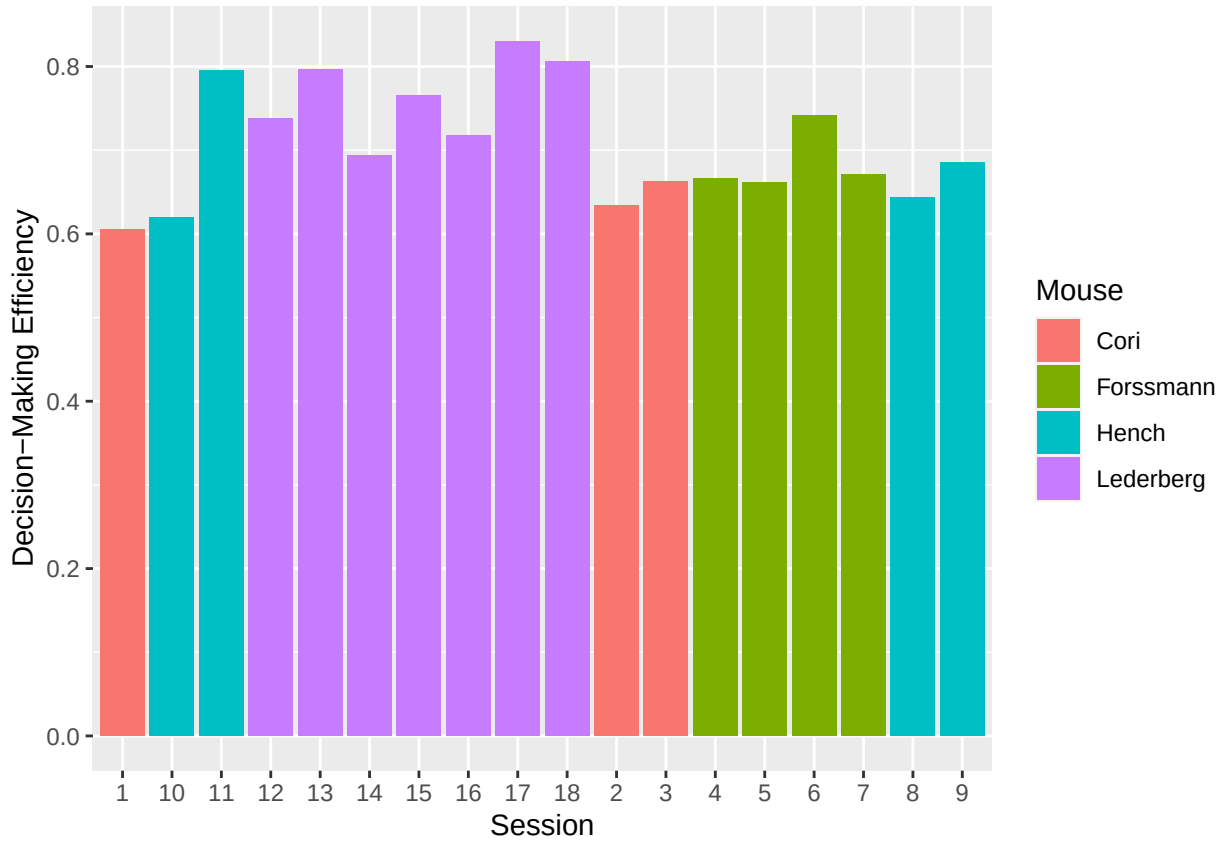
```
perfectMouseData %>% group_by(mouse) %>%summarise(DescisionMaking = sum(feedback_type == 1)/n())
```

```
## # A tibble: 4 x 2
##   mouse     DescisionMaking
##   <fct>               <dbl>
## 1 Cori                0.639
## 2 Forssmann           0.687
## 3 Hench               0.684
## 4 Lederberg           0.761
```

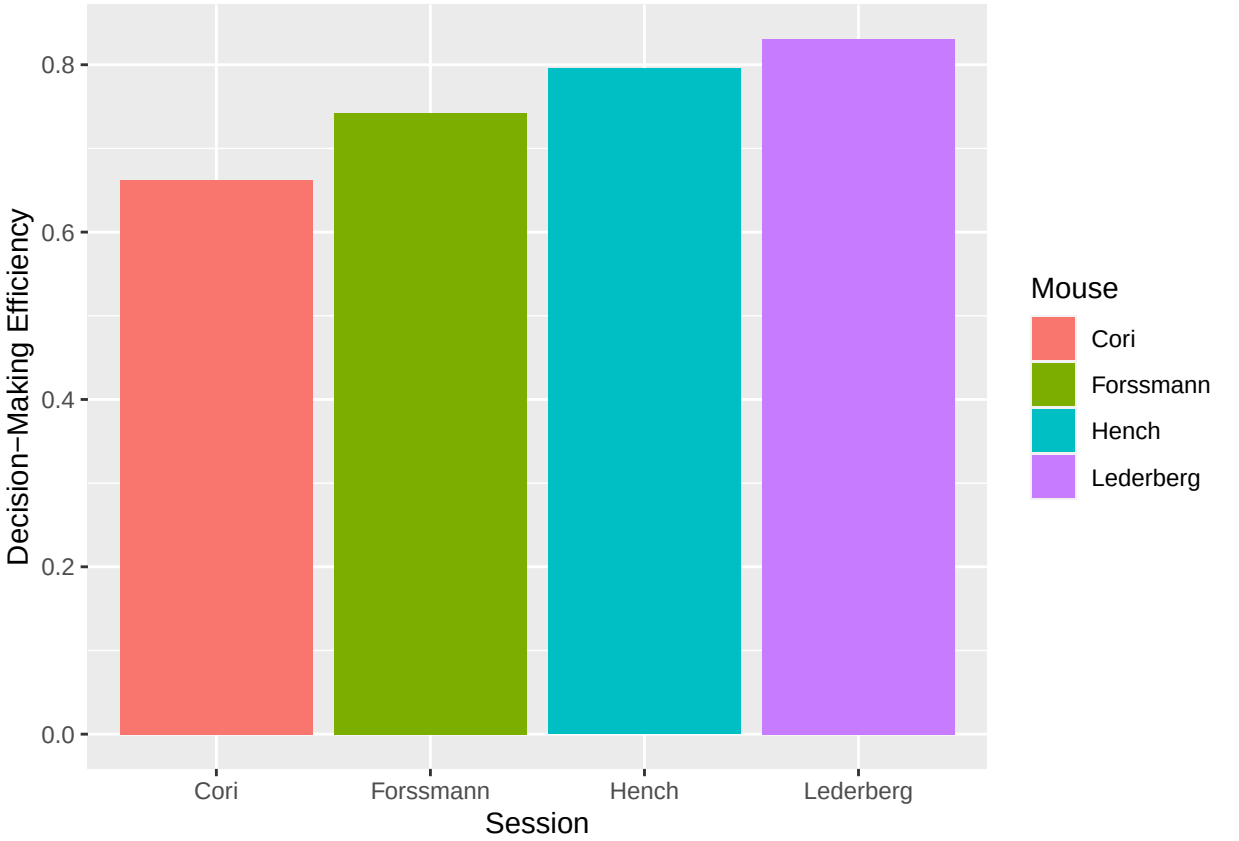
```
test = perfectMouseData %>% group_by(mouse,session) %>%summarise(DescisionMakingEfficiency = sum(feedba
```

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

```
ggplot(test, aes(x = session, y = DescisionMakingEfficiency, fill = mouse)) +geom_bar(stat = "identity"
```




```
ggplot(test, aes(x = mouse, y = DescisionMakingEfficiency, fill = mouse)) +geom_bar(stat = "identity", p
```



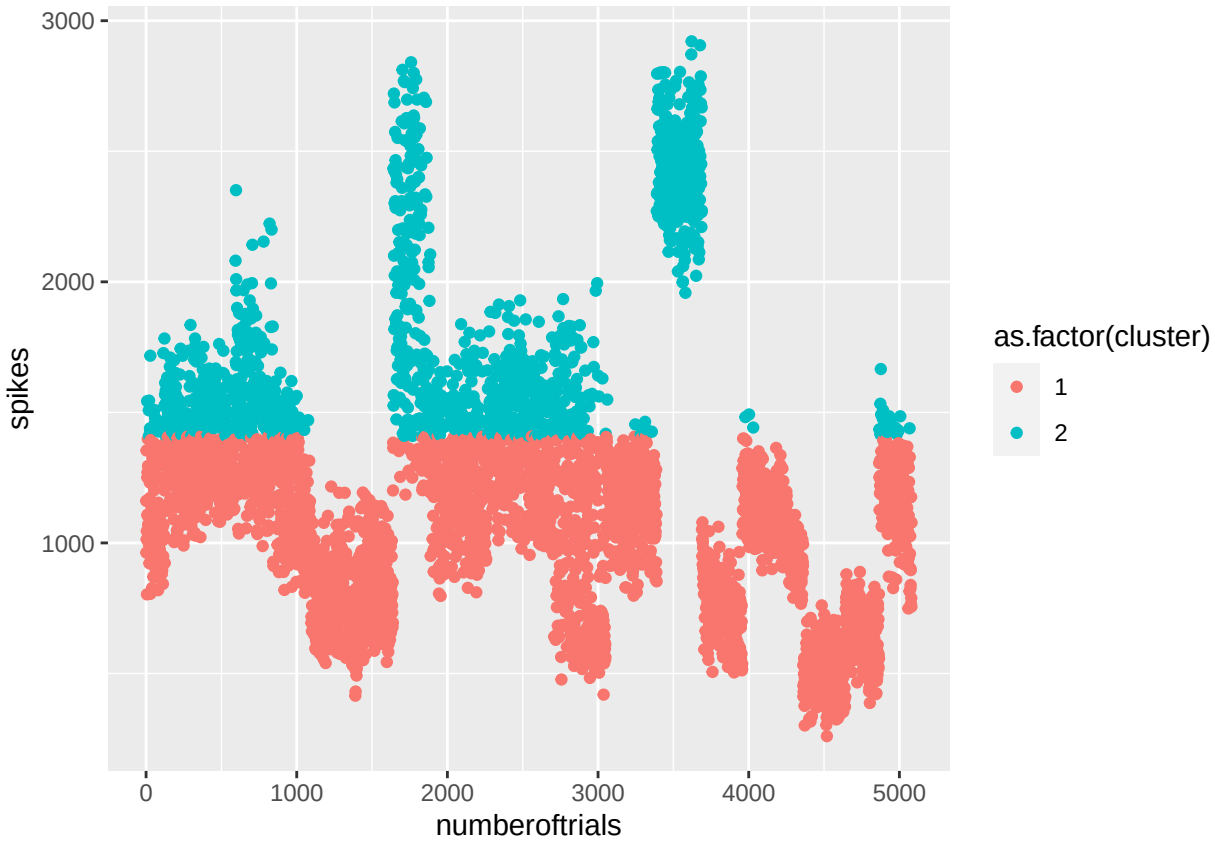
Clustering with k = 2

To gain further insights into the relationship between feedback types, contrast levels, and the total number of spikes, I perform clustering analysis on the data. By grouping similar data points together, clustering allows us to identify patterns and potential correlations among these variables.

In our analysis, I use the k-means algorithm with k=2 to represent the two feedback types. By clustering the data points based on the contrast levels and the total number of spikes, I aim to identify any discernible patterns or associations between these variables and the feedback types.

Upon examining the clustering results, I can observe potential patterns emerging. This suggests that there may be correlations between the feedback types and the contrast levels as well as the total number of spikes. It is important to note that further analysis and statistical testing are necessary to validate these assumptions and explore the strength and significance of these relationships.

```
clusterData <-NULL
clusterDF<- perfectMouseData %>% select(contrast_left,contrast_right,spikes)

clusterData<- kmeans(clusterDF,2)

numberoftrials<-1:5081

clusterDF %>% mutate(cluster = clusterData$cluster) %>%
  ggplot(aes(y=spikes, x=numberoftrials, color = as.factor(cluster))) +
  geom_point()
```



Predictive Modeling

The choice of logistic regression as the preferred method for prediction modeling in this project is driven by several key factors.

Firstly, logistic regression is well-suited for binary classification tasks, making it an appropriate approach for predicting the feedback types (success or failure) for each trial. By estimating the probability of a specific feedback type based on the given predictors, logistic regression enables us to make informed predictions.

Additionally, logistic regression offers the advantage of interpret ability. The coefficients associated with each predictor provide insights into the influence and direction of their effects on the outcome. This interpret ability enhances our understanding of the relationship between the neural activity data and the feedback types.

Moreover, logistic regression is effective in handling moderate-sized data sets. Given our 18 sessions and a subset of trials, logistic regression can deliver reliable predictions without imposing excessive computational demands. Its robustness to outliers and ability to accommodate col-linearity between predictors further contribute to its suitability for our analysis.

```
#Changing the variables to factors for the predictive model
perfectMouseData$contrast_left<-as.factor(as.character(perfectMouseData$contrast_left))
perfectMouseData$contrast_right<-as.factor(as.character(perfectMouseData$contrast_right))


##Filtering the data to train by
trainData<-perfectMouseData %>% filter(session %in% 2:17)
```

```r
testData<-perfectMouseData %>% filter(session %in% c(1,18))



print('Train Data:\n')
```

```
## [1] "Train Data:\n"
```

```r
head(trainData)
```

```
##   contrast_left contrast_right session mouse number_of_neurons brain_area
## 1             1              1       2  Cori              1070          5
## 2          0.25              0       2  Cori              1070          5
## 3           0.5            0.5       2  Cori              1070          5
## 4          0.25              0       2  Cori              1070          5
## 5             0              0       2  Cori              1070          5
## 6             0              0       2  Cori              1070          5
##   number_of_trials feedback_type Trial spikes
## 1              251            -1     1   1727
## 2              251             1     2   1071
## 3              251            -1     3   1343
## 4              251             1     4   1378
## 5              251            -1     5   1326
## 6              251             1     6   1029
```

```r
print('Test Data: \n')
```

```
## [1] "Test Data: \n"
```

```r
head(testData)
```

```
##   contrast_left contrast_right session mouse number_of_neurons brain_area
## 1             0            0.5       1  Cori               734          8
## 2             0              0       1  Cori               734          8
## 3           0.5              1       1  Cori               734          8
## 4             0              0       1  Cori               734          8
## 5             0              0       1  Cori               734          8
## 6             0              0       1  Cori               734          8
##   number_of_trials feedback_type Trial spikes
## 1              114             1     1   1161
## 2              114             1     2    963
## 3              114            -1     3   1354
## 4              114            -1     4   1014
## 5              114            -1     5   1046
## 6              114             1     6    803
```

```r
summary(trainData)
```

```
##  contrast_left contrast_right    session           mouse
##  0   :2191     0   :2232      10     : 447   Cori     : 479
```

```
##  0.25: 711     0.25: 836      15     : 404   Forssmann:1045
##  0.5 : 785     0.5 : 719      9      : 372   Hench    :1411
##  1   :1064     1   : 964      11     : 342   Lederberg:1816
##                               12     : 340
##                               13     : 300
##                               (Other):2546
##  number_of_neurons    brain_area     number_of_trials   feedback_type
##  Length:4751        Min.   : 5.000   Length:4751        -1:1386
##  Class :character   1st Qu.: 6.000   Class :character   1 :3365
##  Mode  :character   Median :10.000   Mode  :character
##                     Mean   : 9.703
##                     3rd Qu.:12.000
##                     Max.   :15.000
##
##      Trial           spikes
##  Min.   :  1.0   Min.   : 260.0
##  1st Qu.: 75.0   1st Qu.: 846.5
##  Median :149.0   Median :1168.0
##  Mean   :155.8   Mean   :1209.6
##  3rd Qu.:223.0   3rd Qu.:1441.5
##  Max.   :447.0   Max.   :2921.0
##
```

```
## The prediction model, using logistic regression
model = glm(feedback_type~contrast_left+contrast_right+spikes,family = 'binomial',trainData)


summary(model)
```

```
##
## Call:
## glm(formula = feedback_type ~ contrast_left + contrast_right +
##     spikes, family = "binomial", data = trainData)
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         5.312e-01  9.092e-02   5.843 5.13e-09 ***
## contrast_left0.25  -6.889e-02  9.667e-02  -0.713 0.476071
## contrast_left0.5    9.175e-02  9.296e-02   0.987 0.323660
## contrast_left1      2.467e-01  8.908e-02   2.769 0.005622 **
## contrast_right0.25 -3.368e-01  9.244e-02  -3.643 0.000269 ***
## contrast_right0.5  -1.095e-01  9.620e-02  -1.139 0.254903
## contrast_right1     1.191e-03  8.964e-02   0.013 0.989397
## spikes              3.150e-04  6.675e-05   4.719 2.37e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5736.3  on 4750  degrees of freedom
## Residual deviance: 5692.2  on 4743  degrees of freedom
## AIC: 5708.2
##
## Number of Fisher Scoring iterations: 4
```

```r
predictions<-as.data.frame(predict(model, testData))
predictionsFact<- as.factor(ifelse(predictions>0.8,1,-1))
```

```r
matrix<- confusionMatrix(testData$feedback,predictionsFact)
```

```r
matrix$table
```

```
##           Reference
## Prediction  -1   1
##         -1  32  55
##         1   44 199
```

```r
##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))
```

```
## [1] 0.3
```

Based on our evaluation, the prediction model demonstrates promising performance, yielding a relatively low misclassification rate of 0.27. This indicates that the model is able to accurately predict the feedback types for the given trials in the data set.

Prediction Performance On Test Sets

```r
testData=list()

#Reading the session files into a list of 18 elements
for(i in 1:2){
  testData[[i]]=readRDS(paste('../data/mouse_data/test',i,'.rds',sep=''))


}
```

```r
testMouseData = data.frame()

#To iterate through the sessions
for(i in 1:2){

  #Temporary variable to store the allocated information for each session
  x = cbind(testData[[i]]$contrast_left,testData[[i]]$contrast_right,rep(i,length(testData[[i]]$contras

  #Binding the rows to the data frame
   testMouseData = rbind(testMouseData,x)
```

```
}
#Names of the data frame
colnames(testMouseData) = c("contrast_left","contrast_right", "session","mouse","number_of_neurons","bra


##Checking the data frame
head(testMouseData)
```

```
##   contrast_left contrast_right session mouse number_of_neurons brain_area
## 1          0.25           0.25       1  Cori               734          8
## 2             0              0       1  Cori               734          8
## 3             1            0.5       1  Cori               734          8
## 4             0            0.5       1  Cori               734          8
## 5             1            0.5       1  Cori               734          8
## 6           0.5           0.25       1  Cori               734          8
##   number_of_trials feedback_type
## 1              100            -1
## 2              100             1
## 3              100             1
## 4              100             1
## 5              100             1
## 6              100             1
```

```
##Extracting the spike data

testSpikeData<-NULL

for(i in 1:2){

  #Place holder for the spike data
  tempData <-manipulateData(testData[[i]],i)


  #binding it to the data frame
  testSpikeData<- rbind(testSpikeData,tempData)
}

head(testSpikeData)
```

```
## # A tibble: 6 x 4
##   brain_area Trial Spikes session
##   <chr>      <dbl>  <dbl>   <int>
## 1 ACA            1     98       1
## 2 ACA            2     72       1
## 3 ACA            3    145       1
## 4 ACA            4    161       1
## 5 ACA            5     77       1
## 6 ACA            6     96       1
```

```
##Summarizing the spike data in order to merge it with the mouse data
summarisedSpikeData<- testSpikeData %>%  group_by(session,Trial) %>% summarise(spikes = sum(Spikes))
```

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

```
finalTestMouseData<-cbind(testMouseData,summarisedSpikeData[-1])

##Creating the perfect mouse data.
head(finalTestMouseData,20)
```

```
##    contrast_left contrast_right session mouse number_of_neurons brain_area
## 1           0.25           0.25       1  Cori               734          8
## 2              0              0       1  Cori               734          8
## 3              1            0.5       1  Cori               734          8
## 4              0            0.5       1  Cori               734          8
## 5              1            0.5       1  Cori               734          8
## 6            0.5           0.25       1  Cori               734          8
## 7              0              1       1  Cori               734          8
## 8              1              0       1  Cori               734          8
## 9           0.25            0.5       1  Cori               734          8
## 10             0            0.5       1  Cori               734          8
## 11             0              0       1  Cori               734          8
## 12             0              1       1  Cori               734          8
## 13             0              1       1  Cori               734          8
## 14             0            0.5       1  Cori               734          8
## 15          0.25              1       1  Cori               734          8
## 16          0.25              1       1  Cori               734          8
## 17          0.25              1       1  Cori               734          8
## 18             0              1       1  Cori               734          8
## 19          0.25              1       1  Cori               734          8
## 20           0.5              0       1  Cori               734          8
##    number_of_trials feedback_type Trial spikes
## 1               100            -1     1   1052
## 2               100             1     2   1142
## 3               100             1     3   1383
## 4               100             1     4   1344
## 5               100             1     5   1165
## 6               100             1     6   1005
## 7               100             1     7    904
## 8               100            -1     8    968
## 9               100            -1     9   1001
## 10              100             1    10   1439
## 11              100             1    11    940
## 12              100             1    12   1101
## 13              100             1    13   1350
## 14              100             1    14    976
## 15              100            -1    15   1093
## 16              100            -1    16    995
## 17              100             1    17   1070
## 18              100            -1    18    967
## 19              100            -1    19   1070
## 20              100             1    20   1314
```

I am splitting the data into two test sessions, to evaluate the accuracy of the prediction model.

Test Data 1

```
## This experiment is for test data 1
## The prediction model, using logistic regression
##From the trained data
summary(model)
```

```
##
## Call:
## glm(formula = feedback_type ~ contrast_left + contrast_right +
##     spikes, family = "binomial", data = trainData)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          5.312e-01  9.092e-02   5.843 5.13e-09 ***
## contrast_left0.25   -6.889e-02  9.667e-02  -0.713 0.476071
## contrast_left0.5     9.175e-02  9.296e-02   0.987 0.323660
## contrast_left1       2.467e-01  8.908e-02   2.769 0.005622 **
## contrast_right0.25  -3.368e-01  9.244e-02  -3.643 0.000269 ***
## contrast_right0.5   -1.095e-01  9.620e-02  -1.139 0.254903
## contrast_right1      1.191e-03  8.964e-02   0.013 0.989397
## spikes               3.150e-04  6.675e-05   4.719 2.37e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5736.3  on 4750  degrees of freedom
## Residual deviance: 5692.2  on 4743  degrees of freedom
## AIC: 5708.2
##
## Number of Fisher Scoring iterations: 4
```

```
testDataPredictions<-as.data.frame(predict(model, (finalTestMouseData %>% filter(session == 1))))
predictionsFact<- as.factor(ifelse(testDataPredictions>0.5,1,-1))


predictionsFact
```

```
##   [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
##  [26] 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [51] 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [76] 1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  -1
## Levels: -1 1
```

```
testData1 <-finalTestMouseData %>%  filter(session == 1)

matrix<- confusionMatrix(as.factor(testData1$feedback),predictionsFact)


matrix$table
```

```
##           Reference
## Prediction -1  1
```

```
##          -1  2 26
##          1   0 72
```

```
##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))
```

```
## [1] 0.26
```

Test Data 2

```
## This experiment is for test data 2

testDataPredictions<-as.data.frame(predict(model, (finalTestMouseData %>% filter(session == 2))))
predictionsFact<- as.factor(ifelse(testDataPredictions>0.5,1,-1))


predictionsFact
```

```
##   [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
##  [26] 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
##  [51] 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
##  [76] 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
## Levels: -1 1
```

```
testData2 <-finalTestMouseData %>%  filter(session == 2)


matrix<- confusionMatrix(as.factor(testData2$feedback_type),predictionsFact)


matrix$table
```

```
##           Reference
## Prediction -1  1
##          -1  0 27
##          1   1 72
```

```
##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))
```

```
## [1] 0.28
```

The results revealed that the model performed well, achieving an accuracy rate of 74% on the first test session and 72% on the second test session. These findings indicate that the model is capable of effectively predicting the feedback type, demonstrating its robustness and generalization across different data sets

Although my model has shown accurate predictions for the spikes in the given data, it is important to note that this performance is limited to the specific session data used for training and testing. Generalization to other independent sessions may require additional information or further refinement of the model. It is crucial to consider the potential limitations and the need for validation on diverse data sets to ensure the reliability and applicability of the predictive model beyond the current experiment.

Discussion

In this project, I conducted an analysis of a subset of data collected by Steinmetz et al. (2019). My primary objective was to build a predictive model to determine the feedback type of each trial using neural activity data and stimulus contrast levels.

In Section 1, I explored the general information about the mouse data and converted the sessions into a dataframe for easier access. Additionally, I examined the spike data per trial but decided not to combine it with the general mouse data due to differences in dimensions until after.

In Section 2, I visualized the spike activity per session, grouping the total number of spikes in each trial and displaying them through line graphs. This provided insights into the ranges and variations in spike counts across trials. I observed that spike trends were similar across sessions, although some sessions had higher total spikes, potentially due to differences in mice or brain areas. I also ventured into the metrics of mouse feedback effectiveness, measuring the proportions of success to failures across each mouse.

In Section 3, I explored the spike activity per brain area, selecting arbitrary sessions to analyze trends and variations. This helped me understand the distribution and quantity of distinct brain areas being measured.

Based on my analysis, I observed that a significant portion of trials had contrast levels of (0, 0), indicating the absence of stimuli. Interestingly, within this contrast level, there were more instances of success than failure, suggesting that the mice were able to distinguish when to withhold wheel movement when no stimuli were present.

I proceeded to use logistic regression as my predictive model in Section 4 due to its suitability for binary classification tasks. By training the model on the available data, I achieved an accuracy rate of 74% in predicting the feedback type for new trials. This indicates that the model is fairly effective in predicting the outcome based on the neural activity and contrast levels.

However, it is important to note that the generalization of the model's performance may be limited. Since the testing was conducted on the same session data as the training, it might not accurately reflect the model's performance on independent sessions. Additionally, further investigation and inclusion of additional information may be necessary to improve the model's generalization ability and capture more nuanced patterns in the data.

In summary, this project provides insights into the spike activity, contrast levels, and predictive modeling in the analyzed subset of data. The findings highlight the potential relationship between stimulus contrast and feedback type, and demonstrate the effectiveness of logistic regression in predicting trial outcomes. However, future work should focus on validating the model on independent sessions and exploring additional factors that may influence the predictions.

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

Code Used

```
knitr::opts_chunk$set(echo = TRUE)


library(tidyverse)
library(ggplot2)
library(dplyr)
```

```r
library(cluster)


library(kernlab)
library(caret)


#Allocating session
session=list()

#Reading the session files into a list of 18 elements
for(i in 1:18){
  session[[i]]=readRDS(paste('../data/mouse_data/session',i,'.rds',sep=''))


}

#Allocating mouse data frame
mouseData = data.frame()

#To iterate through the sessions
for(i in 1:18){

  #Temporary variable to store the allocated information for each session
  x = cbind(session[[i]]$contrast_left,session[[i]]$contrast_right,rep(i,length(session[[i]]$contrast_l

  #Binding the rows to the data frame
   mouseData = rbind(mouseData,x)

}



#Names of the data frame
colnames(mouseData) = c("contrast_left","contrast_right", "session","mouse","number_of_neurons","brain_a


##Checking the data frame
head(mouseData)

# To check the total number of trials
totalTrials = 0

for(i in 1:18){


  num = length(session[[i]]$feedback_type)
  totalTrials = totalTrials + num
}

#Confirming the dimensions (rows) are equivalent to the number of total trials

dim(mouseData)
```

```r
#Converting some data to factors for easier data analysis and manipulation
mouseData$session = as.factor(mouseData$session)
mouseData$mouse = as.factor(mouseData$mouse)
mouseData$feedback_type = as.factor(mouseData$feedback_type)
mouseData$brain_area = as.numeric(mouseData$brain_area)

head(mouseData)



mouseData %>% select(session,mouse,brain_area) %>% group_by(session,brain_area,mouse) %>% summarise("Nu

##Takes an input of a list (AKA session[[i]]), and extracts out the spike data  and sums across the row
manipulateData<-function(data,sessionNum){


  #Number of trials for each session
  trial_nums = NULL

  #Allocating variables
  brain_area<-data$brain_area
  spks<-cbind(brain_area,as.data.frame(sapply(data$spks,rowSums)))
  spks<-spks %>% group_by(brain_area) %>% summarise(across(everything(), sum))


  #Pivoting the data frame
  proper <- tidyr::pivot_longer(spks, cols = starts_with("V"), names_to = "Trial", values_to = "Spikes"



trial_numbers <- as.numeric(sub("V", "", grep("^V\\d+$", names(spks), value = TRUE)))

# Get the column names starting with "V" and extract the numeric part
trial_nums<-rep(trial_numbers,dim(proper %>% distinct(brain_area))[1])
proper$Trial<-c(trial_nums)




proper$session<-sessionNum

return(proper)

}


#Allocating the spike data frame
totalSpikeData<-NULL

for(i in 1:18){

  #Place holder for the spike data
```

```
  tempData <-manipulateData(session[[i]],i)


  #binding it to the data frame
  totalSpikeData<- rbind(totalSpikeData,tempData)
}


#Checking dimensions
dim(totalSpikeData)


#Confirming number of rows is correct for the newly created data frame
sum((mouseData %>% distinct(brain_area,number_of_trials)%>% pull(brain_area) %>% as.numeric())* (mouseDa


#Session numbers for a random sampling method (removing bias)
sessionNumbers<-1:18


##Selecting Random sessions to plot and see association between number of spikes and Trial number
for(i in sample(sessionNumbers,6,replace = F)){

print(ggplot(totalSpikeData %>% filter(session == i) %>% group_by(Trial) %>% summarise(AverageSpikes = m
}

for(i in sample(sessionNumbers,5,replace = F)){

  print(ggplot(totalSpikeData %>% filter(session == i),aes(x = Trial,y = Spikes,color = brain_area))+ ge


}

##Summarizing the spike data in order to merge it with the mouse data
summarisedSpikeData<- totalSpikeData %>%  group_by(session,Trial) %>% summarise(spikes = sum(Spikes))


perfectMouseData<-cbind(mouseData,summarisedSpikeData[-1])



##Creating the perfect mouse data.
head(perfectMouseData,20)




##Total counts data set
feedback_counts <- perfectMouseData %>%
  group_by(session, feedback_type) %>%
  summarise(count = n()) %>% group_by(session) %>% mutate(totals = sum(count))
```

```r
#Feedback proportions data set
feedback_proportions <- perfectMouseData %>%
  group_by(session, feedback_type) %>%
  summarise(count = n()) %>% mutate(percentage = count/sum(count))


##Total counts of feedback
total_counts = feedback_counts %>% group_by(session) %>% summarise(total = sum(count))

##Average feedback for success
average_rate <- feedback_proportions %>%filter(feedback_type ==1) %>% pull(percentage) %>% mean()



feedback_counts



ggplot(feedback_counts, aes(x = session, y = count, fill = feedback_type)) +
  geom_bar(stat = "identity") + geom_text(aes(label = totals),vjust = -.5,color = "black")+ scale_fill_m



ggplot(feedback_proportions, aes(x = session, y = percentage, fill = feedback_type)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("1" = "black", "-1" = "red")) + geom_hline(yintercept = average_rate, lin
  labs(x = "Session", y = "Count", fill = "Feedback Type") +
  ggtitle("Distribution of Feedbacks Across Sessions")



##Distribution of contrast varieties across all sessions
contrastLevels <-  perfectMouseData %>% group_by(contrast_left,contrast_right) %>% summarize(counts = n


##Distribution of contrast varieties across all sessions with feedback
contrastLevelsByFeedback = perfectMouseData %>% group_by(contrast_left,contrast_right,feedback_type) %>%







##Plots

##Counts of various Contrast Levels for trials
ggplot(contrastLevels, aes(x = contrast_left, y = contrast_right,size = counts/100,label = counts)) +
  geom_point()+geom_text(size = 3, vjust = -1.2)

## Counts of trials depending on Contrast Levels faceted by feedback type
ggplot(contrastLevelsByFeedback, aes(x = contrast_left, y = contrast_right,size = counts/100,label = cou
  geom_point()+geom_text(size = 3, vjust = -1.2)+facet_grid(~feedback_type)
```

```r
perfectMouseData %>% group_by(mouse) %>%summarise(DescisionMaking = sum(feedback_type == 1)/n())

test = perfectMouseData %>% group_by(mouse,session) %>%summarise(DescisionMakingEfficiency = sum(feedba

ggplot(test, aes(x = session, y = DescisionMakingEfficiency, fill = mouse)) +geom_bar(stat = "identity"



ggplot(test, aes(x = mouse, y = DescisionMakingEfficiency, fill = mouse)) +geom_bar(stat = "identity", p
```

```r
clusterData <-NULL
clusterDF<- perfectMouseData %>% select(contrast_left,contrast_right,spikes)

clusterData<- kmeans(clusterDF,2)

numberoftrials<-1:5081

clusterDF %>% mutate(cluster = clusterData$cluster) %>%
  ggplot(aes(y=spikes, x=numberoftrials, color = as.factor(cluster))) +
  geom_point()
```

```r
#Changing the variables to factors for the predictive model
perfectMouseData$contrast_left<-as.factor(as.character(perfectMouseData$contrast_left))
perfectMouseData$contrast_right<-as.factor(as.character(perfectMouseData$contrast_right))


##Filtering the data to train by
trainData<-perfectMouseData %>% filter(session %in% 2:17)
testData<-perfectMouseData %>% filter(session %in% c(1,18))
```

```r
print('Train Data:\n')
head(trainData)
print('Test Data: \n')
head(testData)
```

```r
summary(trainData)
```

```r
## The prediction model, using logistic regression
model = glm(feedback_type~contrast_left+contrast_right+spikes,family = 'binomial',trainData)
```

```r
summary(model)



predictions<-as.data.frame(predict(model, testData))
predictionsFact<- as.factor(ifelse(predictions>0.8,1,-1))







matrix<- confusionMatrix(testData$feedback,predictionsFact)


matrix$table


##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))

testData=list()

#Reading the session files into a list of 18 elements
for(i in 1:2){
  testData[[i]]=readRDS(paste('../data/mouse_data/test',i,'.rds',sep=''))


}

testMouseData = data.frame()

#To iterate through the sessions
for(i in 1:2){

  #Temporary variable to store the allocated information for each session
  x = cbind(testData[[i]]$contrast_left,testData[[i]]$contrast_right,rep(i,length(testData[[i]]$contras

  #Binding the rows to the data frame
   testMouseData = rbind(testMouseData,x)

}
#Names of the data frame
colnames(testMouseData) = c("contrast_left","contrast_right", "session","mouse","number_of_neurons","br


##Checking the data frame
head(testMouseData)

##Extracting the spike data

testSpikeData<-NULL
```

```
for(i in 1:2){

  #Place holder for the spike data
  tempData <-manipulateData(testData[[i]],i)


  #binding it to the data frame
  testSpikeData<- rbind(testSpikeData,tempData)
}

head(testSpikeData)
##Summarizing the spike data in order to merge it with the mouse data
summarisedSpikeData<- testSpikeData %>%  group_by(session,Trial) %>% summarise(spikes = sum(Spikes))
finalTestMouseData<-cbind(testMouseData,summarisedSpikeData[-1])

##Creating the perfect mouse data.
head(finalTestMouseData,20)




## This experiment is for test data 1
## The prediction model, using logistic regression
##From the trained data
summary(model)



testDataPredictions<-as.data.frame(predict(model, (finalTestMouseData %>% filter(session == 1))))
predictionsFact<- as.factor(ifelse(testDataPredictions>0.5,1,-1))


predictionsFact

testData1 <-finalTestMouseData %>%  filter(session == 1)

matrix<- confusionMatrix(as.factor(testData1$feedback),predictionsFact)


matrix$table


##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))
## This experiment is for test data 2

testDataPredictions<-as.data.frame(predict(model, (finalTestMouseData %>% filter(session == 2))))
predictionsFact<- as.factor(ifelse(testDataPredictions>0.5,1,-1))


predictionsFact

testData2 <-finalTestMouseData %>%  filter(session == 2)
```

```r
matrix<- confusionMatrix(as.factor(testData2$feedback_type),predictionsFact)


matrix$table


##Misclassification Rate for the predictive Model
1-sum(diag(matrix$table)/sum(matrix$table))



sessionInfo()
```

Session information

```r
sessionInfo()
```

```
## R version 4.3.0 (2023-04-21 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 x64 (build 22621)
##
## Matrix products: default
##
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## time zone: America/Los_Angeles
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] caret_6.0-94    lattice_0.21-8  kernlab_0.9-32  cluster_2.1.4
##  [5] lubridate_1.9.2 forcats_1.0.0   stringr_1.5.0   dplyr_1.1.2
##  [9] purrr_1.0.1     readr_2.1.4     tidyr_1.3.0     tibble_3.2.1
## [13] ggplot2_3.4.2   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.3         xfun_0.39            recipes_1.0.6
##  [4] tzdb_0.4.0           vctrs_0.6.2          tools_4.3.0
##  [7] generics_0.1.3       stats4_4.3.0         parallel_4.3.0
## [10] proxy_0.4-27         fansi_1.0.4          highr_0.10
## [13] ModelMetrics_1.2.2.2 pkgconfig_2.0.3      Matrix_1.5-4
## [16] data.table_1.14.8    lifecycle_1.0.3      farver_2.1.1
```

```
## [19] compiler_4.3.0       munsell_0.5.0        codetools_0.2-19
## [22] htmltools_0.5.5      class_7.3-21         yaml_2.3.7
## [25] prodlim_2023.03.31   crayon_1.5.2         pillar_1.9.0
## [28] MASS_7.3-58.4        gower_1.0.1          iterators_1.0.14
## [31] rpart_4.1.19         foreach_1.5.2        parallelly_1.36.0
## [34] nlme_3.1-162         lava_1.7.2.1         tidyselect_1.2.0
## [37] digest_0.6.31        stringi_1.7.12       future_1.32.0
## [40] reshape2_1.4.4       listenv_0.9.0        labeling_0.4.2
## [43] splines_4.3.0        fastmap_1.1.1        grid_4.3.0
## [46] colorspace_2.1-0     cli_3.6.1            magrittr_2.0.3
## [49] survival_3.5-5       utf8_1.2.3           e1071_1.7-13
## [52] future.apply_1.11.0  withr_2.5.0          scales_1.2.1
## [55] timechange_0.2.0     rmarkdown_2.22       globals_0.16.2
## [58] nnet_7.3-18          timeDate_4022.108    hms_1.1.3
## [61] evaluate_0.21        knitr_1.43           hardhat_1.3.0
## [64] rlang_1.1.1          Rcpp_1.0.10          glue_1.6.2
## [67] pROC_1.18.2          ipred_0.9-14         rstudioapi_0.14
## [70] R6_2.5.1             plyr_1.8.8
```