Hockey Analytics Final Project Report

Jeff Jung

Fall 2023 Nov 26th

Contents

DAR Project and Group Members **1**

Project **1**

Abstract **1**

Introduction and Background **2**

Problems Tackled **4**

Data Description **4**

Data Analytics Methods **15**

Experimental Results **18**

Discussion of Results and Key Findings **22**

Conclusions **22**

Directions for Future Investigation **23**

Bibliography **23**

Files and Github Commits **23**

Contribution **23**

Appendix **23**

DAR Project and Group Members

Jeff Jung Caleb Smith Liebin Zhang Ashley Woodson Amy Enyenihi

Project

Hockey Analytics

Abstract

- Data cleaning by removing the outliers

- Categorization by analyzing density plots
- Making cluster models using k means
- Cluster analysis

Introduction and Background

Since we did not have much data for goals, my research was focused on examining dataset only with the saves. My initial approach was finding the median and standard deviation of saves, so that I can find the modes of failure. However, median and standard deviation cannot explain the details of data. So I used density plot for visualization and saw some interesting observations. Data for each variable can be well categorized depending on the shape of a density plot. We can clearly see that there are certain ranges where data are concentrated. Therefore, I decided to categorize continuous variables and examine those variables.

```
# Code

# Load required library
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.1.1 --

## v broom        1.0.5     v rsample      1.2.0
## v dials        1.2.0     v tibble       3.2.1
## v ggplot2      3.4.3     v tidyr        1.3.0
## v infer        1.0.5     v tune         1.1.2
## v modeldata    1.2.0     v workflows    1.1.3
## v parsnip      1.1.1     v workflowsets 1.0.1
## v purrr        1.0.2     v yardstick    1.2.0
## v recipes      1.0.8

## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
## * Search for functions across packages at https://www.tidymodels.org/find/
```

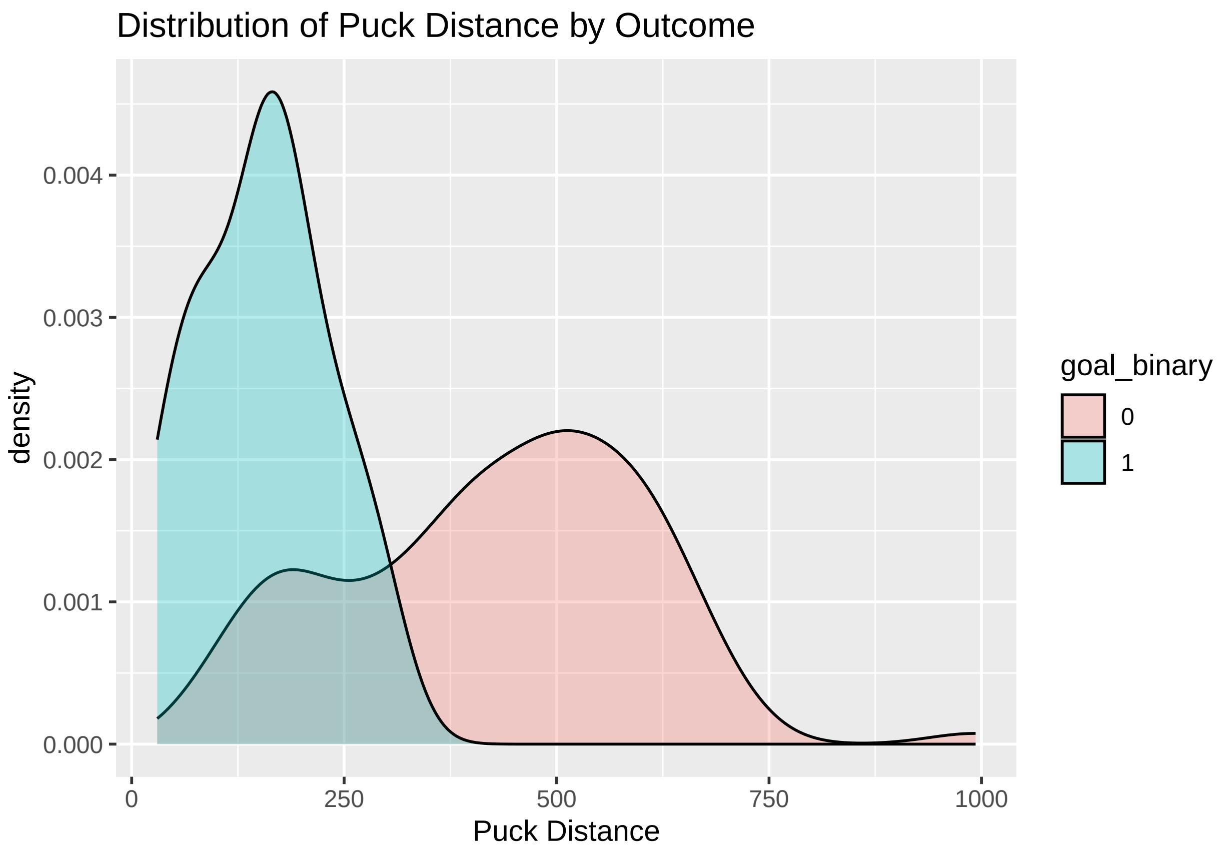
```
library(knitr)
library(pheatmap)

# load datasets
shots <- readRDS('shots_stats_goal.df.Rds')

# Creates a variable goal_binary that contains binary values 0 for save 1 for goal
shots <- shots %>%
  mutate(goal_binary = as.integer(outcomes.goal == 2))

# convert goal_binary into a factor
shots$goal_binary <- as.factor(shots$goal_binary)

# Subset the data for not goals (e.g., "Save")
saves <- shots[shots$goal_binary != "1", ]

# Mean and standard deviation of puckDist from saved shots data
median_puckDist_saves <- median(saves$puckDist)
sd_puckDist_saves <- sd(saves$puckDist)
cat("Median of Puck Distance for Saves:", median_puckDist_saves, "\n")
```

```
## Median of Puck Distance for Saves: 454.3136
```

```
cat("Standard Deviation of Puck Distance for Saves:", sd_puckDist_saves, "\n")
```

```
## Standard Deviation of Puck Distance for Saves: 174.835
```

```
ggplot(shots, aes(x = puckDist, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Distance by Outcome",
       x = "Puck Distance",)
```



Problems Tackled

The goal of my work is discretization. To do this, we first need to clean the data so that it does not produce potentail skewness in the data. Then, with the clean data we look at the density plots to determine the range used for discretization. Using the k-means method, we can make two groups of clusters - one for the continuous variables and the other for the categorical variables. Then we can put them into a matrix and produce a heatmap to show how two clusters are correlated. Also, we can analyze how each cluster is correlated with different outcomes: save, goal, defender block, or miss.

Data Description

The dataset that is primarily used ("shots") is generated from the games played by RPI women's hockey team. There are 210 observations or shots recorded with 9 continous and 5 categorical variables. Continuous variables include puckDist, puckAngle, puckSpeed, shooterSpeed, goalieDist, goalieAngle, posTime, defDist, and defAngle. And categorical variables include NumOffense, NumDefense, rightHanded, closestDef, and shotOutcome.

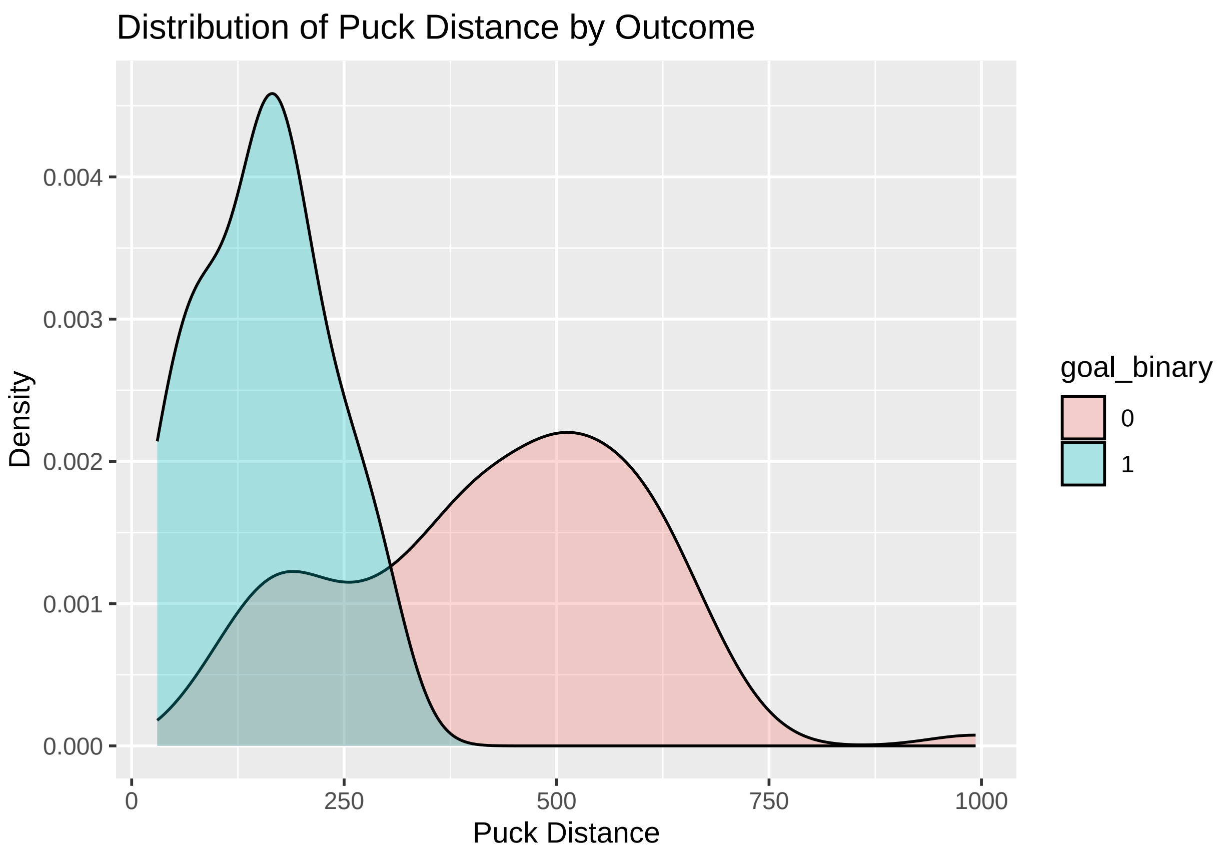
The first step is data cleaning. For data cleaning, I made two functions: detect_outlier and remove_outlier. The first function, detect_outlier find all the outliers and remove_outlier removes all data that were classified as outliers in detect_outlier function.

Then, I examined density plots for each continuous variable and found range(s) to categorize continuous variables. The categorization was done by finding the number of concentrated regions in each density plot and the corresponding x range for each variable.

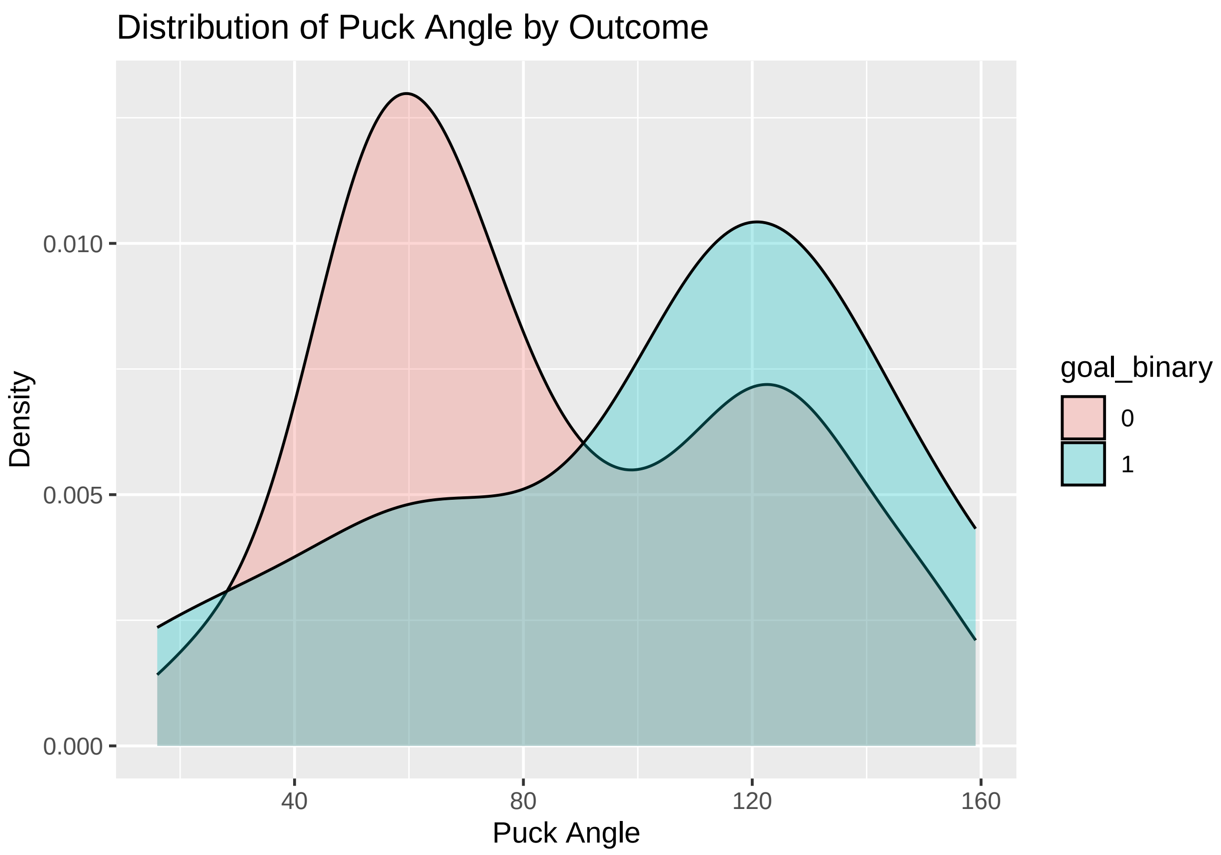
```r
# Code

# load datasets
shots <- readRDS('shots_stats_goal.df.Rds')

# create detect outlier function
detect_outlier <- function(x) {

    # calculate first quantile
    Quantile1 <- quantile(x, probs=.25)

    # calculate third quantile
    Quantile3 <- quantile(x, probs=.75)

    # calculate inter quartile range
    IQR = Quantile3-Quantile1

    # return true or false
    x > Quantile3 + (IQR*1.5) | x < Quantile1 - (IQR*1.5)
}

# create remove outlier function
remove_outlier <- function(dataframe,
                            columns=names(dataframe)) {

    # for loop to traverse in columns vector
    for (col in columns) {

        # remove observation if it satisfies outlier function
        dataframe <- dataframe[!detect_outlier(dataframe[[col]]), ]
    }
}

# Select the columns where you want to remove outliers
remove_outlier(shots, c('puckDist', 'puckAngle', 'shooterSpeed','goalieDist','goalieAngle', 'posTime',

# Manual selection for categories based on the shape of density plots above
puckDist_q <- c(310)
puckAngle_q <- c(90)
puckSpeed_q <- c(15, 50)
shooterSpeed_q <- c(17, 25)
goalieDist_q <- c(58, 80)
goalieAngle_q <- c(80)
posTime_q <- c(45)
defDist_q <- c(250)
defAngle_q <- c(63, 112)

# copy of shots for categorical a dataset
catshots <- shots

# Create categorical variables
catshots <- catshots %>%
  mutate(puckSpeedCategory = case_when(
    puckSpeed <= puckSpeed_q[1] ~ 0,
    puckSpeed <= puckSpeed_q[2] ~ 1,
    TRUE ~ 2
  ))
catshots <- catshots %>%
  mutate(puckAngleCategory = case_when(
    puckAngle <= puckAngle_q[1] ~ 0,
    TRUE ~ 1
  ))
catshots <- catshots %>%
  mutate(puckDistCategory = case_when(
    puckDist <= puckDist_q[1] ~ 0,
    TRUE ~ 1
  ))
catshots <- catshots %>%
  mutate(posTimeCategory = case_when(
    posTime <= posTime_q[1] ~ 0,
    TRUE ~ 1
  ))
catshots <- catshots %>%
  mutate(goalieDistCategory = case_when(
    goalieDist <= goalieDist_q[1] ~ 0,
    goalieDist <= goalieDist_q[2] ~ 1,
    TRUE ~ 2
  ))
catshots <- catshots %>%
  mutate(shooterSpeedCategory = case_when(
    shooterSpeed <= shooterSpeed_q[1] ~ 0,
    shooterSpeed <= shooterSpeed_q[2] ~ 1,
    TRUE ~ 2
```

```r
  ))
catshots <- catshots %>%
  mutate(goalieAngleCategory = case_when(
    goalieAngle <= goalieAngle_q[1] ~ 0,
    TRUE ~ 1
  ))
catshots <- catshots %>%
  mutate(defDistCategory = case_when(
    defDist <= defDist_q[1] ~ 0,
    TRUE ~ 1
  ))
catshots <- catshots %>%
  mutate(defAngleCategory = case_when(
    defAngle <= defAngle_q[1] ~ 0,
    defAngle <= defAngle_q[2] ~ 1,
    TRUE ~ 2
))

# Drop continous variables for categorical dataset
catshots <- catshots %>%
    select(- puckDist, - puckAngle, - puckSpeed, - shooterSpeed, - goalieDist, - goalieAngle, - posTime

# Convert categorized variable as factors
catshots[sapply(catshots, is.numeric)] <- lapply(catshots[sapply(catshots, is.numeric)], as.factor)
```

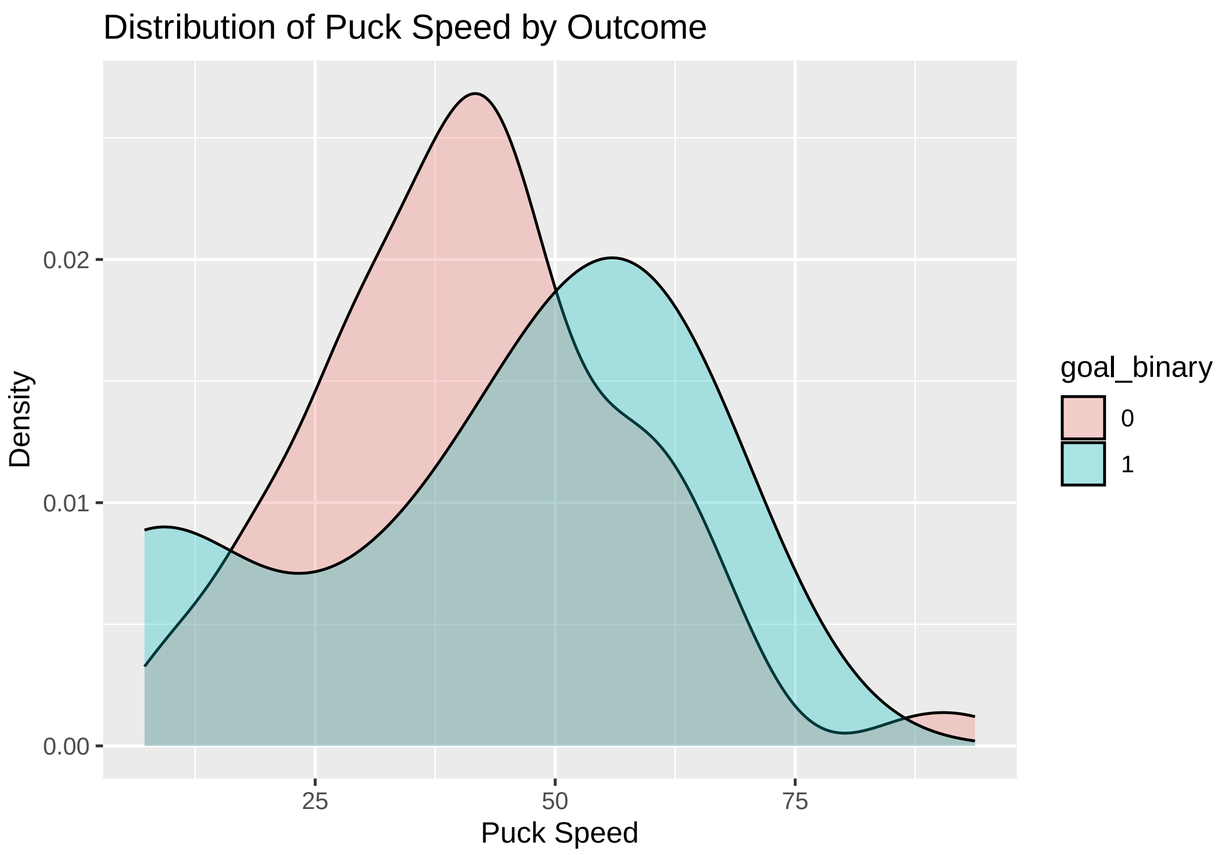
```r
# Code

# Creates a variable goal_binary that contains binary values 0 for save 1 for goal
shots <- shots %>%
  mutate(goal_binary = as.integer(outcomes.goal == 2))

# Convert goal_binary into a factor
shots$goal_binary <- as.factor(shots$goal_binary)

# Density plot of each variable
ggplot(shots, aes(x = puckDist, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Distance by Outcome",
       x = "Puck Distance",
       y = "Density")
```



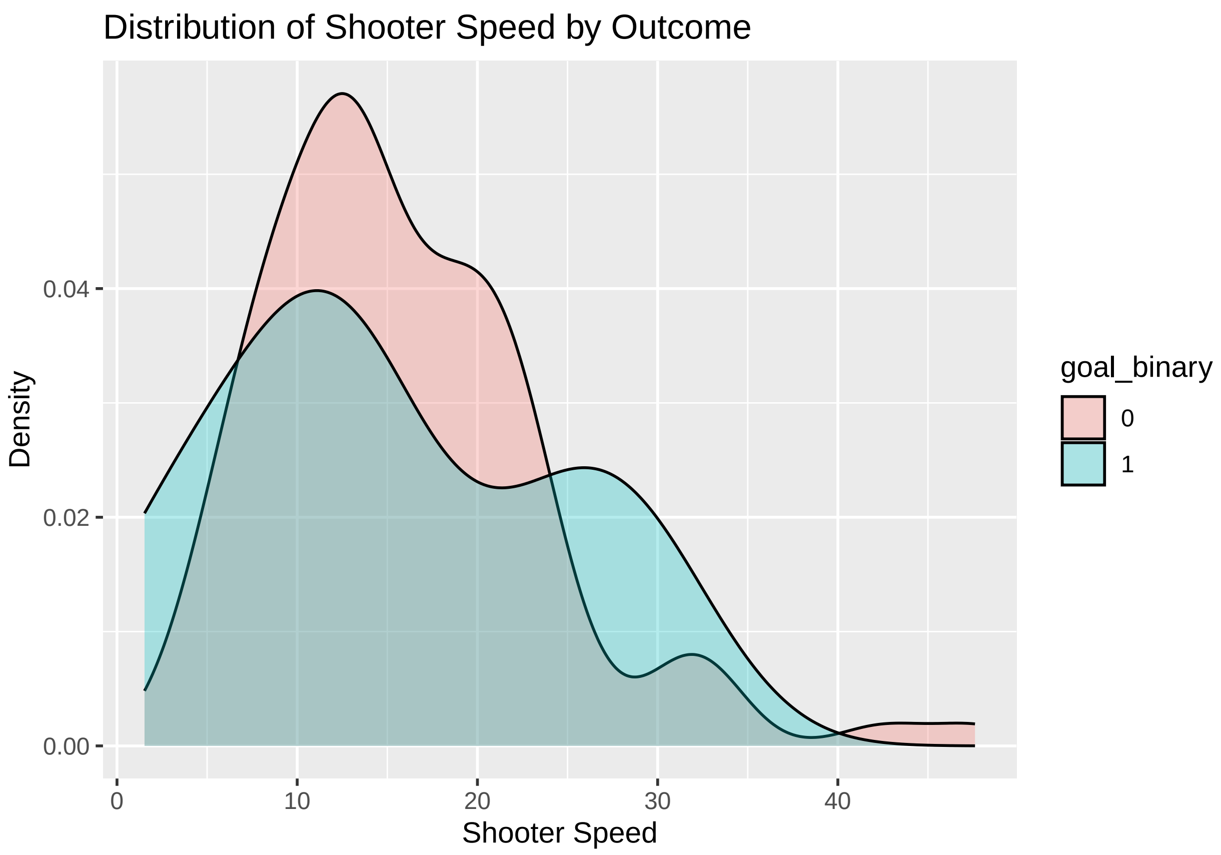
```
ggplot(shots, aes(x = puckAngle, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Angle by Outcome",
       x = "Puck Angle",
       y = "Density")
```



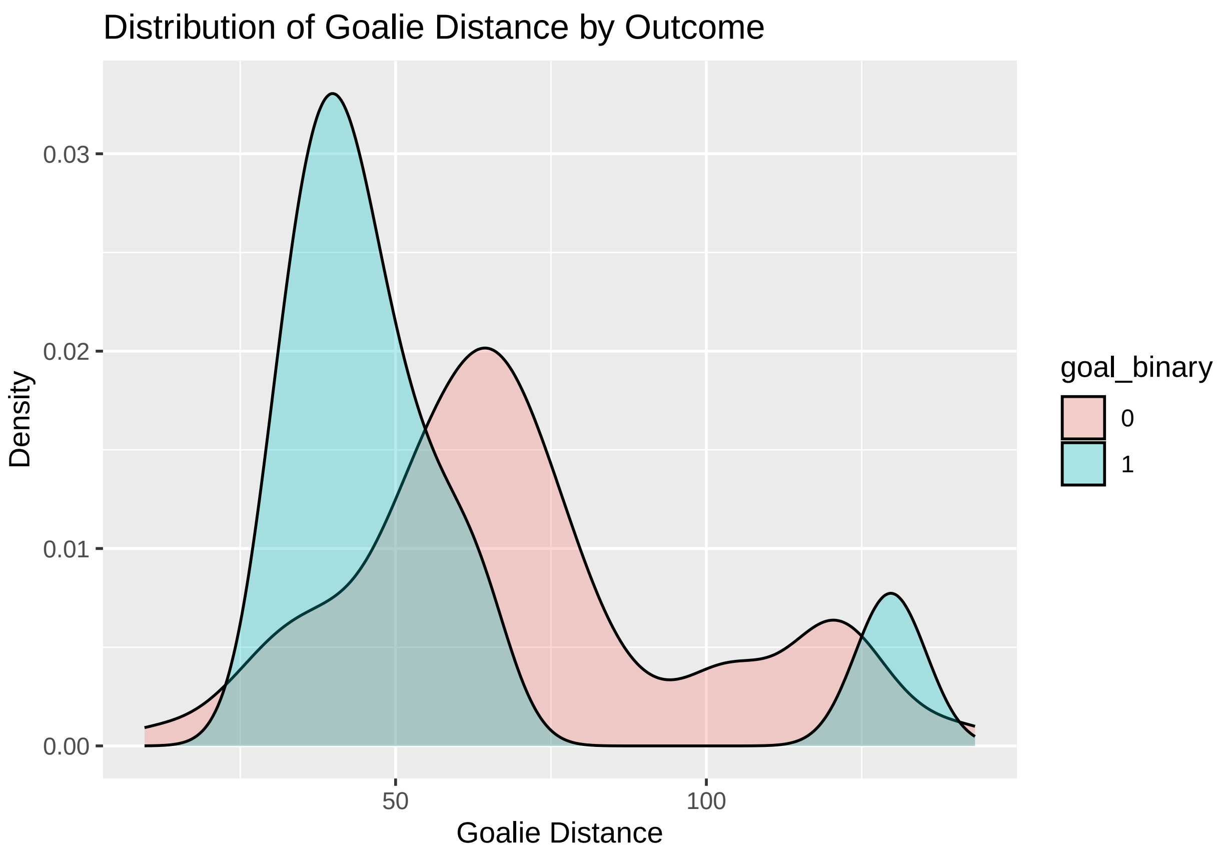
```
ggplot(shots, aes(x = puckSpeed, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Speed by Outcome",
       x = "Puck Speed",
       y = "Density")
```



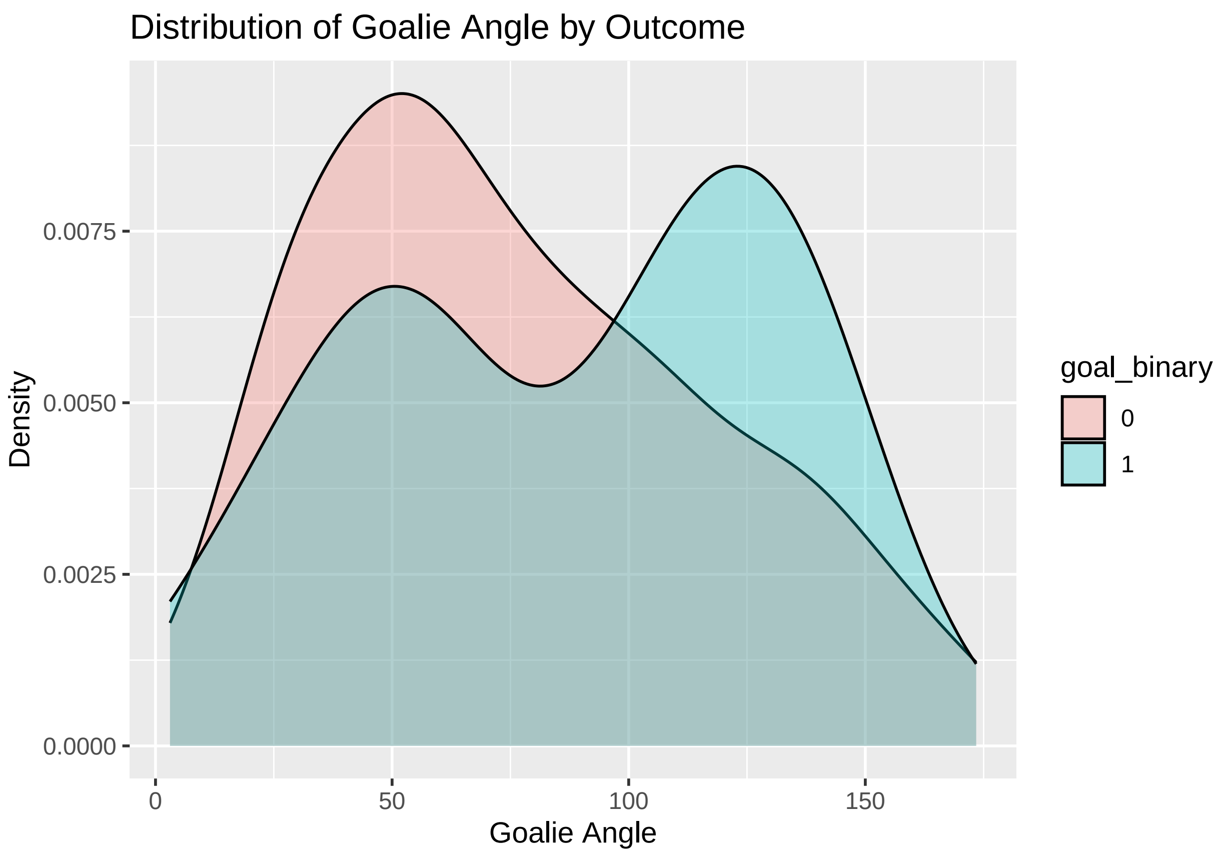
```r
ggplot(shots, aes(x = shooterSpeed, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Shooter Speed by Outcome",
       x = "Shooter Speed",
       y = "Density")
```



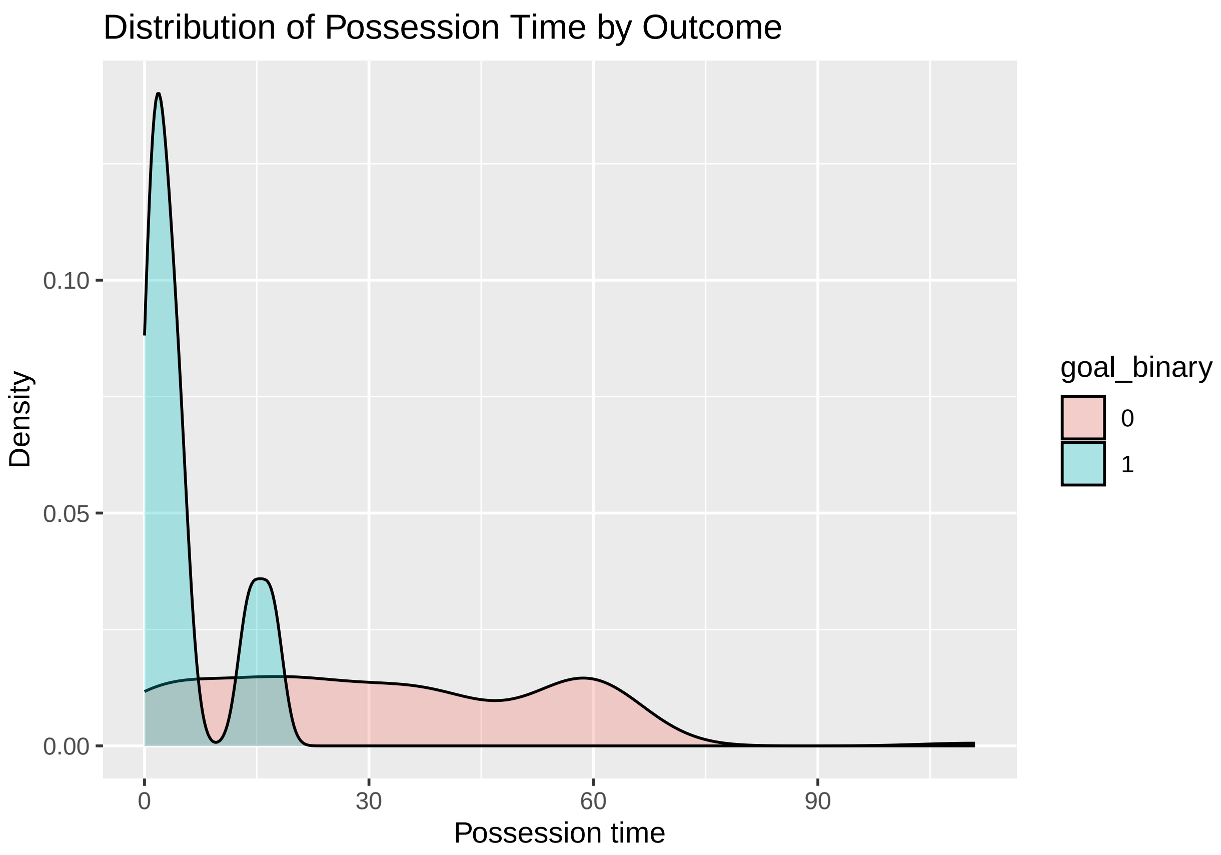
```r
ggplot(shots, aes(x = goalieDist, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Goalie Distance by Outcome",
       x = "Goalie Distance",
       y = "Density")
```



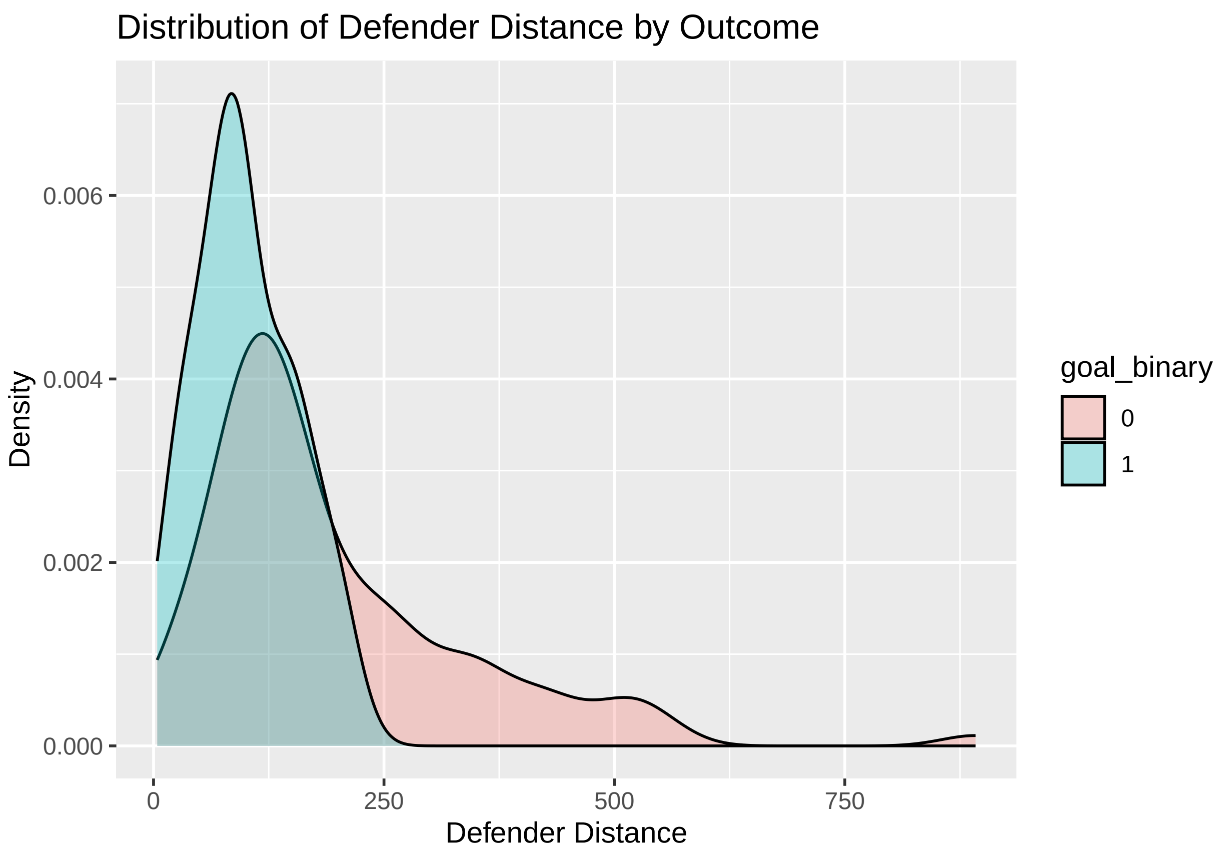
```
ggplot(shots, aes(x = goalieAngle, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Goalie Angle by Outcome",
       x = "Goalie Angle",
       y = "Density")
```



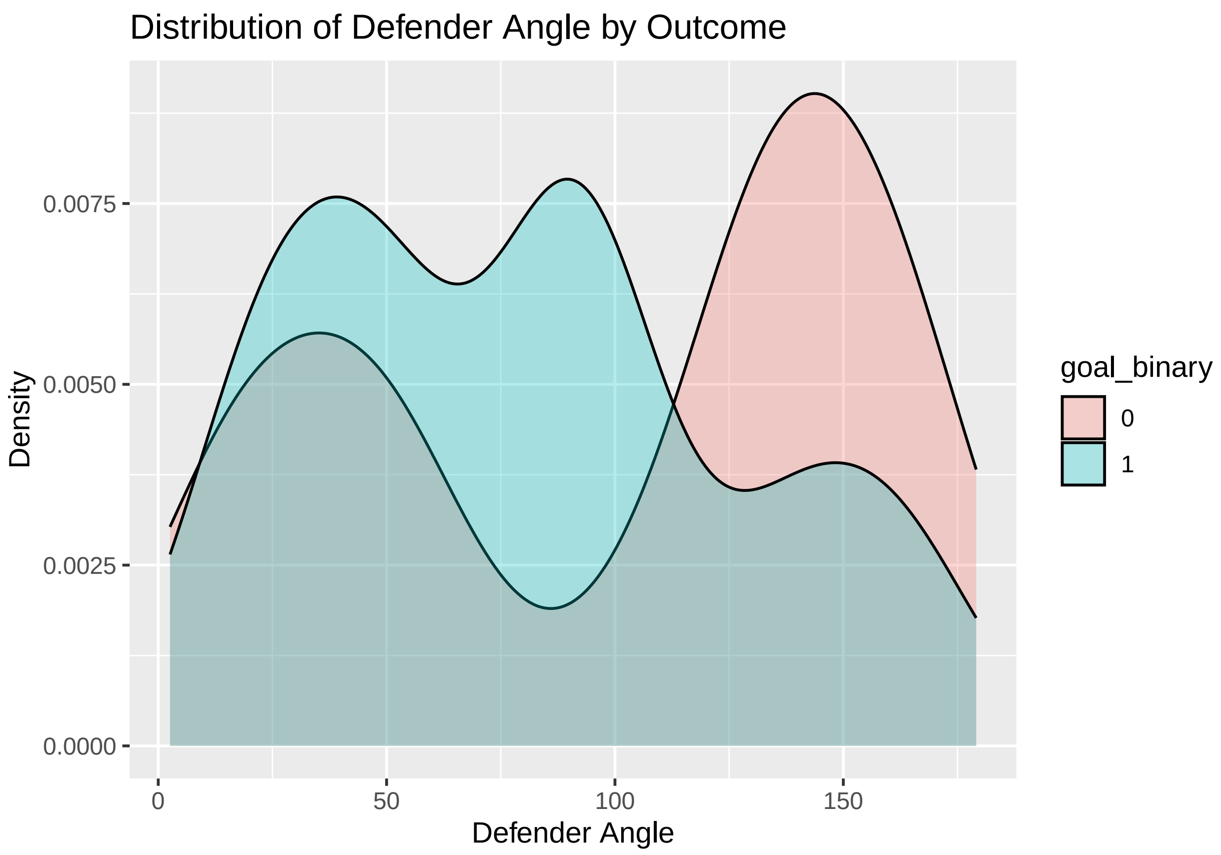
```
ggplot(shots, aes(x = posTime, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Possession Time by Outcome",
       x = "Possession time",
       y = "Density")
```



```
ggplot(shots, aes(x = defDist, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Defender Distance by Outcome",
       x = "Defender Distance",
       y = "Density")
```



```
ggplot(shots, aes(x = defAngle, fill = goal_binary)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Defender Angle by Outcome",
       x = "Defender Angle",
       y = "Density")
```

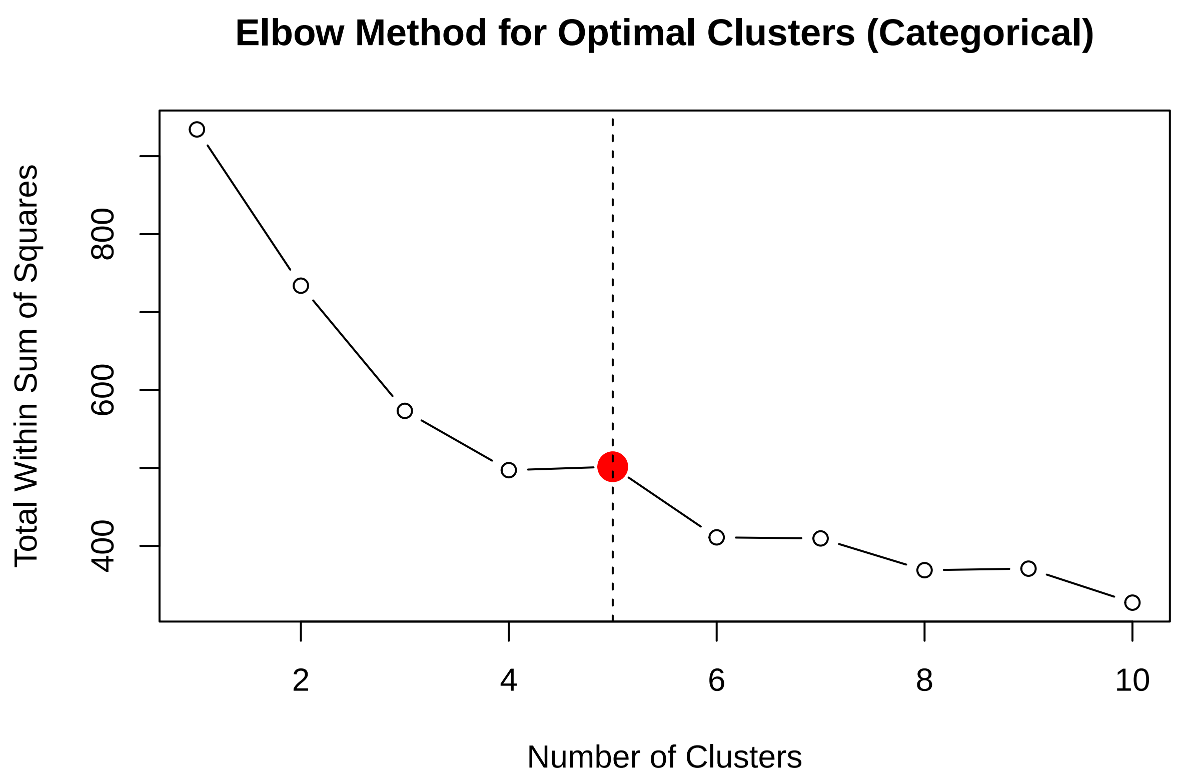


Data Analytics Methods

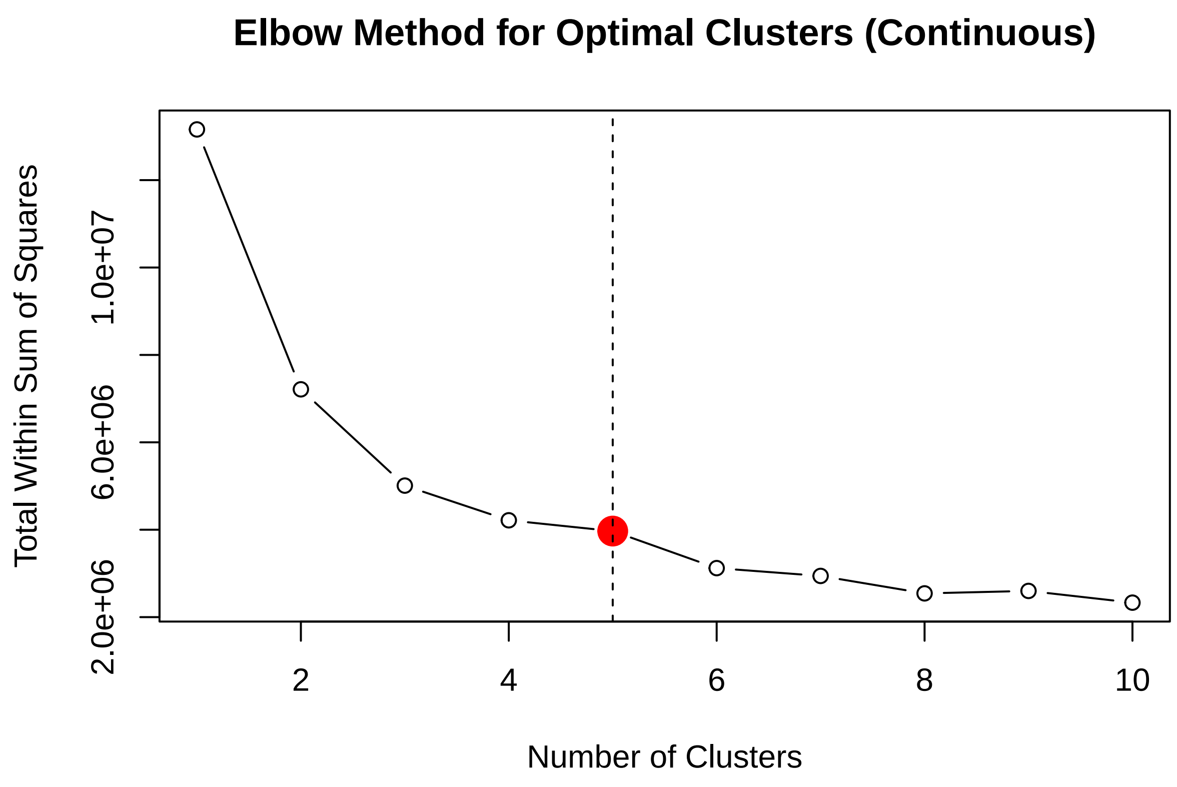
With the cleaned data (no outliers) that I obtained from above, I dropped categorical variables that originally existed in the shot__stats__goal data so that I can only compare the categorical variables that I made to be compared with corresponding continuous variables. Using the k-means machine learning algorithm, I made two models, one for each original and categorical data, and saved them as a file because k-means algorithm assigns different cluster values each time. To make sure that we have an optimal number of clusters, I performed an elbow test. However, I still used 5 clusters to align with Caleb's cluster data.

```
# Code

# Drop variables that were categorical at the first place
catshots <- catshots %>%
    select(- NumOffense, - NumDefense, - rightHanded, -closestDef, - shotOutcome)
shots <- shots %>%
  select(- NumOffense, - NumDefense, - rightHanded, -closestDef, - shotOutcome, - goal_binary)

# Making models using k means
# org_model <- kmeans(shots, centers = 5)
# cat_model <- kmeans(catshots, centers = 5)

# Save models derived from using k means
# saveRDS(org_model, file = "org_model.rds")
# saveRDS(cat_model, file = "cat_model.rds")

# Read the models
```

```r
cat_model <- readRDS('cat_model.rds')
org_model <- readRDS('org_model.rds')

# # Get cluster assignments
org_cluster <- org_model$cluster
cat_cluster <- cat_model$cluster

# Compute the total within-cluster sum of squares for different numbers of clusters
cat_wss_values <- c()
for (i in 1:10) {
  cat_elbow <- kmeans(catshots, centers = i)
  cat_wss_values <- c(cat_wss_values, cat_elbow$tot.withinss)
}
org_wss_values <- c()
for (i in 1:10) {
  org_elbow <- kmeans(shots, centers = i)
  org_wss_values <- c(org_wss_values, org_elbow$tot.withinss)
}

# This is for categorical model
# Plot the Elbow plot to visualize the within-cluster sum of squares by number of clusters
plot(1:10, cat_wss_values, type = "b", xlab = "Number of Clusters", ylab = "Total Within Sum of Squares
     main = "Elbow Method for Optimal Clusters (Categorical)")

# Identify the elbow point
cat_elbow_k <- 1
for (i in 2:(length(cat_wss_values) - 1)) {
  if ((cat_wss_values[i] - cat_wss_values[i - 1]) > (cat_wss_values[i + 1] - cat_wss_values[i])) {
    cat_elbow_k <- i
    break
  }
}

# Highlight the elbow point in the plot
points(cat_elbow_k, cat_wss_values[cat_elbow_k], col = "red", cex = 2, pch = 19)
abline(v = cat_elbow_k, lty = 2)
```



```r
# This is for the continous model
# Plot the Elbow plot to visualize the within-cluster sum of squares by number of clusters
plot(1:10, org_wss_values, type = "b", xlab = "Number of Clusters", ylab = "Total Within Sum of Squares
     main = "Elbow Method for Optimal Clusters (Continuous)")

# Identify the elbow point
org_elbow_k <- 1
for (i in 2:(length(org_wss_values) - 1)) {
  if ((org_wss_values[i] - org_wss_values[i - 1]) > (org_wss_values[i + 1] - org_wss_values[i])) {
    org_elbow_k <- i
    break
  }
}

# Highlight the elbow point in the plot
points(org_elbow_k, org_wss_values[org_elbow_k], col = "red", cex = 2, pch = 19)
abline(v = org_elbow_k, lty = 2)
```



Experimental Results

Using the original and categorical clusters I found above, I created a heatmap using correlation matrix between the two groups of clsuters. They show how two variables are correlated based on the number and color within each cell. The bigger the number is the higher the correlation is between and the smaller the number is the lower the correlation is. Also, I created the heatmap using correlation matrix between each outcome and two groups of clusters. These heatmaps will show how each cluster is likely to have a certain outcome. I found the mean value for each variable in shots and mode value for each variable in catshots, so that I can classify what kind of shots are in that cluster.

Looking at the correlation matrix between the original and categorical clusters, they have pretty good correlations because ideally, we want one original cluster to be highly correlated with exactly one categorical cluster. Original 1 is highly correlated with categorical 1, and all others have slightly negative correlation. This means that the shots in these two clusters are very similar. Original 2 is highly correlated with categorical 2 and 4 and somewhat correlated with 3. Original 3 is highly correlated with categorized 5. Original 4 is not well correlated with clusters from categorical data. Original 5 is highly correlated with categorical 3. Overall, except for original 2 and 4, other original clusters show one to one correlation.

(Naming each cluster using means and modes will be added later)

```
# Code

# Calculate means for each cluster for continuous variables
means_org <- aggregate(shots, by = list(org_cluster), FUN = mean)
```

```
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA

## Warning in mean.default(X[[i]], ...): argument is not numeric or logical:
## returning NA
```

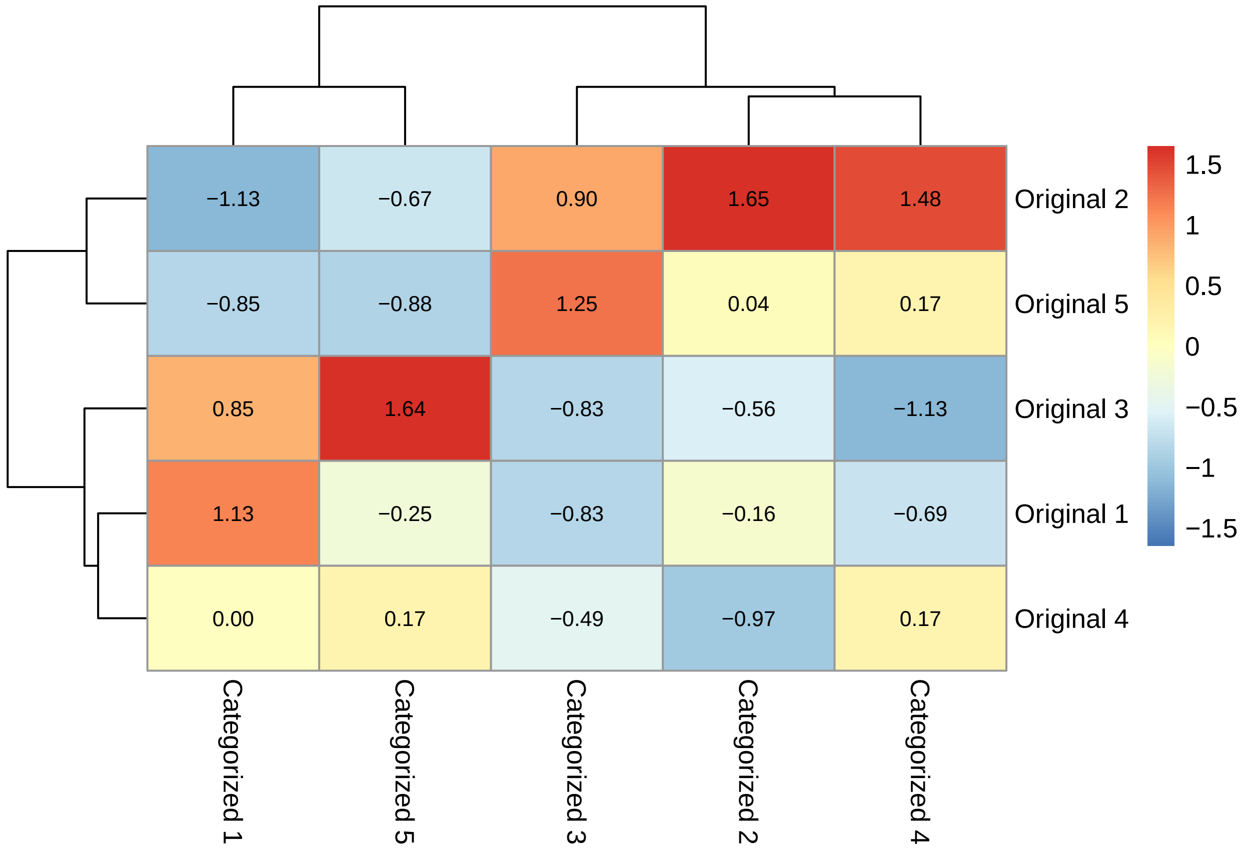
```
print(means_org)
```

```
##   Group.1 puckDist puckAngle puckSpeed shooterSpeed goalieDist goalieAngle
## 1       1 429.0734  85.08309  43.24438     18.63413   74.60591    78.96551
## 2       2 577.3543  82.26851  38.55734     12.54376   83.53557    75.38863
## 3       3 147.5481  77.51730  38.48058     17.18447   45.33600    63.08071
## 4       4 290.6015  98.85316  42.68814     18.39713   65.12851    96.68207
## 5       5 549.3437  90.63352  47.81298     14.04467   75.97900    80.39762
##    posTime   defDist  defAngle outcomes.goal
## 1 42.78947 120.64397 116.34889            NA
## 2 30.60714 161.56247  94.38426            NA
## 3 20.66667  94.58971 102.01265            NA
## 4 27.18750 159.86220  87.33023            NA
## 5 25.44444 446.61064  92.61817            NA
```

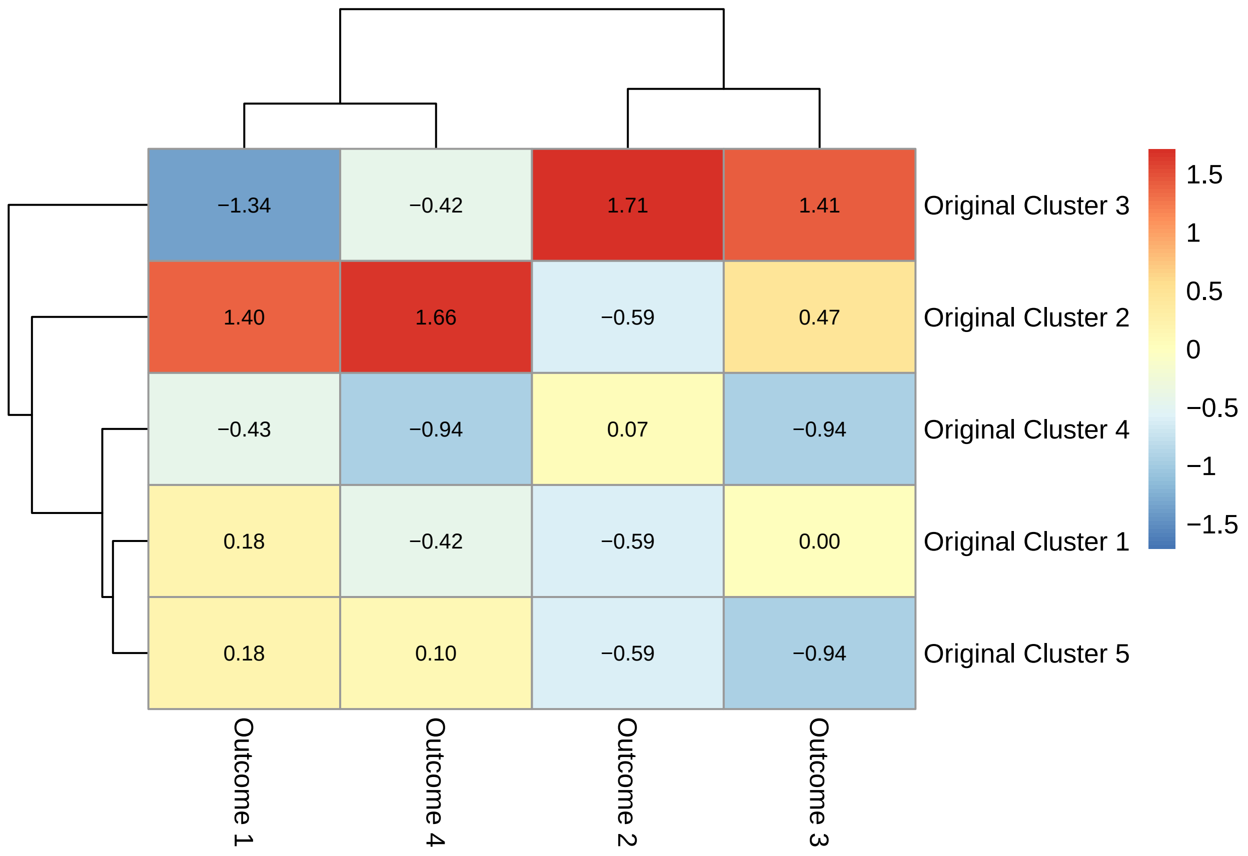
```
# function for finding the mode
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

# calculate modes for each cluster for categorical variables
modes_by_cluster <- aggregate(. ~ cat_cluster, data = catshots, FUN = Mode)

# Print the result
print(modes_by_cluster)
```

```
##   cat_cluster outcomes.goal puckSpeedCategory puckAngleCategory
## 1           1             3                 3                 1
## 2           2             3                 2                 1
## 3           3             3                 2                 2
## 4           4             1                 2                 2
## 5           5             3                 2                 2
##   puckDistCategory posTimeCategory goalieDistCategory shooterSpeedCategory
## 1                2               2                  2                    2
## 2                2               1                  2                    1
## 3                2               1                  3                    1
## 4                2               1                  2                    1
## 5                1               1                  1                    1
##   goalieAngleCategory defDistCategory defAngleCategory
## 1                   1               1                3
## 2                   1               1                3
## 3                   2               2                1
## 4                   2               1                1
## 5                   2               1                1
```

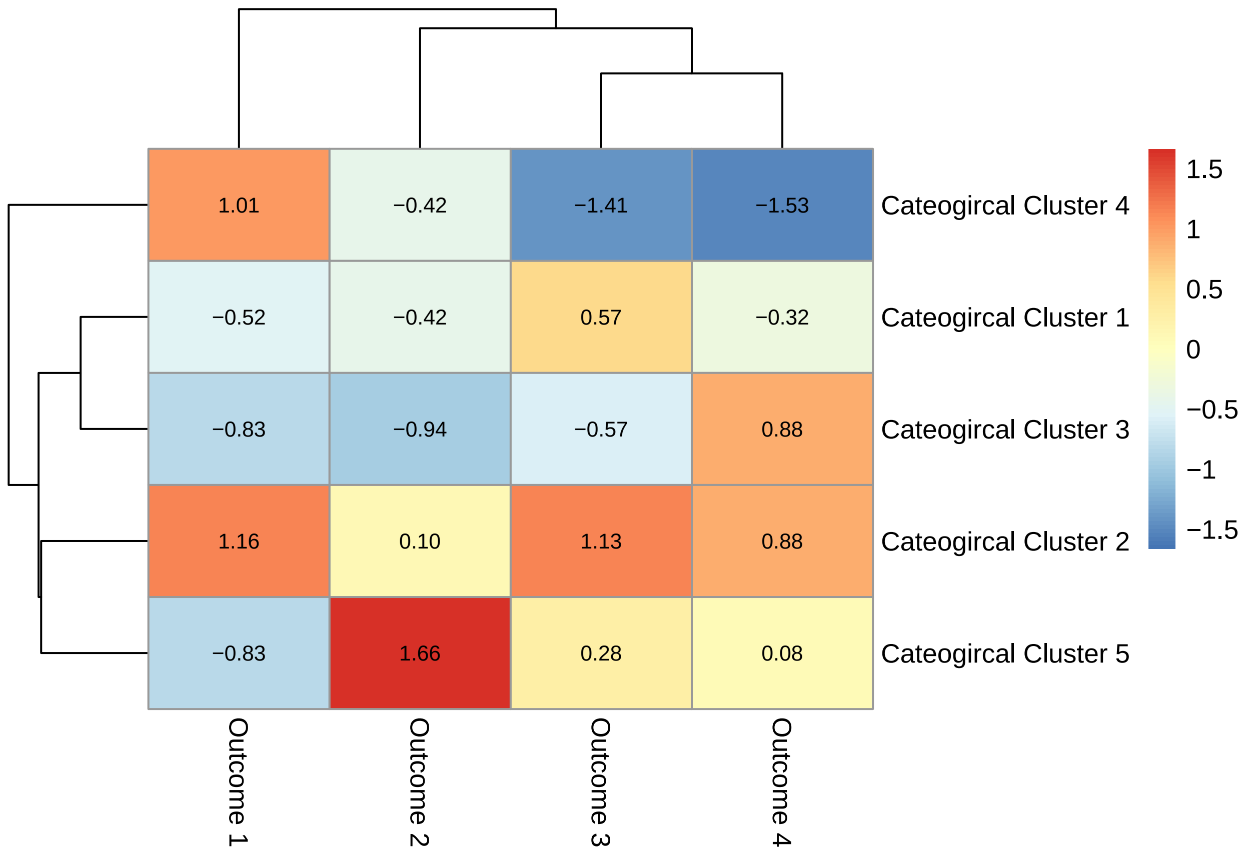
```r
# Create a matrix
conf_mat <- table(org_cluster, cat_cluster)

# Modify row and column names
rownames(conf_mat) <- paste("Original", 1:5)
colnames(conf_mat) <- paste("Categorized", 1:5)
# Plot the confusion matrix as a heatmap
pheatmap(conf_mat,
         scale = "column",
         display_numbers = TRUE,
         number_color = "black",
         fontsize_number = 8)
```



```r
# Make a matrix with the clusters from the original data and 4 outcomes of shots
org_conf_mat <- table(org_cluster, shots$outcomes.goal)

# Plot the confusion matrix as a heatmap
rownames(org_conf_mat) <- paste("Original Cluster", 1:5)
colnames(org_conf_mat) <- paste("Outcome", 1:4)

pheatmap(org_conf_mat,
         scale = "column",
         display_numbers = TRUE,
         number_color = "black",
         fontsize_number = 8)
```



```
# Make a matrix with the clusters from the original data and 4 outcomes of shots
cat_conf_mat <- table(cat_cluster, catshots$outcomes.goal)

# Plot the confusion matrix as a heatmap
rownames(cat_conf_mat) <- paste("Cateogircal Cluster", 1:5)
colnames(cat_conf_mat) <- paste("Outcome", 1:4)

pheatmap(cat_conf_mat,
         scale = "column",
         display_numbers = TRUE,
         number_color = "black",
         fontsize_number = 8)
```



Discussion of Results and Key Findings

The continuous variables are only able to show a single value for the best shots, but making the continuous variables categorical is capable of showing even more valuable findings because it is easier to show what different combinations of each feature will produce as an outcome. Using the k-means method to clustering the categorical and continuous variables show that even with different measures of clustering, there are some patterns in shots. And we saw this by looking at the correlation matrix, which showed mostly showed one to one correlation between two groups of clusters. Therefore, we know that these shots did not happen by chance, but these shots are commonly happening in the games of hockey.

These findings are very imporatnt because further analyzing on these clusters of shots using categorical variables can suggest to players what the best combinations of a shot is under different situations. This approach to hockey can answer many questions that is not possible to see without data analysis. For example, when you have to make a shot when the puckDist is large, then what is the ideal goalie angle in terms of goalie angle category?

(Further analysis on clusters after naming clusters will be added later)

Conclusions

We can clearly see that the clusters have significant meanings, and that they are not clustered in that way by a chance. They are clustered in a way that shots are often taken. I believe hockey data analytics will open a new world of understanding hockey because there are findings that can only be done through data analytics. Discretization of data will help doing so because continuous variables only tell you a single number, but a categorical variable can tell you what type of shots are likely to be a goal. With hockey data analytics, we can take more scientific approach to making better shots using insights hidden behind numbers.

Directions for Future Investigation

Using larger data, we can further analyze using categorical variables. With more data, we can come up with better categorization, and we might be able to have more clusters since we might find different kinds of shots. The purpose of doing so is, as I mentioned earlier, to find out what the best shot is under some situations where you are forced to make a shot.

Bibliography

http://www.sthda.com/english/wiki/saving-data-into-r-data-format-rds-and-rdata#google_vignette
https://stackoverflow.com/questions/32684931/how-to-aggregate-data-in-r-with-mode-most-common-value-for-each-row

Files and Github Commits

Commit your final files and any supporting files to github. List your files and their locations here and explain purpose of each one. Make sure to include all files necessary to reproduce your results.

Uploaded the final draft to Assignment07 folder

Contribution

N/A

Appendix

(Will be added to the final notebook)