DAR F23 Hockey Analytics

Hockey Analytics

Jeff Jung

2023-09-29

Contents

Analysis: Puck Dist by Outcome

Question being asked

puckDist variable is the most important variable in terms of relative performance used to estimating the outcome of goal. So, I have decided to further analyze on how puckDist data is different between saves and goals.

Data Preparation

Overall data preparation is the same as I used the same hockeyTrain variable, but in order to differentiate the puckDist by outcomes, I prepared two different datasets for goals and saves. I have also included the ggplot for every feature in this notebook (see below).

```
# Include all data processing code (if necessary), clearly commented

# Install required packages
r = getOption("repos")
r["CRAN"] = "http://cran.rstudio.com"
options(repos = r)

if (!require("jpeg")) {
   install.packages("jpeg")
}
```

```
## Loading required package: jpeg
```

```
if (!require("grid")) {
   install.packages("grid")
}
```

```
## Loading required package: grid
```

```
if (!require("scales")) {
   install.packages("scales")
}
```

```
## Loading required package: scales
```

```r
if (!require("reshape2")) {
   install.packages("reshape2")
}
```

```
## Loading required package: reshape2
```

```r
if (!require("tidyverse")) {
   install.packages("tidyverse")
}
```

```
## Loading required package: tidyverse

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x readr::col_factor() masks scales::col_factor()
## x purrr::discard()    masks scales::discard()
## 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 errors
```

```r
if (!require("tidymodels")) {
   install.packages("tidymodels")
}
```

```
## Loading required package: tidymodels
## -- Attaching packages -------------------------------------- tidymodels 1.1.1 --
## v broom        1.0.5     v rsample      1.2.0
## v dials        1.2.0     v tune         1.1.2
## v infer        1.0.5     v workflows    1.1.3
## v modeldata    1.2.0     v workflowsets 1.0.1
## v parsnip      1.1.1     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 recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
## * Learn how to get started at https://www.tidymodels.org/start/
```

```r
if (!require("ggnewscale")) {
   install.packages("ggnewscale")
}
```

```
## Loading required package: ggnewscale
```

```r
if (!require("glmnet")) {
   install.packages("glmnet")
}
```

```
## Loading required package: glmnet
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
## Loaded glmnet 4.1-8
```

```r
if (!require("MLmetrics")) {
  install.packages("MLmetrics")
}
```

```
## Loading required package: MLmetrics
##
## Attaching package: 'MLmetrics'
##
## The following object is masked from 'package:base':
##
##     Recall
```

```r
if (!require("knitr")) {
  install.packages("knitr")
}
```

```
## Loading required package: knitr
```

```r
if (!require("knitr")) {
  install.packages("knitr")
}
if (!require("magrittr")) {
  install.packages("magrittr")
}
```

```
## Loading required package: magrittr
##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
# Plotting
library(jpeg)
library(grid)
library(ggnewscale)
library(scales)
# Goal shot stats
library(reshape2)
library(knitr)
library(tidyverse)
library(tidymodels)
```

```r
library(magrittr)

# All user-defined functions are contained in the following helper script file.
source("../../AnalysisCodeFunc.R")

# Size of rink image and of all plots
xsize <- 2000
ysize <- 850

# FPS of the video
fps <- 29.97

# Coordinates to the goal pipes
pipes_x <- 1890
lpipe_y <- 395
rpipe_y <- 455

# This file path should contain the hockey rink images and all the sequences
filepath <- '../../FinalGoalShots/'

# See above for explanation of file path syntax
games <- c(24, 27, 33, 34)
# Only take the first and third periods. These are when the opposing team shoots on our goal. Our shots
periods <- map(games, ~ str_c(., 'p', c(1, 3))) %>% unlist

# Get the 'Sequences' folder for every period
period_folders <- map(periods, ~ {
  str_c(filepath, ., '/Sequences')
})

# Get every folder inside each 'Sequences' folder
sequence_folders <- period_folders %>%
  map(~ str_c(., '/', list.files(.))) %>%
  unlist

# Read the rink images and format them to a raster used for graphing
rink_raster <- makeRaster(filepath, 'Rink_Template.jpeg')
half_rink_raster <- makeRaster(filepath, 'Half_Rink_Template.jpeg')

# As every folder is run through the `combinePasses` function, the info.csv file in each sequence folde
info <- matrix(0, nrow = 0, ncol = 4) %>%
  data.frame %>%
  set_names(c('possessionFrame', 'shotFrame', 'outcome', 'rightHanded'))

# Read in all the sequences
# NOTE: This step takes a long time (minutes)
sequences = sequence_folders %>% map(combinePasses)
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
```

```
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## New names:
## * `` -> `...1`
```

```r
# Change outcomes to more verbose names
info$outcome %<>% fct_recode(Goal = 'G', Save = 'GB', 'Defender Block' = 'DB', Miss = 'M')

# Get stats for the shot in every sequence
shots_stats.df <- seq_along(sequences) %>%
  map_dfr(goalShotStats) %>%
  # Some models can't use logical data
  mutate_if(is.logical, as.factor)

# Split data into training and validation sets
outcomes.goal <- (info$outcome == 'Goal') %>% as.numeric %>% as.factor

# Append to shots_stats.df
shots_stats_goal.df <- cbind(shots_stats.df, outcomes.goal)
```

```r
# Save this dataframe on the file system in case we want to simply load it later (to save time)
saveRDS(shots_stats_goal.df, "shots_stats_goal.df.Rds")

#Create training set
set.seed(100)

# Type ?initial_split , ?training , or ?testing in the R console to see how these work!
hockey_split <- initial_split(shots_stats_goal.df, prop = 0.8)
hockeyTrain <- training(hockey_split)
hockeyTest <- testing(hockey_split)

# Check how many observations for each split we have
nrow(hockeyTrain)
```

```
## [1] 84
```

```r
nrow(hockeyTest)
```

```
## [1] 21
```

```r
# How many features are there
ncol(hockeyTrain)
```

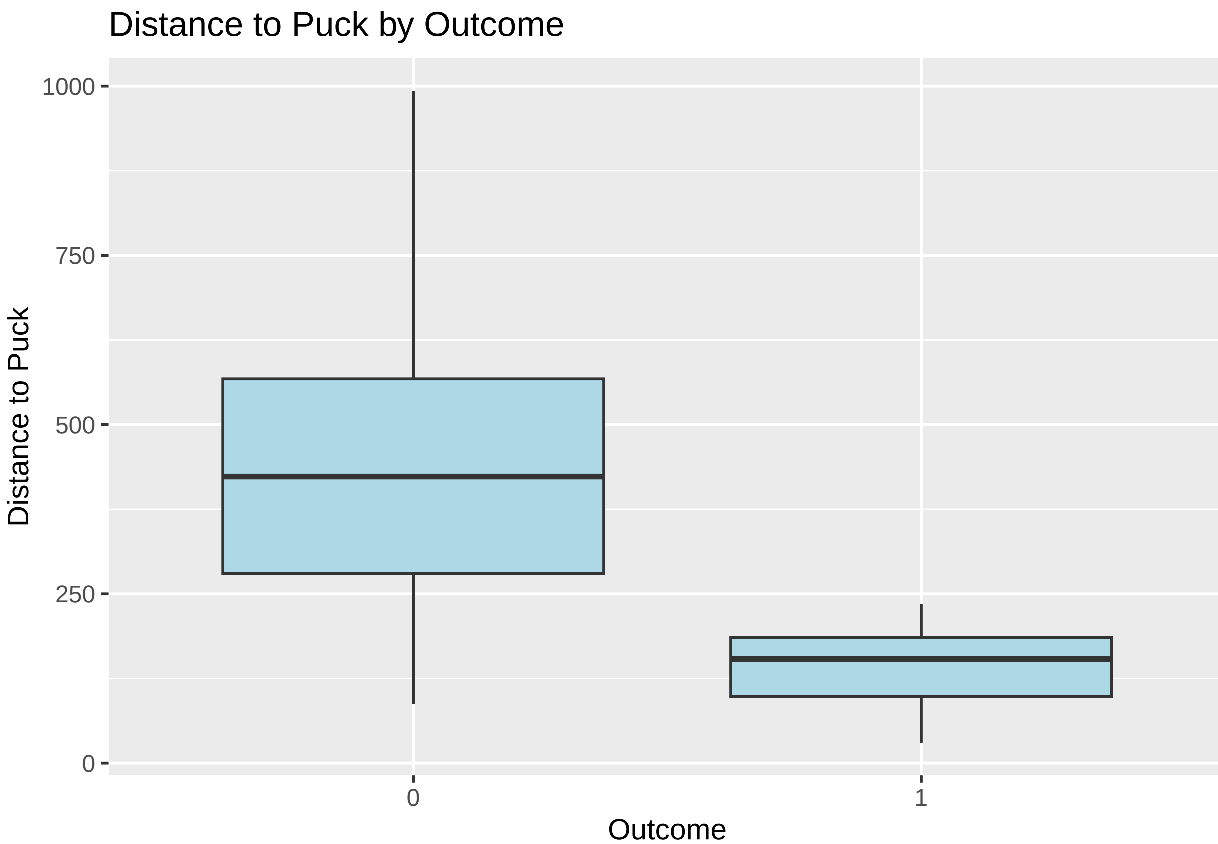
```
## [1] 12
```

```r
# Subset the data for not goals (e.g., "Save")
saves_data <- hockeyTrain[hockeyTrain$outcomes.goal != "1", ]

# Subset the data for goals
goals_data <- hockeyTrain[hockeyTrain$outcomes.goal == "1", ]
```

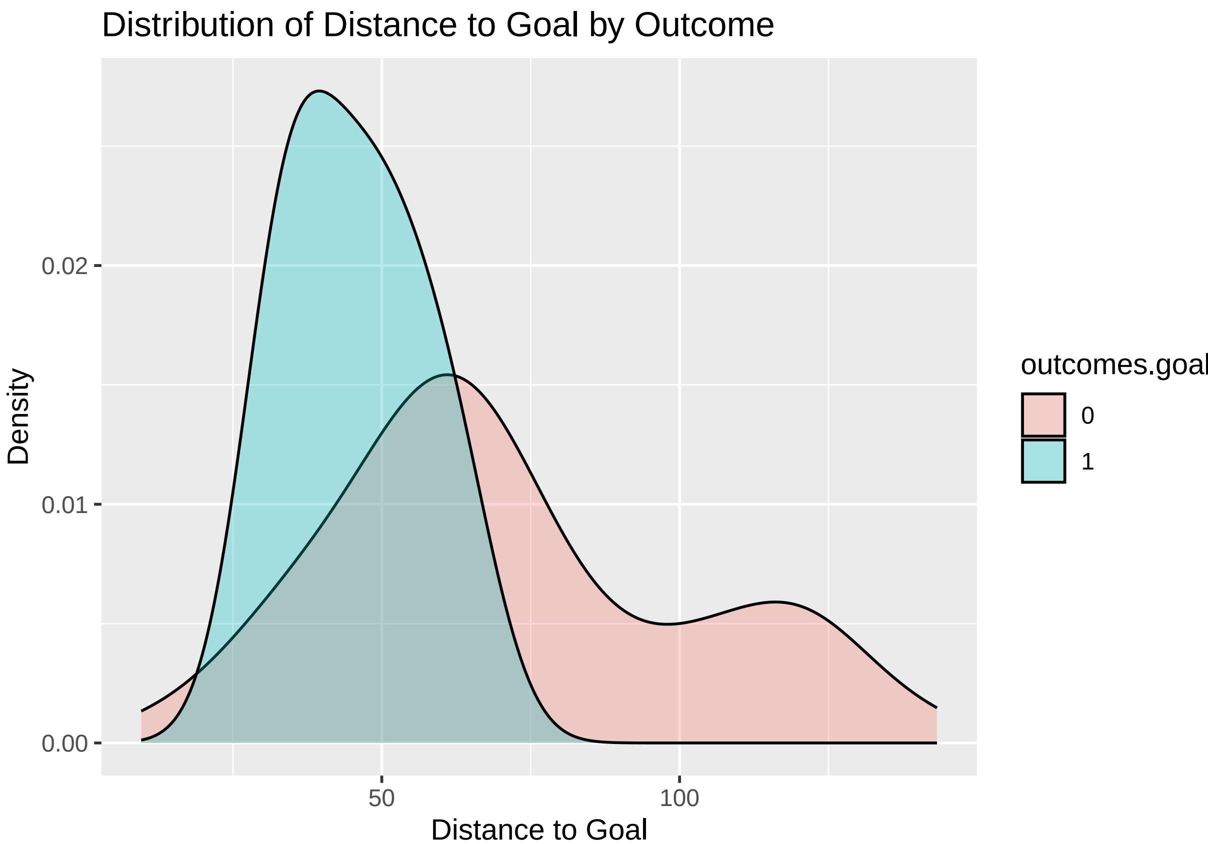
Analysis: Methods and results

I used ggplot to show the difference between goals and saves. For the most part, density plot is very effective in visualization as it is very intuitively illustrated, but boxplot is better in showing how the data is distributed more specifically. And for that purpose, I have also included the boxplot. The standard deviationa and median are also calculated.

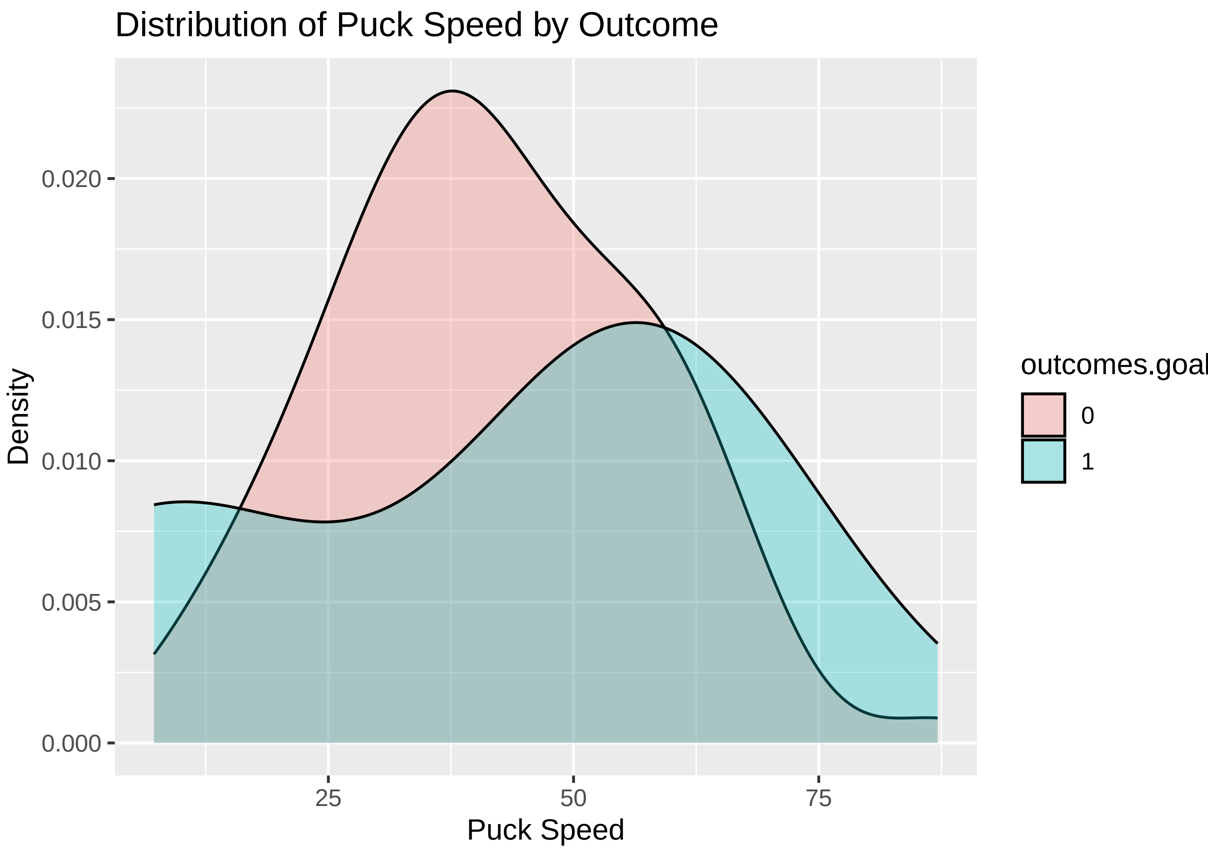
```r
# Include all analysis code, clearly commented
# If not possible, screen shots are acceptable.
# If your contributions included things that are not done in an R-notebook,
#   (e.g. researching, writing, and coding in Python), you still need to do
#   this status notebook in R.  Describe what you did here and put any products
#   that you created in github. If you are writing online documents (e.g. overleaf
#   or google docs), you can include links to the documents in this notebook
#   instead of actual text.

# boxplot
distance_data <- hockeyTrain %>%
  select(puckDist, outcomes.goal)

ggplot(distance_data, aes(x = outcomes.goal, y = puckDist)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Distance to Puck by Outcome",
       x = "Outcome",
       y = "Distance to Puck")
```



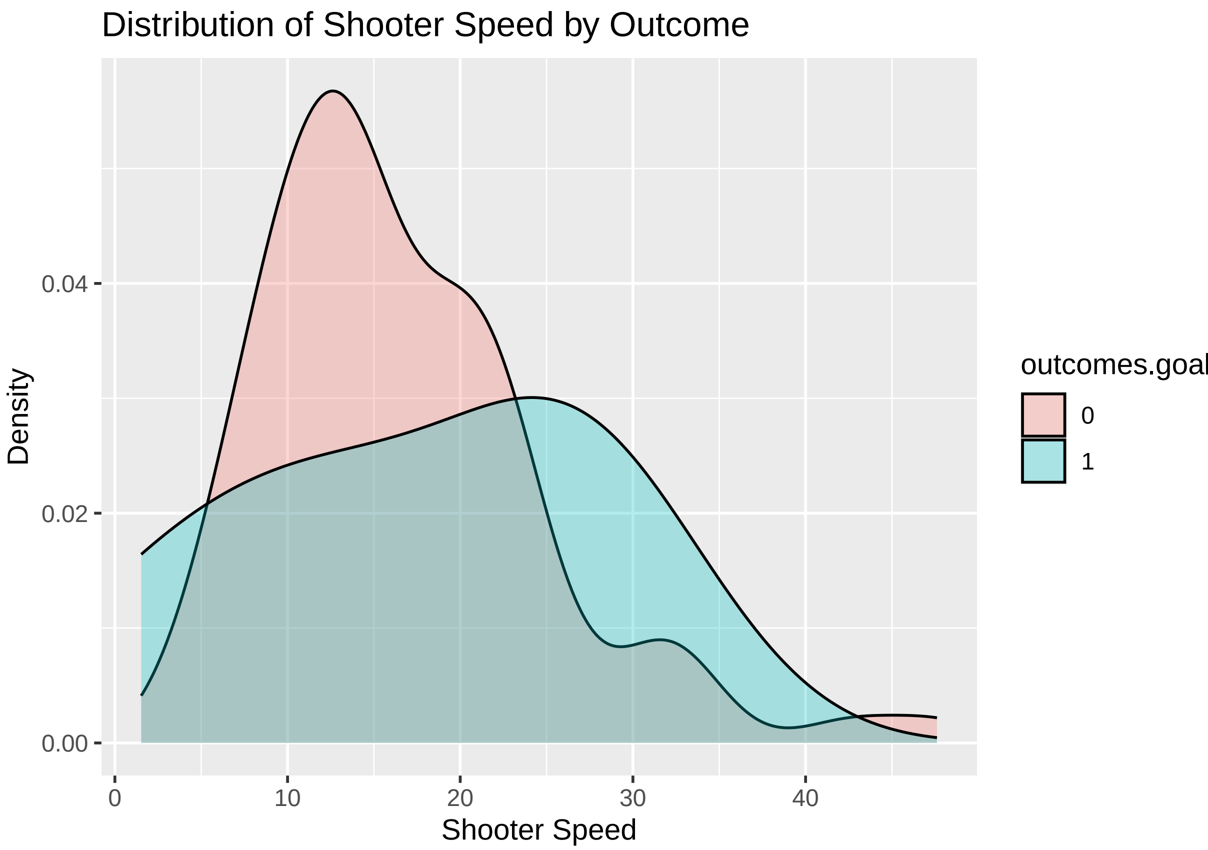

```r
# visualization using ggplot
ggplot(hockeyTrain, aes(x = goalieDist, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Distance to Goal by Outcome",
       x = "Distance to Goal",
       y = "Density")
```



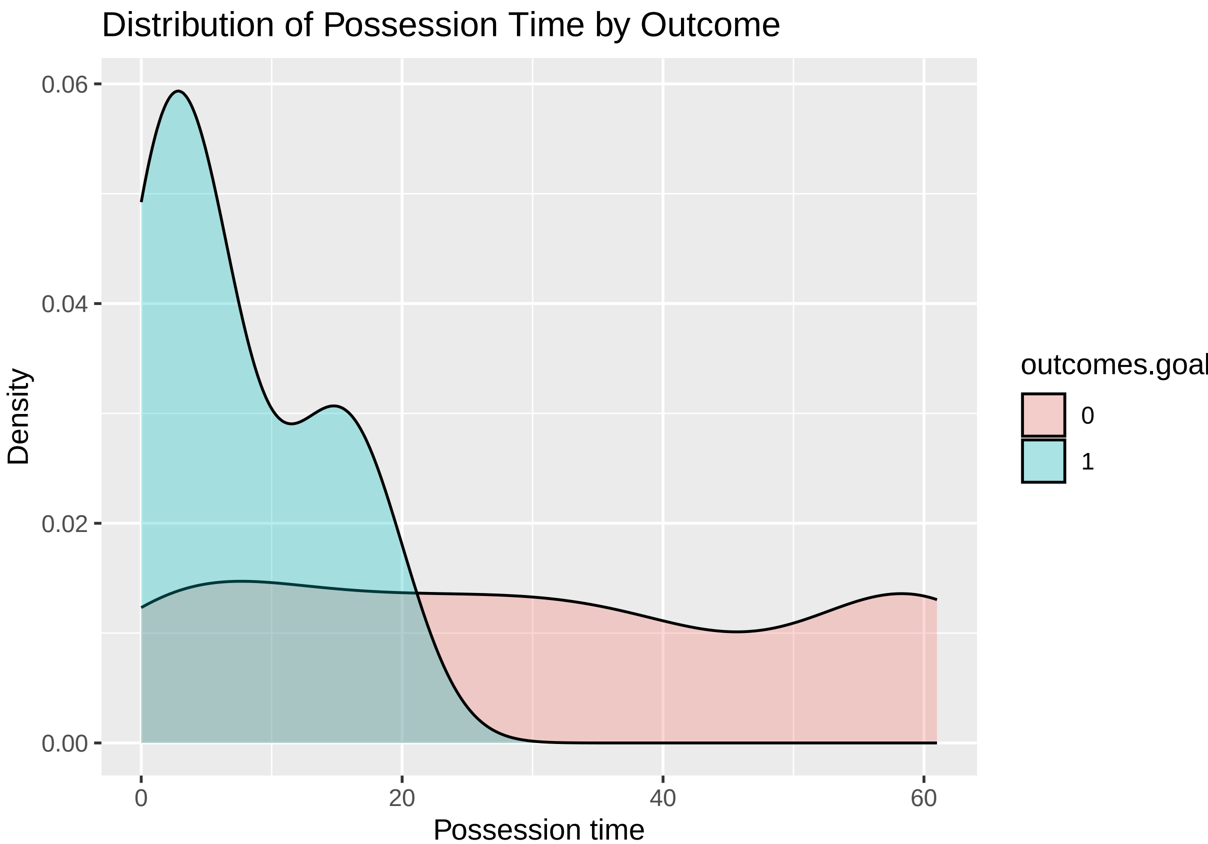

```r
ggplot(hockeyTrain, aes(x = puckSpeed, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Speed by Outcome",
       x = "Puck Speed",
       y = "Density")
```




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




```r
ggplot(hockeyTrain, aes(x = posTime, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Possession Time by Outcome",
       x = "Possession time",
       y = "Density")
```




```
# statistics for puckDist
# Calculate and print the median and standard deviation for goalieDist in goals
median_puck_dist_goals <- median(goals_data$puckDist)
sd_puck_dist_goals <- sd(goals_data$puckDist)

cat("Median Puck Distance for Goals:", median_puck_dist_goals, "\n")
```

```
## Median Puck Distance for Goals: 153.6167
```

```
cat("Standard Deviation of Puck Distance for Goals:", sd_puck_dist_goals, "\n")
```

```
## Standard Deviation of Puck Distance for Goals: 74.67696
```

```
# Calculate and print the median and standard deviation for goalieDist in not goals
median_puck_dist_saves <- median(saves_data$puckDist)
sd_puck_dist_saves <- sd(saves_data$puckDist)

cat("Median Goalie Puck for Saves:", median_puck_dist_saves, "\n")
```

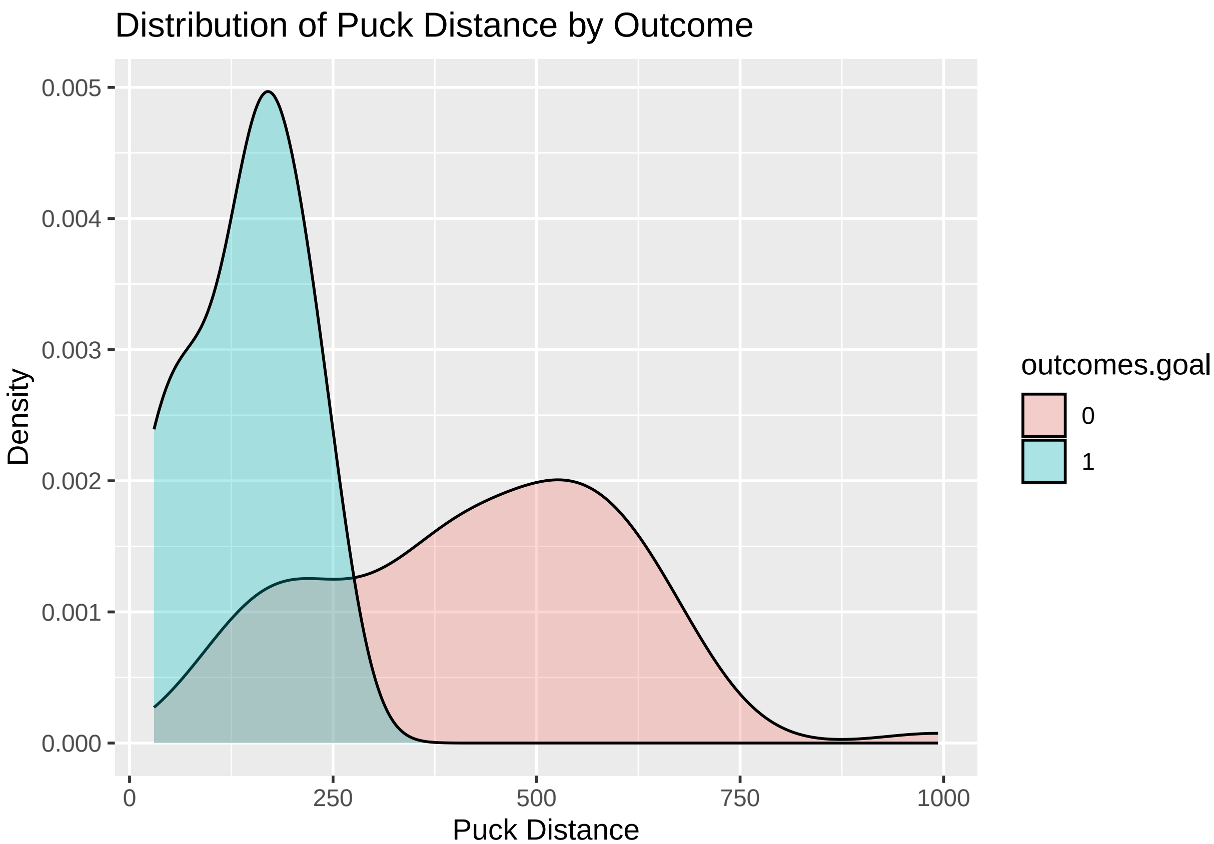
```
## Median Goalie Puck for Saves: 423.2627
```

```
cat("Standard Deviation of Goalie Puck for Saves:", sd_puck_dist_saves, "\n")
```

```
## Standard Deviation of Goalie Puck for Saves: 183.1079
```

```
# density plot for puckDist
ggplot(hockeyTrain, aes(x = puckDist, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Distance by Outcome",
       x = "Puck Distance",
```

```
       y = "Density")
```



Discussion of results

From both the statistics and diagram of puck distance, we can see that there is a significant distinction between goals and saves. Successful shots were made when the distance between the player and the puck was small, and saves were made when the distance between the player and puck was large. And the difference between the two is very significant. Also, another interesting observation is that the standard deviation for saves was very large compared to that of goals. This tells that shots that led to goals were relatively stable in regards to shooting form or pressure put on the shooter. However, unsuccessful shots were unstable, and we can tell this by looking at how much the puck distance can vary for saves.

Analysis: Handedness of a Player

Question being asked

How does a handedness of a player related with goalieAngle and puckAngle?

Data Preparation

I divided the data into two by handedness of players, and I also used hockeyTrain variable as I did above.

```
# Include all data processing code (if necessary), clearly commented

# Subset the data for not goals (e.g., "Save")
righthand_data <- hockeyTrain[hockeyTrain$rightHanded == "1", ]
```

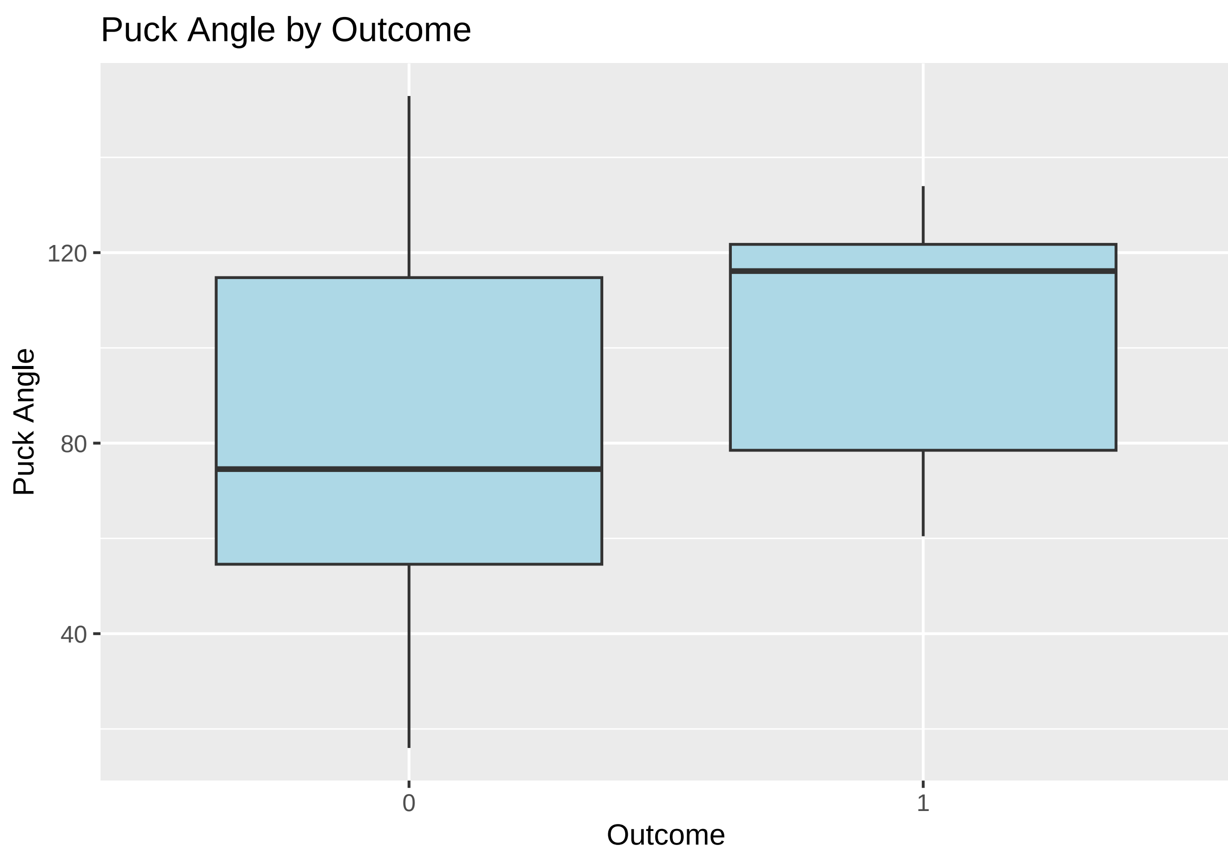
```r
# Subset the data for goals
lefthand_data <- hockeyTrain[hockeyTrain$rightHanded == "0", ]
```

Analysis: Methods and Results

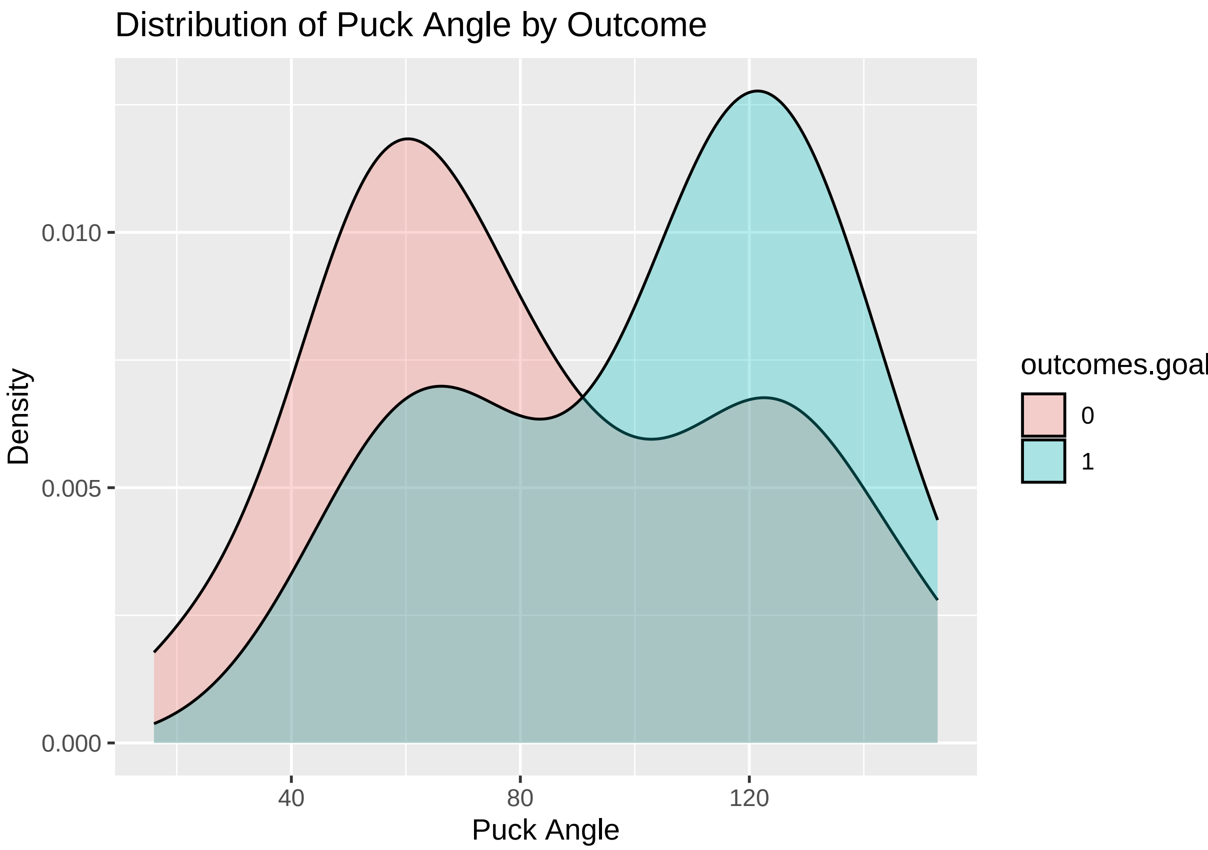
For puck angle, I made 3 different ggplots. The first one is for all players, the second one is for right handed players, and the third one is for left handed players. Also, I have calculated the median and standard deviation for each hand. This statistic is also illustrated in the boxplot.

Similarly, for goalie angle I made 3 different ggplots. The first one is for all players, the second one is for right handed players, and the third one is for left handed players. Also, I have calculated the median and standard deviation for each hand. This statistic is also illustrated in the boxplot.

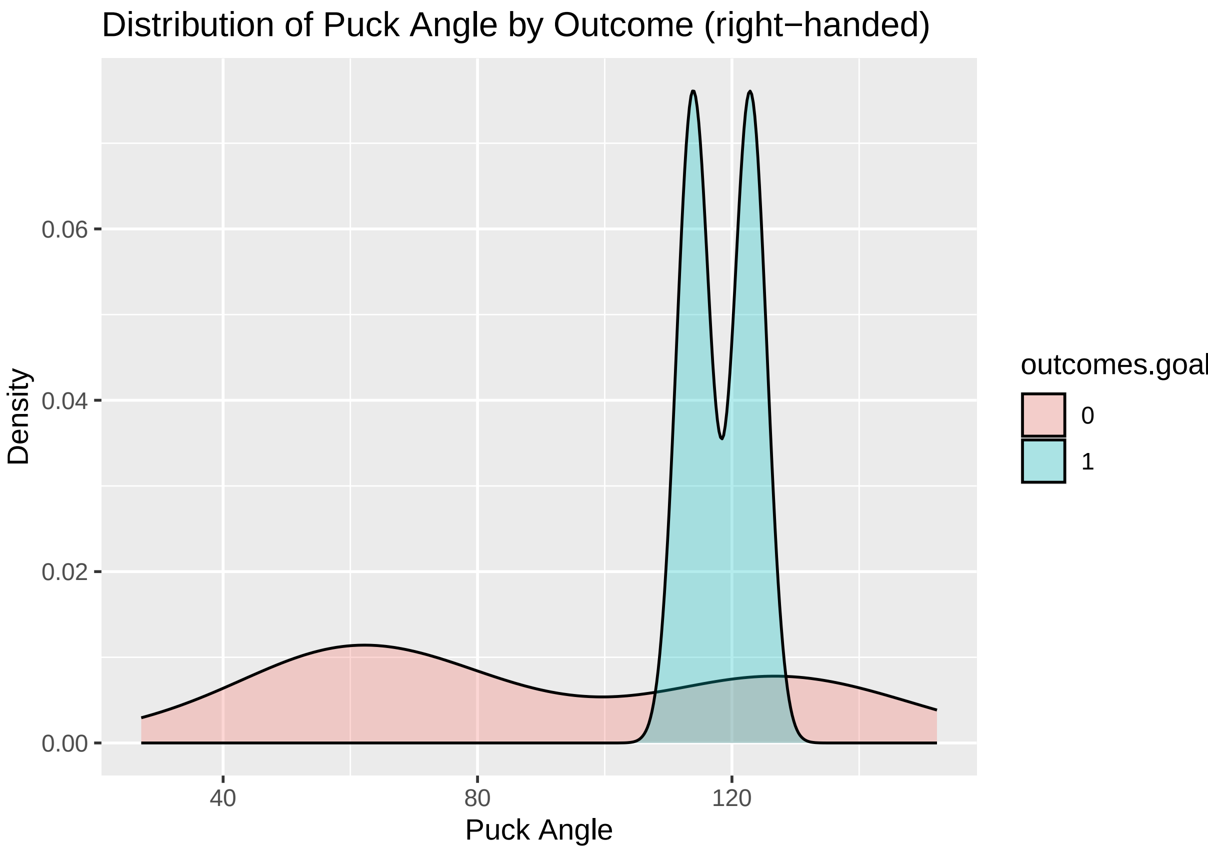
```r
# Include all analysis code, clearly commented
# If not possible, screen shots are acceptable.
# If your contributions included things that are not done in an R-notebook,
#   (e.g. researching, writing, and coding in Python), you still need to do
#   this status notebook in R.  Describe what you did here and put any products
#   that you created in github. If you are writing online documents (e.g. overleaf
#   or google docs), you can include links to the documents in this notebook
#   instead of actual text.

# boxplot for puckAngle
distance_data <- hockeyTrain %>%
  select(puckAngle, outcomes.goal)

ggplot(distance_data, aes(x = outcomes.goal, y = puckAngle)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Puck Angle by Outcome",
       x = "Outcome",
       y = "Puck Angle")
```



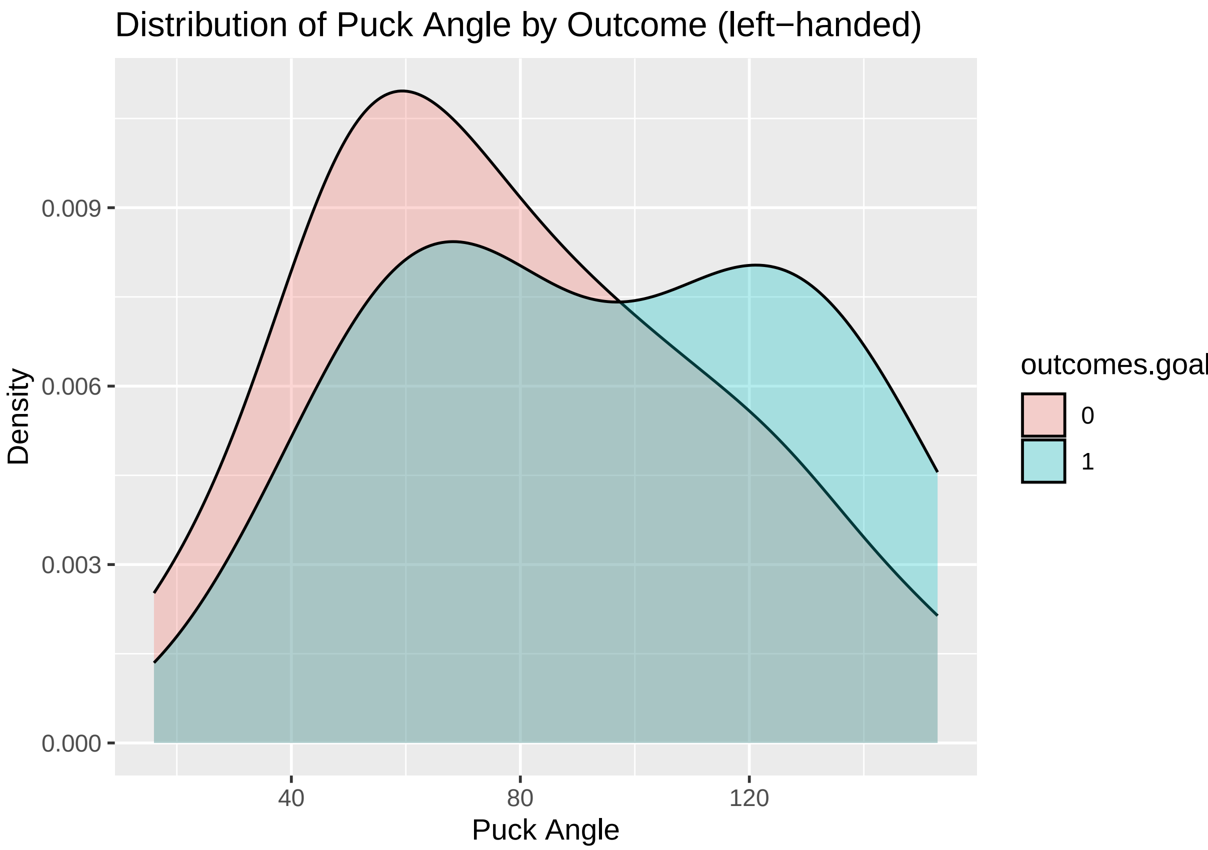

```
# Distribution of goalieAngle differentiated by handedness of players using density plot
ggplot(hockeyTrain, aes(x = puckAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Angle by Outcome",
       x = "Puck Angle",
       y = "Density")
```



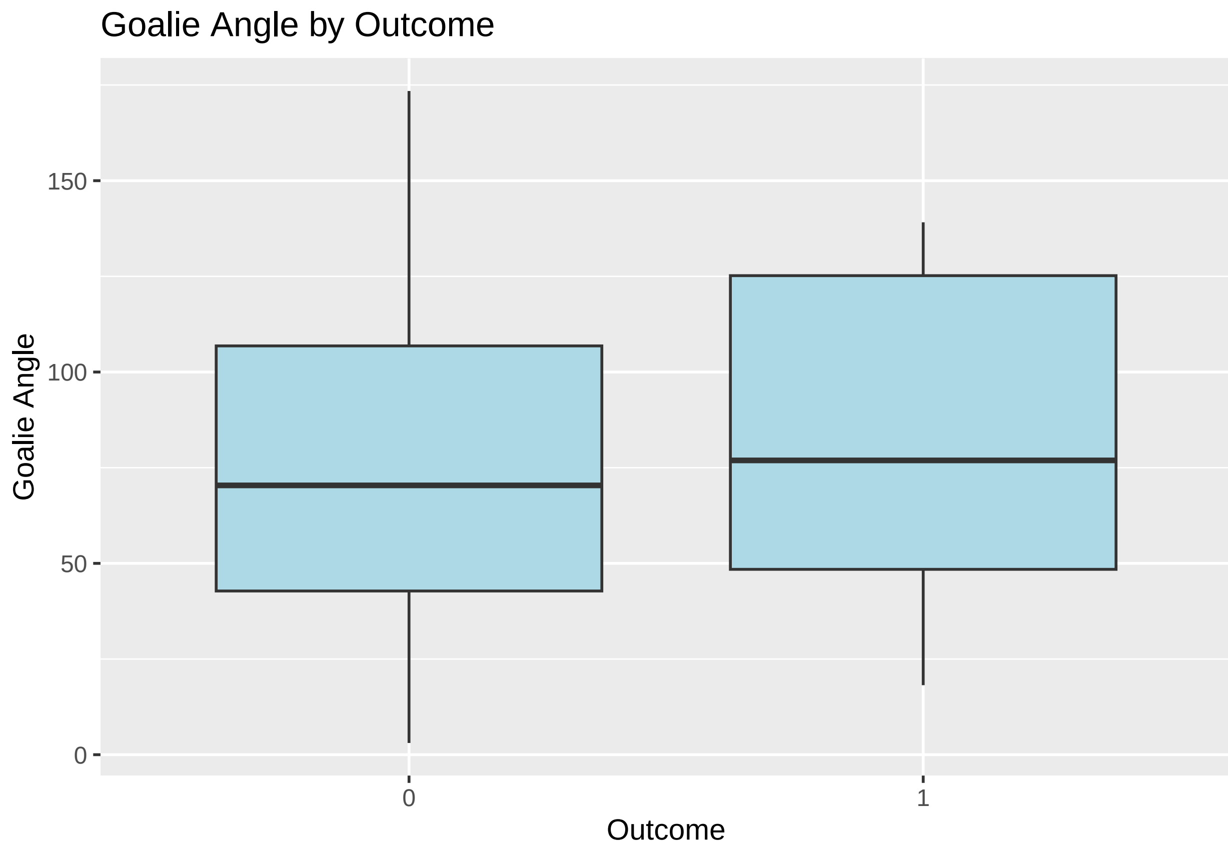

```r
ggplot(righthand_data, aes(x = puckAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Angle by Outcome (right-handed)",
       x = "Puck Angle",
       y = "Density")
```



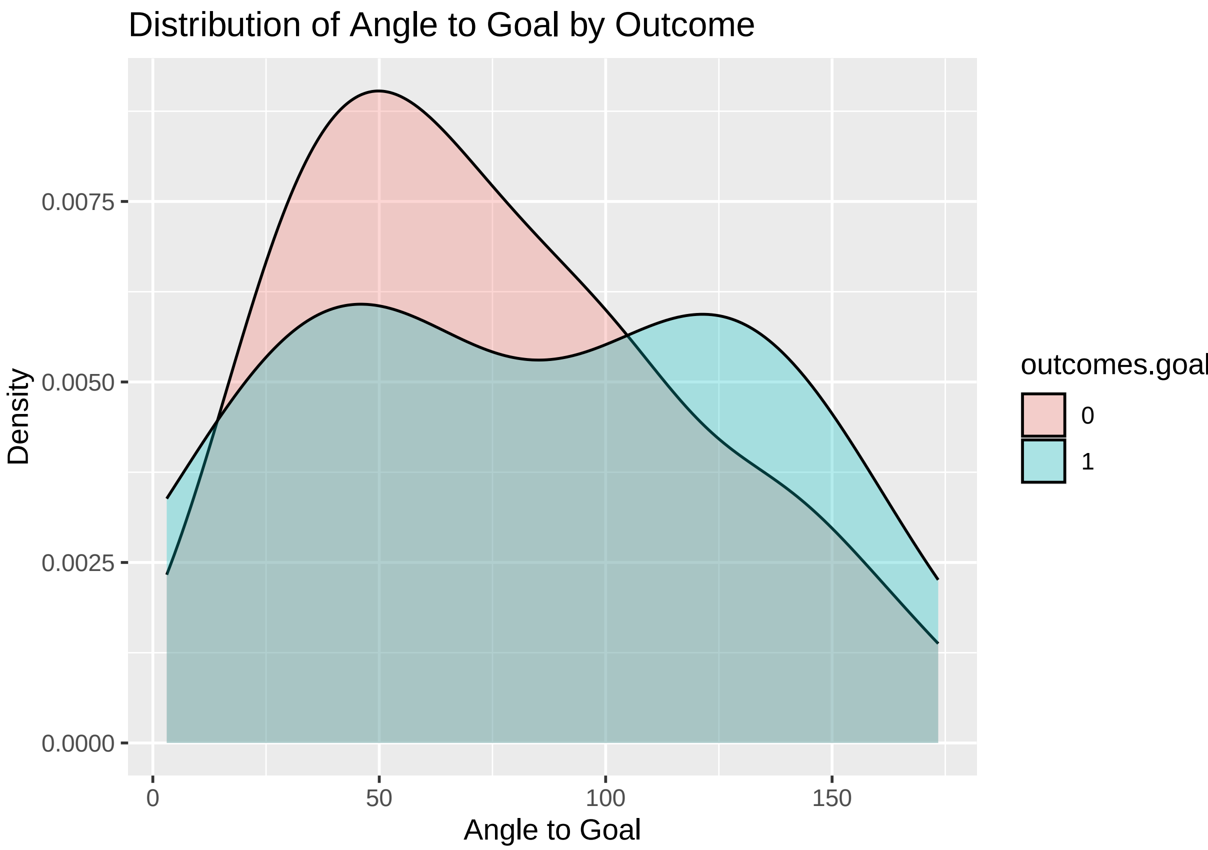

```
ggplot(lefthand_data, aes(x = puckAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Puck Angle by Outcome (left-handed)",
       x = "Puck Angle",
       y = "Density")
```



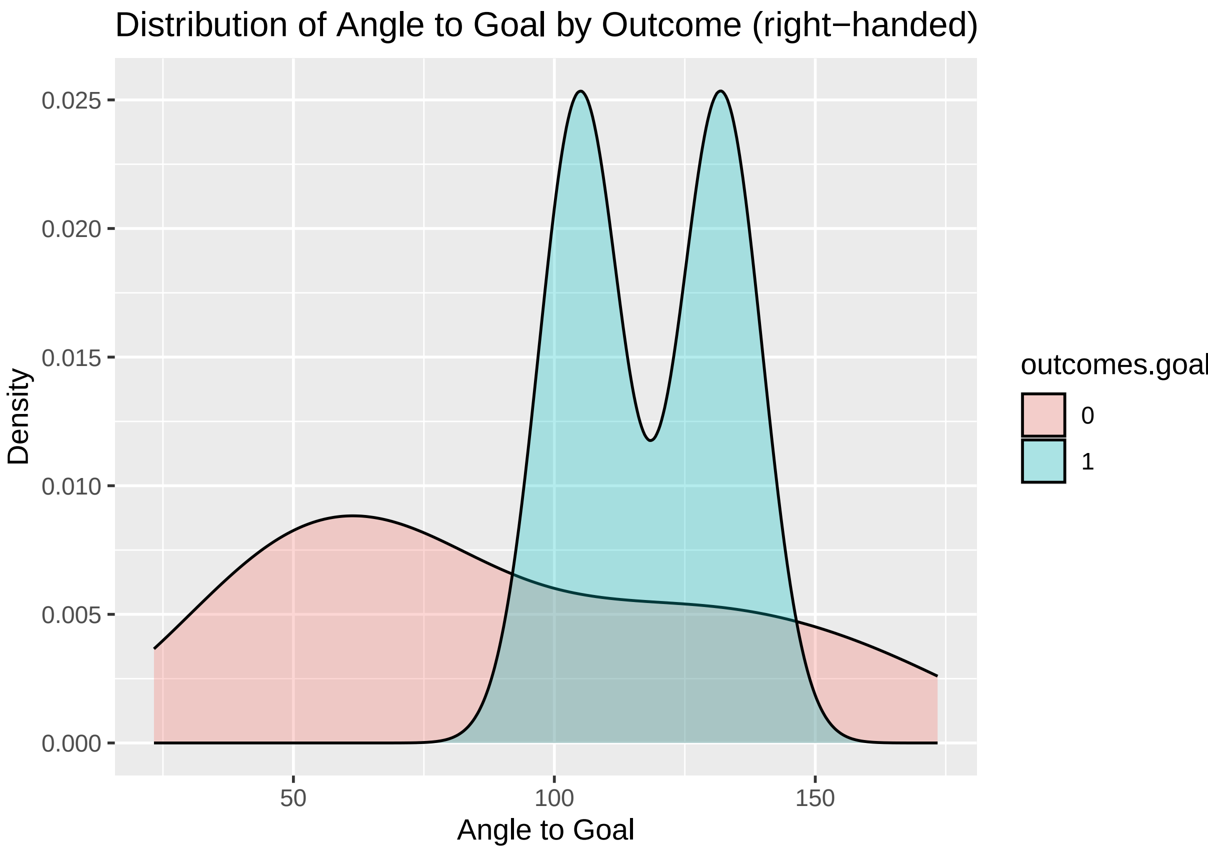

```
# boxplot for goalieAngle
distance_data <- hockeyTrain %>%
  select(goalieAngle, outcomes.goal)

ggplot(distance_data, aes(x = outcomes.goal, y = goalieAngle)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Goalie Angle by Outcome",
       x = "Outcome",
       y = "Goalie Angle")
```



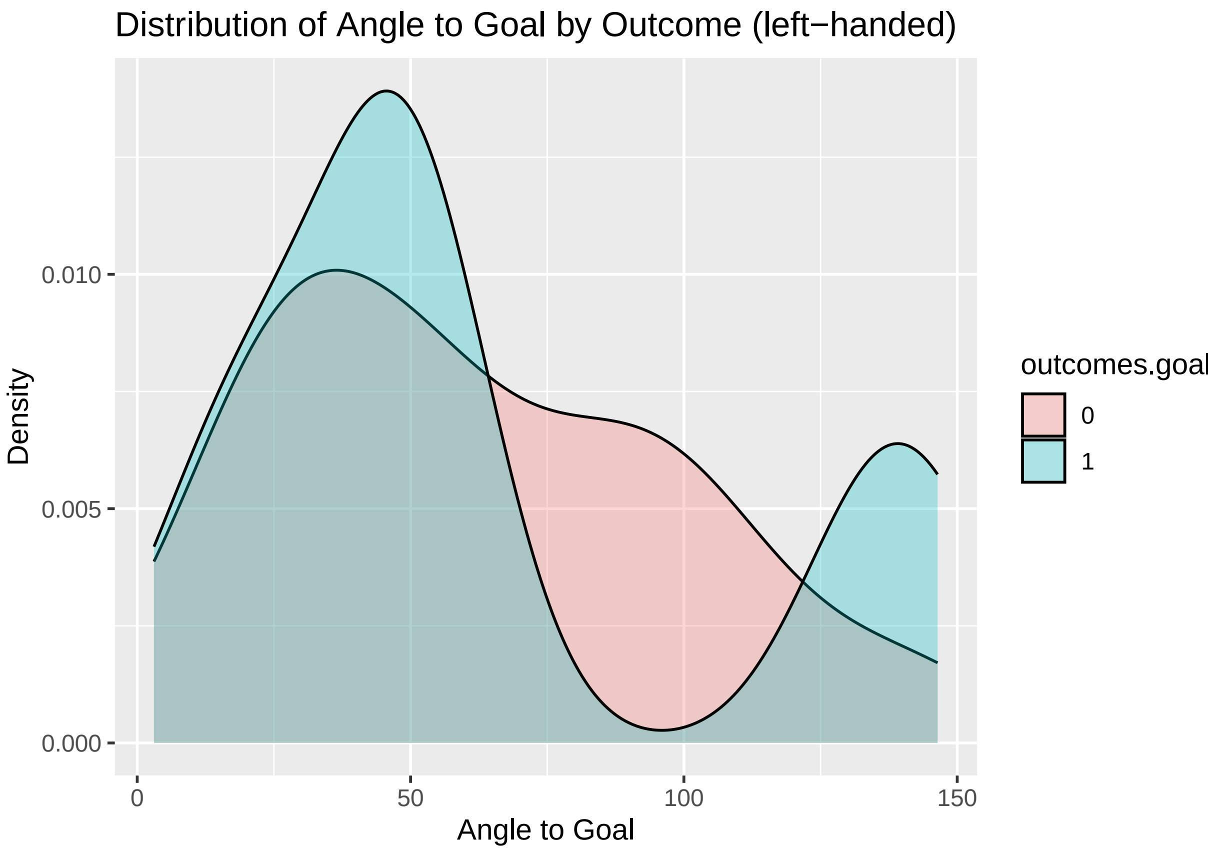

```
# Distribution of goalieAngle differentiated by handedness of players using density plot
ggplot(hockeyTrain, aes(x = goalieAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Angle to Goal by Outcome",
       x = "Angle to Goal",
       y = "Density")
```




```r
ggplot(righthand_data, aes(x = goalieAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Angle to Goal by Outcome (right-handed)",
       x = "Angle to Goal",
       y = "Density")
```




```r
ggplot(lefthand_data, aes(x = goalieAngle, fill = outcomes.goal)) +
  geom_density(alpha = 0.3) +
  labs(title = "Distribution of Angle to Goal by Outcome (left-handed)",
       x = "Angle to Goal",
       y = "Density")
```




```r
# statistics for puckDist
# Calculate and print the median and standard deviation for goalieDist in goals
median_right_puckAngle_goals <- median(righthand_data$puckAngle)
sd_right_puckAngle_goals <- sd(righthand_data$puckAngle)

cat("Median Puck Angle for Right-handed Players:", median_right_puckAngle_goals, "\n")
```

```
## Median Puck Angle for Right-handed Players: 80.0173
```

```r
cat("Standard Deviation of Puck Angle for Right-handed Players:", sd_right_puckAngle_goals, "\n")
```

```
## Standard Deviation of Puck Angle for Right-handed Players: 35.54207
```

```r
# Calculate and print the median and standard deviation for goalieDist in not goals
median_left_puckAngle_goals <- median(lefthand_data$puckAngle)
sd_left_puckAngle_goals <- sd(lefthand_data$puckAngle)

cat("Median Puck Angle for Left-handed Players:", median_left_puckAngle_goals, "\n")
```

```
## Median Puck Angle for Left-handed Players: 73.37532
```

```r
cat("Standard Deviation of Puck Angle for Left-handed Players:", sd_left_puckAngle_goals, "\n")
```

```
## Standard Deviation of Puck Angle for Left-handed Players: 34.88267
```

```r
# statistics for goalieAngle
# Calculate and print the median and standard deviation for goalieDist in goals
median_right_goalieAngle_goals <- median(righthand_data$goalieAngle)
sd_right_goalieAngle_goals <- sd(righthand_data$goalieAngle)
```

```r
cat("Median Goalie Angle for Right-handed Players:", median_right_goalieAngle_goals, "\n")
```

```
## Median Goalie Angle for Right-handed Players: 78.16623
```

```r
cat("Standard Deviation of Goalie Angle for Right-handed Players:", sd_right_goalieAngle_goals, "\n")
```

```
## Standard Deviation of Goalie Angle for Right-handed Players: 42.29154
```

```r
# Calculate and print the median and standard deviation for goalieDist in not goals
median_left_goalieAngle_goals <- median(lefthand_data$goalieAngle)
sd_left_goalieAngle_goals <- sd(lefthand_data$goalieAngle)

cat("Median Goalie Angle for Left-handed Players:", median_left_goalieAngle_goals, "\n")
```

```
## Median Goalie Angle for Left-handed Players: 52.88144
```

```r
cat("Standard Deviation of Goalie Angle for Left-handed Players:", sd_left_goalieAngle_goals, "\n")
```

```
## Standard Deviation of Goalie Angle for Left-handed Players: 38.16943
```

```r
# Independent two-sample t-test (assuming data is normally distributed)
t_test_puckAngle <- t.test(lefthand_data$puckAngle, righthand_data$puckAngle)

# Print the results
cat("T-test for Puck Angle:\n")
```

```
## T-test for Puck Angle:
```

```r
print(t_test_puckAngle)
```

```
##
##  Welch Two Sample t-test
##
## data:  lefthand_data$puckAngle and righthand_data$puckAngle
## t = -1.2349, df = 79.861, p-value = 0.2205
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -24.862134   5.822016
## sample estimates:
## mean of x mean of y
##  80.05023  89.57029
```

```r
# Independent two-sample t-test (assuming data is normally distributed)
t_test_goalieAngle <- t.test(lefthand_data$goalieAngle, righthand_data$goalieAngle)

# Print the results
cat("T-test for Goalie Angle:\n")
```

```
## T-test for Goalie Angle:
```

```r
print(t_test_goalieAngle)
```

```
##
##  Welch Two Sample t-test
##
## data:  lefthand_data$goalieAngle and righthand_data$goalieAngle
## t = -3.3229, df = 77.314, p-value = 0.001363
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
```

```
##  -47.00293 -11.77935
## sample estimates:
## mean of x mean of y
##  62.29246  91.68360
```

Discussion of results

Based on observations from the density plots, it seems like that there is a staitiscal difference between right handed and left handed for puckAngle, but further analyzing using t-test, there is no statistically signifcant difference. The reason why the density plot shows a significant difference could be because the sample size is small. However, for goalie angle, there is a statistically significant difference between right handed and left handed data with left handed mean goalie angle of 62.29246 and right handed mean goalie angle of 91.68360. This means that left handed players are more likely to shoot from the left side of the goalie, and right handed players are more likely to shoot from the right side of the goalie.

Analysis: Best Puck Speed

Question being asked

At what range of puck speed the best shots are made?

Data Preparation

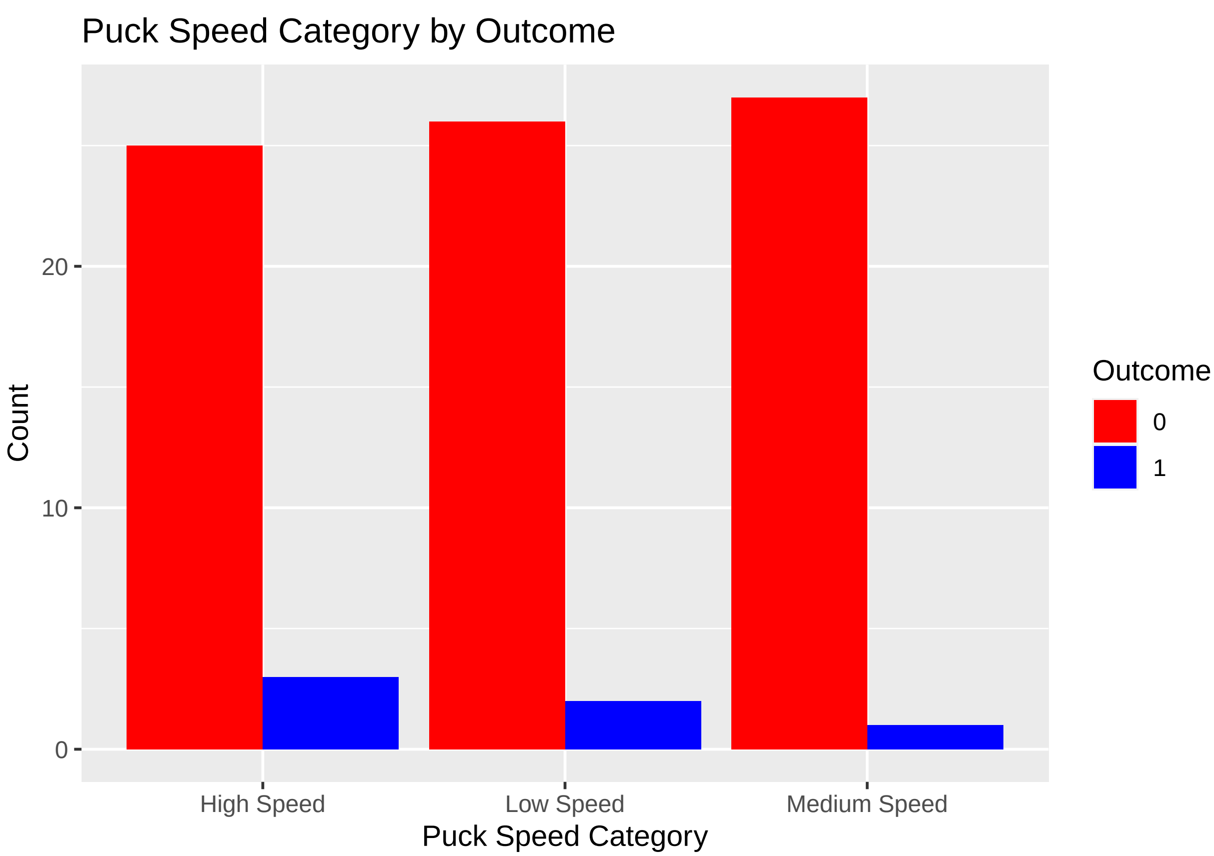
I categorized puck speed into 3 different categories: high speed, medium speed, and low speed. There are the same number of data in each category. Also, the data is grouped so that we can show the average, number of shots for each category, and the chances of goal for each category can be shown.

```r
# Include all data processing code (if necessary), clearly commented

# Load necessary libraries
library(dplyr)

# Calculate quantiles to divide the data into thirds
quantiles <- quantile(hockeyTrain$puckSpeed, probs = c(1/3, 2/3))

# Create three categories based on quantiles
hockeyTrain <- hockeyTrain %>%
  mutate(puckSpeedCategory = case_when(
    puckSpeed <= quantiles[1] ~ "Low Speed",
    puckSpeed <= quantiles[2] ~ "Medium Speed",
    TRUE ~ "High Speed"
  ))

# Group the data by puckSpeedCategory and calculate the mean for each group
average_by_category <- hockeyTrain %>%
  group_by(puckSpeedCategory) %>%
  summarise(average = mean(puckSpeed))

# Calculate the chance of a goal for each speed category
chance_of_goal <- hockeyTrain %>%
  group_by(puckSpeedCategory) %>%
  summarise(
    total_shots = n(),
    goals = sum(outcomes.goal == "1"),
    chance_of_goal = goals / total_shots
  )
```

Analysis methods used

Data categorizationa and calculation are done in data preparation. I chose barplot as it works best for categorical data. The mean values are calculated for each group of puck speed as well as the chances of goal and count.

```
# Include all analysis code, clearly commented
# If not possible, screen shots are acceptable.
# If your contributions included things that are not done in an R-notebook,
#   (e.g. researching, writing, and coding in Python), you still need to do
#   this status notebook in R.  Describe what you did here and put any products
#   that you created in github. If you are writing online documents (e.g. overleaf
#   or google docs), you can include links to the documents in this notebook
#   instead of actual text.

# Create a bar plot for puckSpeedCategory by outcome
ggplot(hockeyTrain, aes(x = puckSpeedCategory, fill = outcomes.goal)) +
  geom_bar(position = "dodge") +
  labs(title = "Puck Speed Category by Outcome",
       x = "Puck Speed Category",
       y = "Count",
       fill = "Outcome") +
  scale_fill_manual(values = c("1" = "blue", "0" = "red"))
```




```
# Print the results
print(average_by_category)
```

```
## # A tibble: 3 x 2
##   puckSpeedCategory average
```

```
##   <chr>                <dbl>
## 1 High Speed            59.8
## 2 Low Speed             23.2
## 3 Medium Speed          41.1
```

```
# Print the results
print(chance_of_goal)
```

```
## # A tibble: 3 x 4
##   puckSpeedCategory total_shots goals chance_of_goal
##   <chr>                   <int> <int>          <dbl>
## 1 High Speed                 28     3         0.107
## 2 Low Speed                  28     2         0.0714
## 3 Medium Speed               28     1         0.0357
```

Discussion of results

Not surprisingly, high speed shots had the highest chance of leading to a goal. But we cannot confidently say that this is always true since our sample size is very small.

Summary and next steps

I can add more features to categorizing different shots, but since the sample size is too small (1 goal for low speed, 2 goals for medium speed, and 3 goals for high speed), I might need to find a different way to analyze this data. I will need to talk to professor Bennet about this so that if it is okay to make more models with a small data.