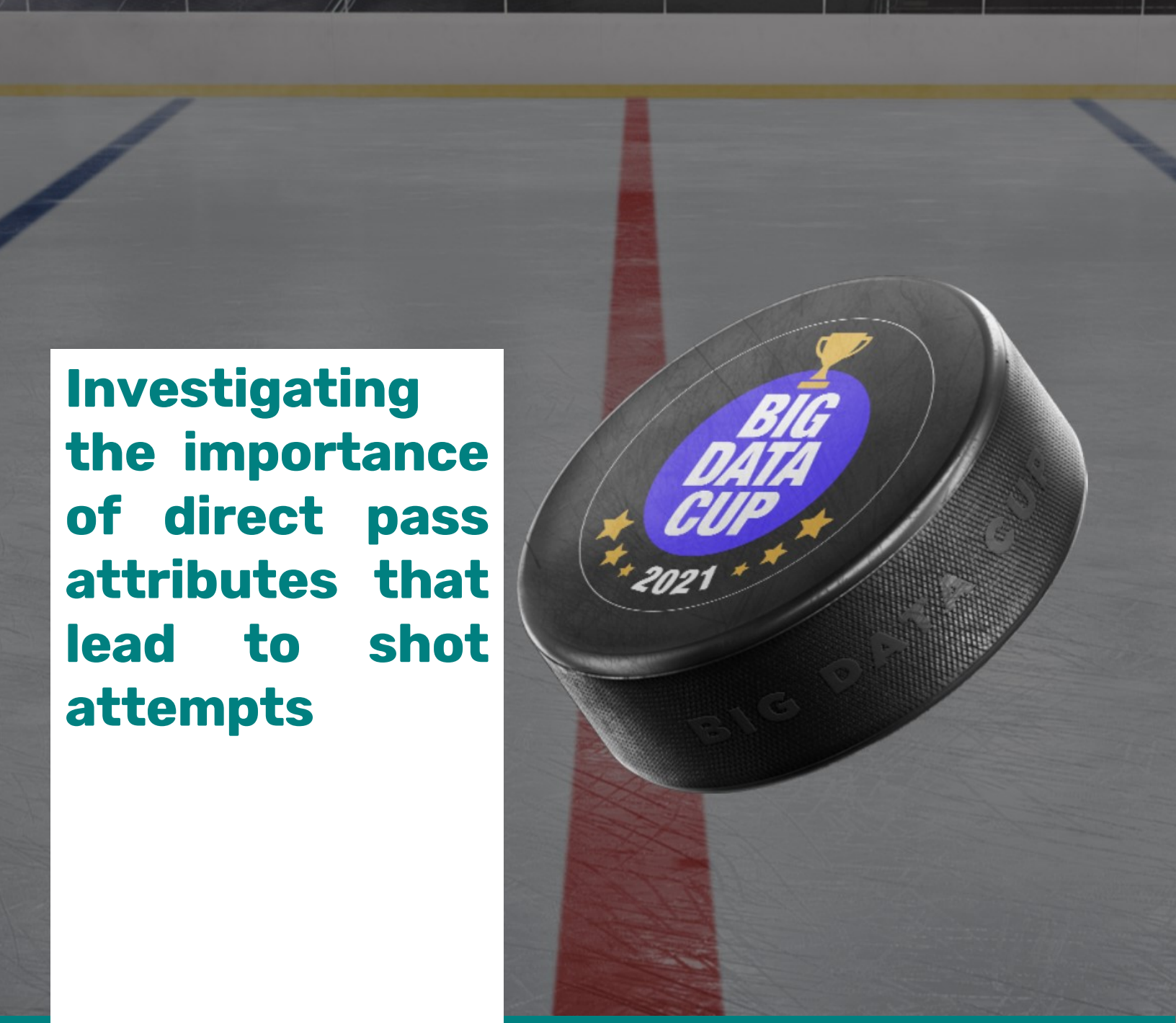


Investigating the importance of direct pass attributes that lead to shot attempts

Authored by:

Bill Tran,
Dave Scott,
Karim Kurji,
Raymond Guan

March 5, 2021

Introduction

Goals and shots dominate most analytical models currently used to evaluate players and teams. Widely available data such as shot distance and shot location have helped to define shot quality and give a better indication of which teams and players are better at generating opportunities that are more likely to score. However, there is more value that can be added with this area of the game.

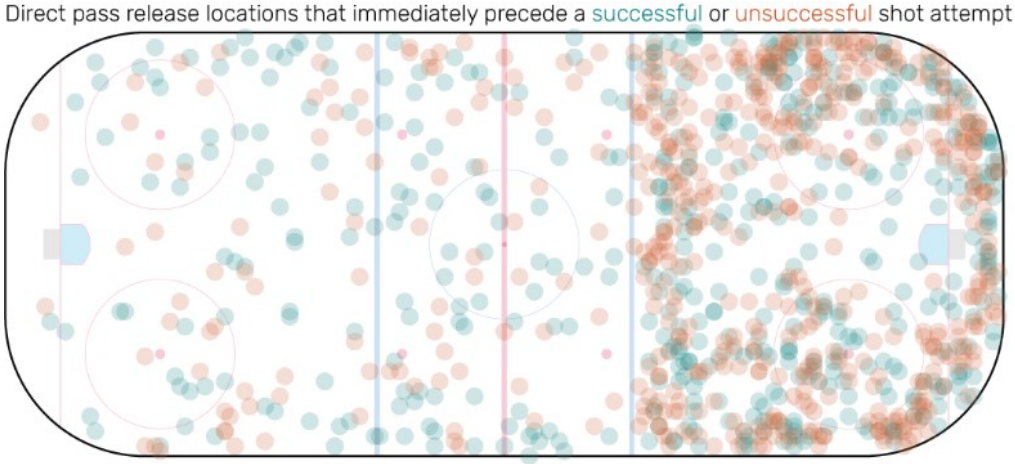
This is due to the limited data available for other parts of the game, including passing. The pass before a shot, or a shot assist, is a key data point in predicting the outcome of a shot. Even on the most basic level, it is intuitive that a bad pass can squander even the best scoring chance, and an excellent pass can create a scoring chance from nothing.

Because passing is such an important part of the game of hockey and is currently underrepresented in the hockey analytics field, we evaluated passing data to better understand what factors are the most important in leading to scoring chances. Having a better idea of how a shot assist affects the outcome of a shot can help teams and evaluators in numerous ways, including to help identify elite playmakers, find which types of passing plays are most effective in leading to true scoring opportunities, and enhance existing expected goals models.

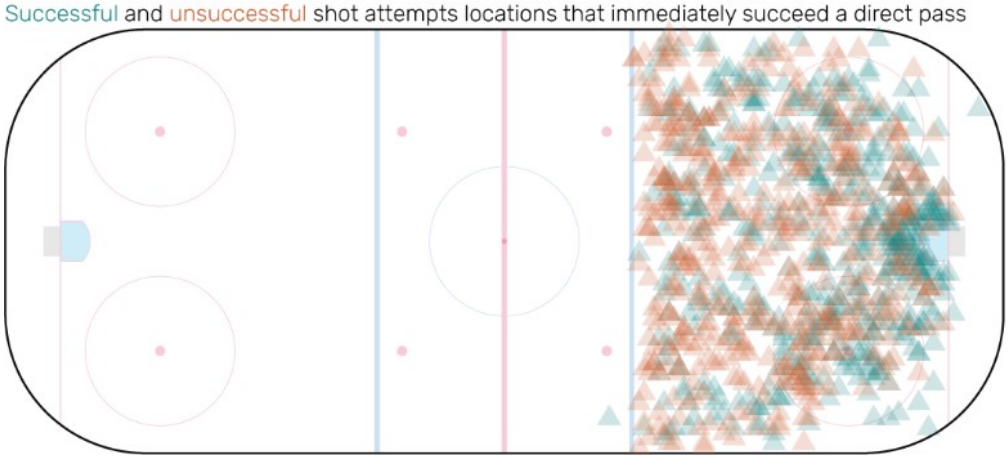
Data & Assumptions

The dataset chosen for the analysis was the National Women's Hockey League (NWHL) data set, however, the methodology used is applicable to all provided data sets. Using R, the data was processed to create a new data frame where each row contained complete event sequences. The event sequences contain information about direct passes and pass reception, the shot that follows, and contextual data about the events (either directly from the raw data or calculated from it). For this analysis, indirect passes were ignored due to insufficient information about puck and player movement between pass and reception.

Direct pass release locations that immediately precede a successful or unsuccessful shot attempt



Successful and unsuccessful shot attempts locations that immediately succeed a direct pass



The coordinate system for the NWHL data was modified such that the skating surface would be in both the positive x- and y-directions based on Cartesian coordinates, and the offensive zone would be contained over the domain of x: [125, 200] and over the range of y: [0, 85].

The processed dataset included the following new variables in the format variable_name, [type], description:

- game_seconds, [time], Seconds elapsed since opening faceoff, from 0 to 3900 to include overtime
- time_since_pass, [time], Seconds elapsed since direct pass was recorded
- game_state, [character], Team players on ice based on attacking team's perspective, "#v#" format
- passer, [text], Name of passer
- shooter, [text], Name of shooter
- pass_x, pass_y, [position], x, y coordinates of pass release location
- rec_x, rec_y, [position], x, y coordinates of pass reception location
- shot_x, shot_y, [position], x, y coordinates of shot release location
- shot_type, [text], Renamed from detail_1 for shot events
- shot_outcome, [text], Renamed from detail_2, goals labeled as "Goal" instead of "On Net"
- shot_traffic, [logical], Renamed from detail_3 for shot events
- shot_onetimer, [logical], Renamed from detail_4 for shot events
- pass_dist, [position], Euclidean distance between (pass_x, pass_y) and (rec_x, rec_y)
- pass_angle, [angle], Angle between positive x-axis and pass vector
- shooter_dist, [position], Euclidean distance between (rec_x, rec_y) and (shot_x, shot_y)
- shot_dist, [position], Euclidean distance between the shot and assumed centre of net at x, y = (189, 43)
- shot_angle, [angle], Angle between positive x-axis and the shot vector

Assumptions made on the data include:

- Centre ice is labelled as (100, 43) as observed from opening faceoffs, therefore the centre of the net in the offensive zone is consequently at (189, 43), used in shot distance calculations
- Direct passes travel in a straight line and are instantaneous, as the data does not discretely label time between pass release and pass reception
- shooter_dist is more accurately described as shooter displacement, as it is calculated as a straight line (which does not reflect skater movements)

The Goal

The analysis helps us further understand if and how measured pass attributes affect the success of shots, and to what extent each attribute affects shot outcome.

Shot generation may preside over player offensive evaluation tools right now, but the addition of pass and shot assist data will help to create better analyses on different player types, including differentiating playmakers and finishers.

Model & Methodology

Setting Up

Following data processing, a classification model was employed to determine the most significant variables contributing to shot success. First data was split into two groups representing, firstly; shots on goal and goals (472 data points), and secondly; missed and blocked shots (544 data points). Because the amount of data points between these two groups was just was just 14%, a class imbalance for training the classification does not have to be stressed.

Preparing the Model

To utilize all the information available, all data needs to be converted into numerical form. Python library panda's *get_dummies* is used to convert every individual textual feature, such as *team_name* into numerical form, where it is turned into a binary form for each selection of the variable. Next, the data is prepared for training a model. The data is split into training data and holdout data for testing to ensure confidence in deploying the model into production. Scikit-learn's *train_test_split* model was used to split the data into a test size of 33% for the total data. A random_state was used to ensure our results and experiment for each model we test is consistent and repeatable.

Training the Model

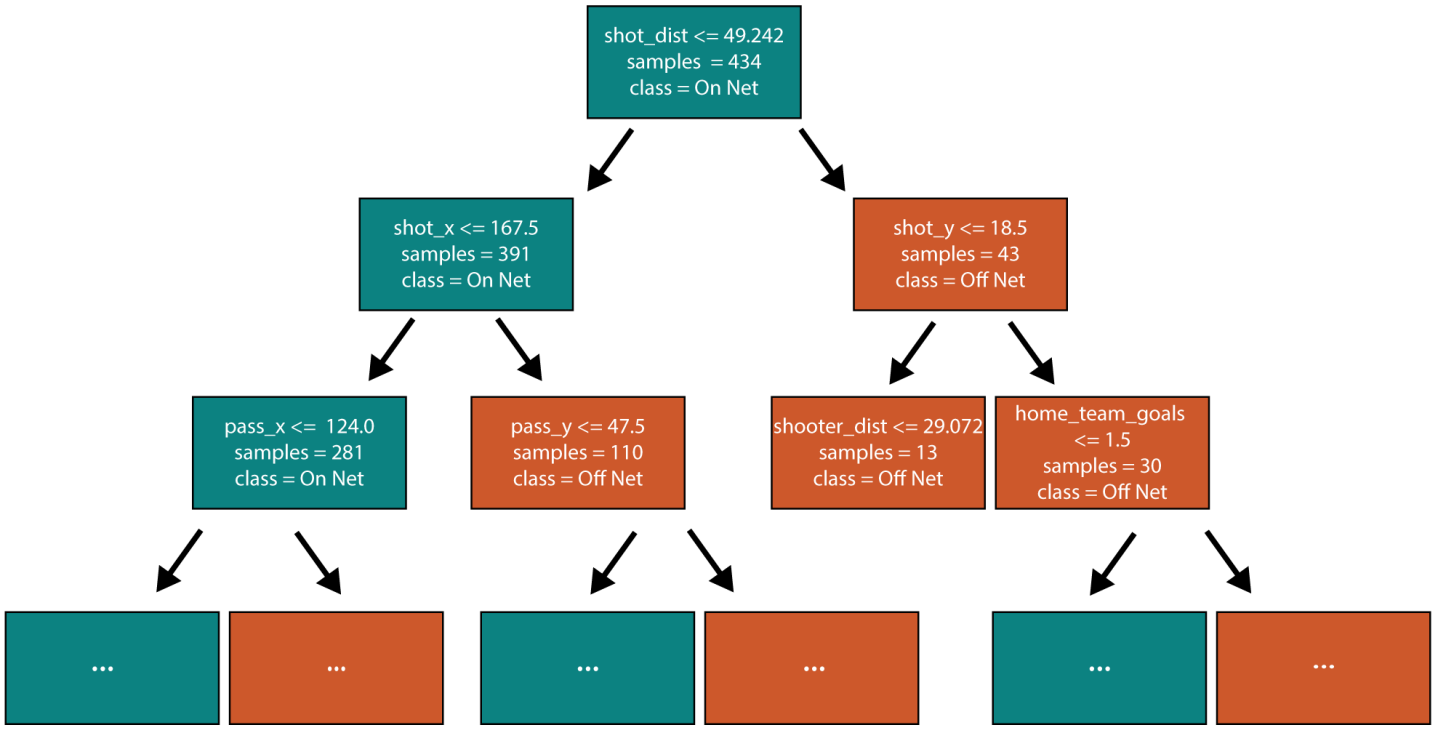
A random forest classifier (RFC) model was used as the model. A RFC is an effective first choice of a model to classify the data as no scaling is required between features, which is necessary for other classifiers such as logistic regression. The first pass of the RFC resulted in a precision score of 66.9% and a recall score of 51.5%. With the additional number of features to build a model after using *get_dummy*, 336 features exist. This caused the model to face the curse of dimensionality, where due to the high volume of features causing a reduction in model performance. To deal with this, a feature reduction mechanism was used.

Tuning the Model

Feature reduction was accomplished using recursive feature estimation (RFE). RFE works by searching for a subset of features by starting with all features in the training dataset and successfully removing features until the desired number remains. The RFC works by fitting the RFC used previously to rank features and keep the most important ones. RFE continues to discard the least important features, and finally provides a re-fit RFC model. This process is repeated until a specified number of features remains [Applied Predictive Modeling, 2013]. 25 features were selected to keep from the original 336 features for the RFC. This resulted in a boost of the precision to 67.9% and as well the recall to 56.5%, respectively a 1% gain for precision and 5% for recall, while reducing the amount of features used by 92.5%.

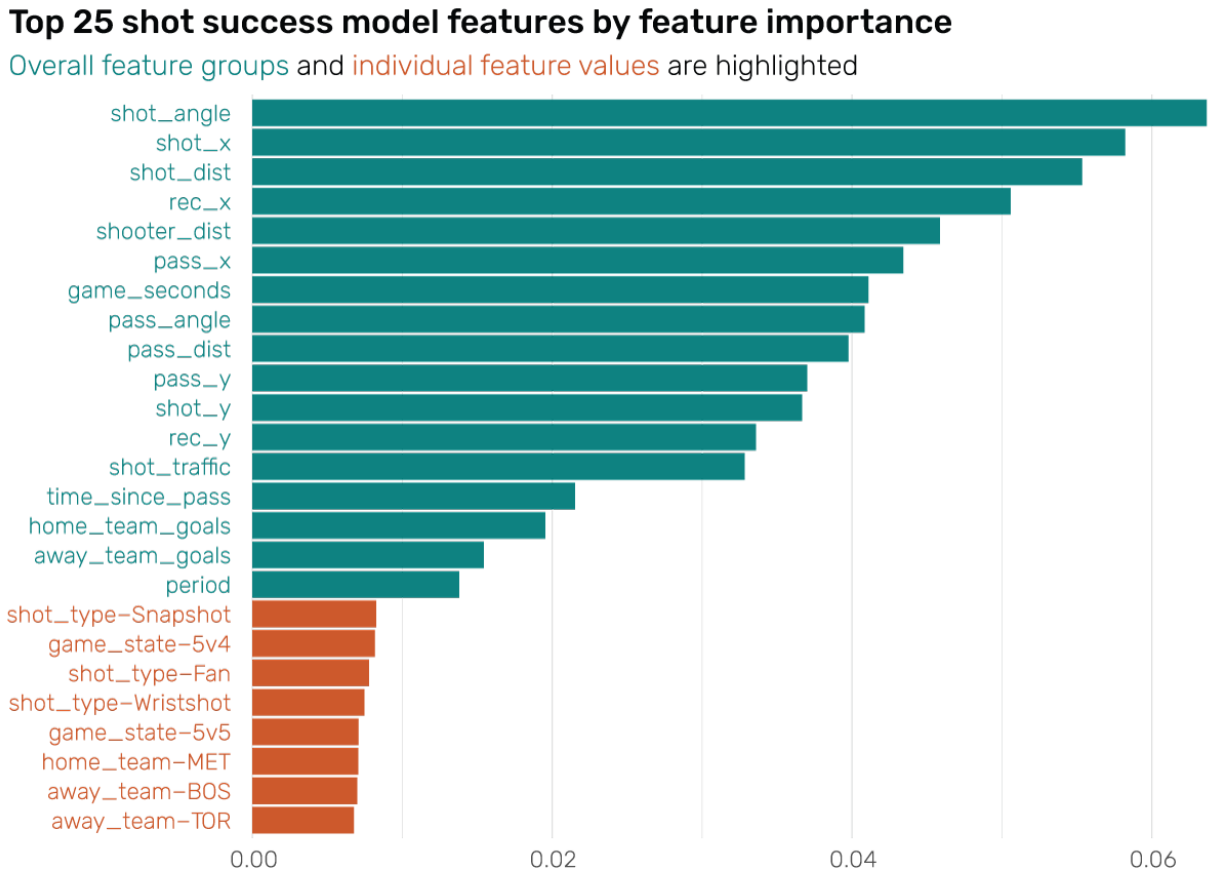
Example RFC Model Schematic

Below is a visual example of the random forest model output. The full output contains many more decisions through many more "branches", but the process entails looking at specific feature thresholds and classifying the next set of nodes based on shot outcome and next feature classification:



Outcome/Findings

The result of the final model is a ranked list of model features based on importance. This list provides insight into the most important factors contributing to shot outcome and can be used by coaches and managers to develop systems and strategies that maximize their team's opportunities for successful shots. The top features are detailed below:



It is important to note that the model outputs the most important features that affect shot outcome. This includes the features that contribute most to positive shot outcomes and negative shot outcomes.

Insights

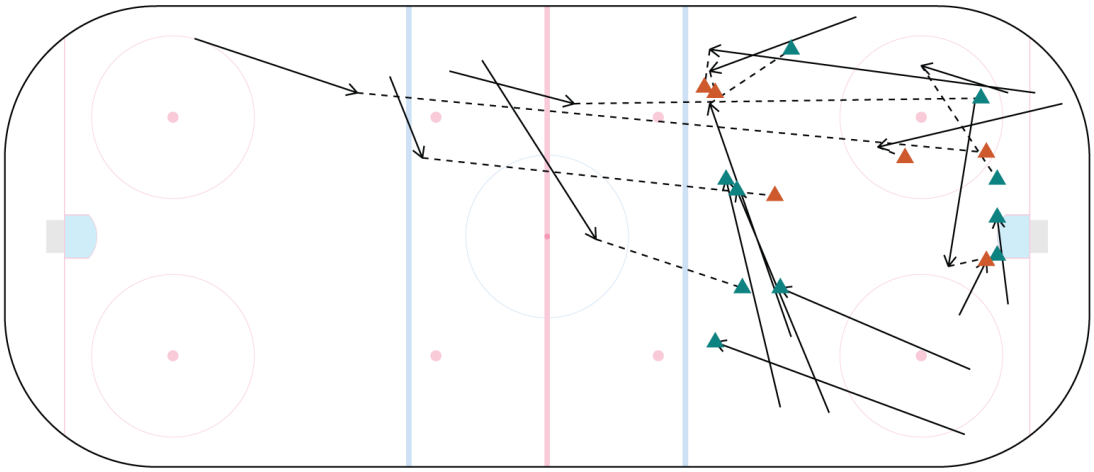
Several insights can be gained even from a cursory glance at the model output. The feature importance can be viewed as being a part of two groups. The first group of features are present in every pass-to-shot sequence. The second group identifies which feature values stand out the most. At this point of the work, the feature importance ranks are used to flag things to explore from a coaching perspective. From the first group, observations are generally intuitive and align with expectations:

- The most important features are related to shot characteristics, including angle, x position, and distance.
- X values are more important than y values, meaning the north-south positioning of players is much more important than the east-west positioning. This makes sense as the x value contributes most to distance from the net.

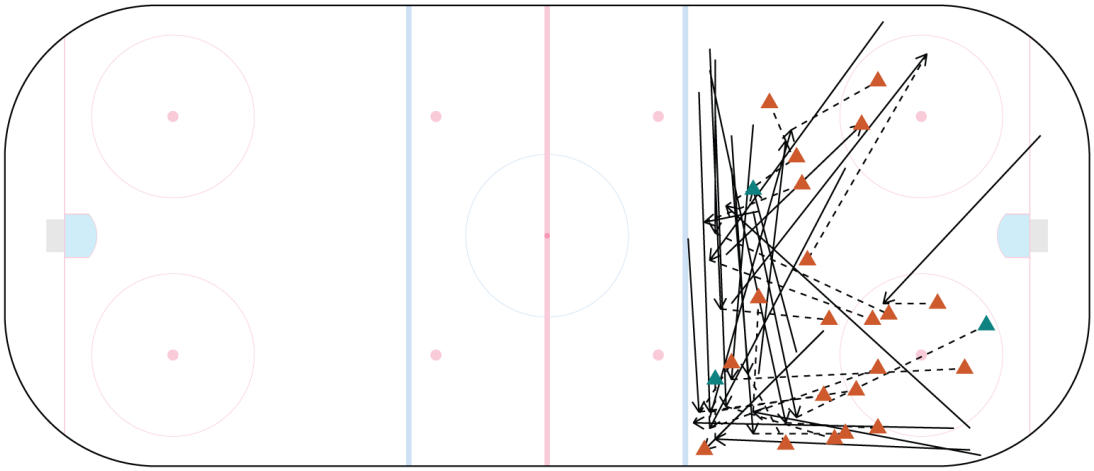
However, the strength of the model comes from gaining insights on the individual feature values that stand out. For example, the skater that is identified as having the highest impact as a passer is Brooke Boquist and as a shooter is Taylor Turnquist. The pass-to-shot sequences of these players can be plotted to fill in the information regarding the pass origin through to the shot outcome as seen in the following figure.

Arrows show pass distance and direction, dashed lines show displacement of skater before shot, and coloured dots show shot location and outcome

Direct passes from Brooke Boquist (Toronto Six) that led to successful or unsuccessful shot attempts



Successful and unsuccessful shots by Taylor Turnquist (Boston Pride) originating from direct passes



From these two individual feature highlights, we are shown two separate insights. First, Brooke Boquist is effective as a direct passer. When she is able to get her pass to her intended target, the shooters tend to take shots that make it on net. Conversely for Taylor Turnquist, she is ineffective as a shooter. She is a volume shooter when it comes to receiving direct passes, but her shots do not tend to make it to the net, regardless of whether she shoots early after pass reception or skates with the puck before shooting. The model identified these two players as worth exploring, and their individual impacts on shots being on net are immediately tangible, one for the better, one for the worse. Without the model, they might have been overlooked, or it would be tedious to go through every player and look for patterns.

The highest weighted passers and shooters are as follows:

Passers	
Brooke Boquist	0.003057
Alyson Matteau	0.002563
Emma Vlasic	0.002506
Taylor Turnquist	0.002459
Kate Leary	0.002443
Mikyla Grant-Mentis	0.002438
Breanne Wilson-Bennett	0.002331
Haley Mack	0.002291
Tereza Vanisova	0.002216
Samantha Davis	0.002207

Shooters	
Taylor Turnquist	0.003671
Lindsay Eastwood	0.002671
McKenna Brand	0.002559
Sydney Baldwin	0.002453
Emma Woods	0.002393
Kristin Lewicki	0.002320
Madison Packer	0.002191
Sarah-Eve Coutu Godbout	0.002130
Mikyla Grant-Mentis	0.002129
Kiira Dosdall	0.002080

Knowing the impact that each player has on the model in terms of feature importance immediately provides an exploration path. The next steps would be to further investigate each of the passers and shooters to determine whether they fall into the positive or negative outcome categories and develop strategy and play adjustments accordingly.

Future Work

The purpose of this project was to determine a robust and reproducible method to analyze NHL play-by-play data that includes the next generation of stats containing more contextual and position information. The framework developed is applicable beyond the NWHL data and could be used in any league. This analysis is a steppingstone to more comprehensive future analyses, including those outlined below.

Implementation of all skater positional data

Including positional data of all skaters and goalies for tracked events opens the ability to incorporate those variables directly into the model for events of interest. This could give more insight into qualifying passes themselves by looking at where the pass was delivered to and received relative to the players involved, qualify pass danger based on the position of other skaters on the ice at that time, and identify defending players who are effective at disrupting passing plays.

Evaluation of individual players and adapting hockey systems

As the model ranks players based on their events and their impacts on shot outcomes, looking at the plots created for each player and event pairing gives immediate insight on what makes them effective or ineffective as passers or shooters. This could help coaches encourage players to focus on specific areas of their game for improvement, or teams to focus on oppositional strengths and exploit oppositional weaknesses in a much more individual level.

Analysis of attacking and defending systems

By looking at how teams break out of their defensive zone and cycle the puck in the offensive zone, a better understanding of in-game tactics can be gained. This allows for expected goals or win probability models to account for competing attacking and defending strategies, help identify coaching effectiveness by tracking adjustments and adaptations to established systems and identify how players respond to opponent systems on the fly.

Evaluation of Goalie Positioning

Properly evaluating goaltenders is difficult to do with the current set of data available. By adding goalie data, variables such as goalie positioning, goalie face angles, and goalie movement can be analyzed and weighted based on their effect on shot outcomes. This would further enhance expected goals models and provide a more effective way to evaluate goaltending ability.

Appendix A: Processed Data

The processed data is included in the submission as "bdc_nwhl_direct_pass_data.csv". This includes all the data accompanying the model creation.

Appendix B: Data Processing R Code

```
library(here)
library(tidyverse)
library(lubridate)
library(janitor)

data_nwhl <- read_csv("https://raw.githubusercontent.com/bigdatacup/Big-Data-Cup-2021/main/hackathon_nwhl.csv") %>%
  clean_names() %>%
  filter(event %in% c("Shot", "Play", "Goal")) %>%
  mutate(game_seconds = case_when(
    period == 1 ~ 1200 - as.numeric(ms(str_sub(clock, 1, 5))),
    period == 2 ~ 2400 - as.numeric(ms(str_sub(clock, 1, 5))),
    period == 3 ~ 3600 - as.numeric(ms(str_sub(clock, 1, 5))),
    period == 4 ~ 3900 - as.numeric(ms(str_sub(clock, 1, 5)))
  )) %>%
  relocate(game_seconds, .after = clock) %>%
  mutate(
    shot_from_pass = if_else(player == lag(player_2, n = 1L) & event %in% c("Shot", "Goal") & lag(event, n = 1L) == "Play", TRUE, FALSE),
    pass_to_shot = if_else(player_2 == lead(player, n = 1L) & event == "Play" & lead(event, n = 1L) %in% c("Shot", "Goal"), TRUE, FALSE)
  ) %>%
  filter(pass_to_shot | shot_from_pass == TRUE) %>%
  mutate(time_btw_events = ifelse(shot_from_pass == TRUE, game_seconds - lag(game_seconds, n = 1L), NA)) %>%
  relocate(time_btw_events, .after = game_seconds)

data <- data_nwhl %>%
  mutate(
    game_state = case_when(
      team == home_team ~ str_c(home_team_skaters, "v", away_team_skaters),
      team == away_team ~ str_c(away_team_skaters, "v", home_team_skaters)
    ),
    passer = if_else(lag(event, n = 1L) == "Play", lag(player, n = 1L), NA_character_),
    pass_type = if_else(lag(detail_1, n = 1L) %in% c("Direct", "Indirect"), lag(detail_1, n = 1L), NA_character_),
    pass_x = if_else(lag(event, n = 1L) == "Play", lag(x_coordinate, n = 1L), NA_real_),
    pass_y = if_else(lag(event, n = 1L) == "Play", lag(85 - y_coordinate, n = 1L), NA_real_),
    rec_x = if_else(lag(event, n = 1L) == "Play", lag(x_coordinate_2, n = 1L), NA_real_),
    rec_y = if_else(lag(event, n = 1L) == "Play", lag(85 - y_coordinate_2, n = 1L), NA_real_),
    shooter = if_else(event %in% c("Shot", "Goal"), player, NA_character_),
    shot_x = if_else(event %in% c("Shot", "Goal"), x_coordinate, NA_real_),
    shot_y = if_else(event %in% c("Shot", "Goal"), 85 - y_coordinate, NA_real_),
    shot_type = if_else(event %in% c("Shot", "Goal"), detail_1, NA_character_),
    shot_outcome = if_else(event %in% c("Shot", "Goal"), detail_2, NA_character_),
    shot_outcome = if_else(event == "Goal", "Goal", shot_outcome),
    shot_traffic = if_else(event %in% c("Shot", "Goal"), detail_3, NA),
```

```
    shot_onetimer = if_else(event %in% c("Shot", "Goal"), detail_4, NA),
    pass_dist = sqrt((rec_x - pass_x)^2 + (rec_y - pass_y)^2),
    pass_angle = atan2(43 - pass_y, 189 - pass_x) * 180 / pi,
    shooter_dist = sqrt((shot_x - rec_x)^2 + (shot_y - rec_y)^2),
    shot_dist = sqrt((189 - shot_x)^2 + (43 - shot_y)^2),
    shot_angle = atan2(43 - shot_y, 189 - shot_x) * 180 / pi
  ) %>%
  filter(shot_from_pass == TRUE,
         pass_type == "Direct") %>%
  select(-c(
    clock, home_team_skaters, away_team_skaters, event, player,
    pass_type, detail_1, detail_2, detail_3, detail_4, player_2,
    x_coordinate, y_coordinate, x_coordinate_2, y_coordinate_2,
    shot_from_pass, pass_to_shot
  )) %>%
  rename(time_since_pass = time_btw_events) %>%
  relocate(shooter, .after = passer)
```

Appendix C: Machine Learning Python Code

```
### imports
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_selection i RFE
from sklearn.metrics import confusion_matrix
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score

df = pd.read_csv('data_2020-02-24.csv')

y = df['shot_outcome'].str.contains('Goal|On Net')
X = df.drop(columns=['shot_outcome'])
dropped_columns = ['game_date']
X = X.drop(columns=dropped_columns)

# utilize pandas get dummies to turn textual features into numerical binary features
dummies_columns = ['team','passer','shooter', 'shot_type', 'game_state', 'home_team',
'away_team']
X_dummies = pd.get_dummies(X, prefix_sep='-', columns = dummies_columns)

# split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_dummies, y, test_size=0.33,
random_state=13)
rfc = RandomForestClassifier(random_state=12)
rfc.fit(X_train, y_train)

# perform recursive feature estimation to determine the most important features
rfe = RFE(estimator=rfc, n_features_to_select=25, step=1)
rfe.fit(X_train, y_train)

df_feature_importance = pd.DataFrame(rfc.feature_importances_, index = X_test.columns )
df_feature_importance.to_csv('df_feature_importances.csv')
```