# NHL Player Point Prediction

Matthew Jung 301261171, Alex Hua 301261352
CMPT 353- 3 Summer 2019
Simon Fraser University

**Abstract:** Predicting how your favourite NHL player will perform in the future is a difficult task, given the nontrivial means necessary to make such a prediction. Here, taking advantage of methods in statistics, machine learning, and computer science we sought to determine if we could predict whether a player gets a point in a game. Using correlative statistics, we identified that a player's time on ice, the number of shots per game, the power play time on ice, and a player's points per game, were the most influential features in making such a prediction. Using the support vector classifier, we then were able to correctly predict with 67.6% accuracy whether a player gets a point.

## 1. Introduction

An intriguing aspect of sports – and hockey in particular - is whether an outcome can be predicted. Until recently, predictions relied more on qualitative aspects. Such qualitative aspects can be subject to bias and human error. However, with the advent of readily available National Hockey League (NHL) data and the rise in popularity of machine learning and statistics, quantitative techniques are becoming commonplace for the hockey enthusiasts looking to track their favourite teams and players. As such, given a dataset of past NHL individual player performances, is it possible to predict how an NHL player will perform in the future? Namely, given this dataset, is it possible to predict if a player will get a point in some other game? Here, we use techniques in statistics, machine learning, and computer science to identify meaningful patterns within the dataset to make such predictions. Namely, we identify correlated features to player points then, using these features, train a machine learning model. Predicting whether a player (forward) gets a point or not can be reduced to a binary classification problem. Thus, we decided to use the support vector classifier model.

## 2. Materials and Methods

The datasets were obtained from a user on Kaggle (1). Three of the datasets were joined together to create a cumulative dataset that included every NHL skater who has been on an NHL roster since the 2010-2011 season (to 2018-2019). Here, skater is defined as any player that is not a goalie. This cumulative dataset, after the join, also included every game each player had played in that span. A preliminary filter was used to drop players from the dataset who had not played any games/had no game data. This cumulative dataset had 411579 entrees and over 2000 unique skaters. The dataset was further cleaned by eliminating all defenseman, leaving only forwards. Finally, forwards' data were ordered chronologically, from the least recent game played to the most recent.

Preliminary analysis of the dataset was performed by creating histograms for player variables. Histograms were used to identify any potential outliers, within the data (Appendix III). Then, scatterplots and heatmaps of points vs player variables were created as a visualization tool for the data. Features for the SVC model were chosen if they were correlated with points (Table

1, Appendix I-II). Using the data in the dataset, new variables were created. These variables were an attempt to create more meaningful relationships between the data. Scatterplot, heatmaps, and correlation analyses (player variables vs player points) were performed to identify which player variables could be used as features in our machine learning model.

Predicting whether a player gets a point in some game is a binary classification problem. Concretely, we either classify whether a player gets a point (1) or does not get a point (0). Thus, we used support vector classifier (SVC) to make our predictions, using Python's sklearn.svm SVC library. The dataset was split into 75% used for training and 25% used for testing. Before training our model, we used MinMaxScaler() to make all features have the same scale.

GridSearchCV() is used to determine what parameters were optimal. A range of 1-10 for the C error term value was tested five times for each C value, as to find the optimal SVC parameters.

## 3. Results

### *3.1 Feature Engineering and Choosing Features*

More meaningful variables were created to represent the more complex relationships between the data. The features chosen for our final model were time on ice (timeOnIce) ($r^2$ = 0.286), shots ($r^2$ = 0.318), power play time on ice (powerPlayTimeOnIce) ($r^2$ = 0.252), and points per game (ppg) ($r^2$=0.273) (Table 1). The features that were were rejected from the final model were Pts y/n l_7 ($r^2$ = 0.217), Pts y/n l_3 ($r^2$ =0.173), games l_21 ($r^2$=0.0579), and games l_7 ($r^2$=0.0393) (Appendix I-II).

**Table 1:** Feature Correlation Analysis

| Feature 1 | Feature 2 | Correlation $r^2$ Forwards |
|---|---|---|
| timeOnIce | points | 0.286 |
| Shots | points | 0.318 |
| powerPlayTimeOnIce | points | 0.252 |
| ppg | points | 0.273 |

### *3.2 Support Vector Classifier*

A support vector classifier model was created to make our prediction. The model included feature scaling (MinMaxScaler). GridSeachCV() determined that the optimal parameter for SVC is C = 4 with a linear kernel. The model achieved a training score of 0.676 and a testing score of 0.676 (Figure 1). More specifically, the model predicted a total 68314 player games which resulted in 41047 correct predictions for games in which the players did not have a point, 5019 correct games in which the players had a point,19085 false negatives, and 3163 false positives (Figure 2).

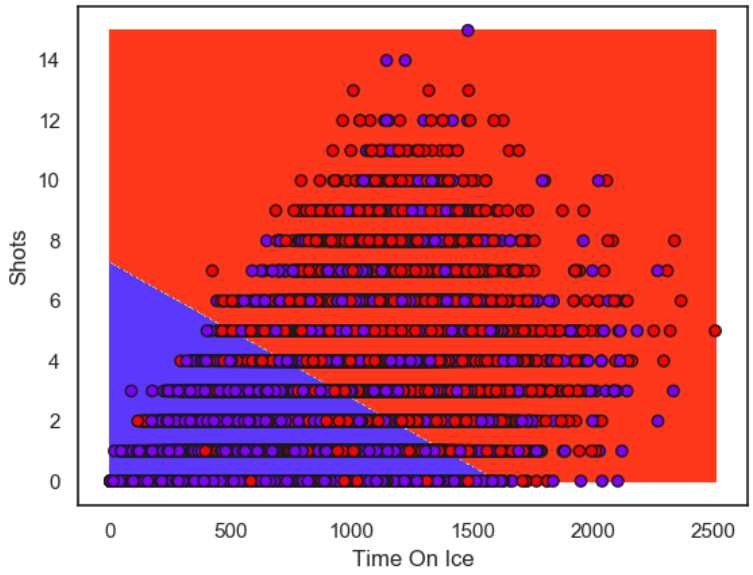



**Figure 1:** Support Vector Classifier. Purple dots indicate games for players that have a point and red dots indicate player games that do not have a point. The purple and red background indicates the decision boundary created by the model. The model's parameters were C = 4, kernel = 'linear'. Training score = 0.676. Testing score = 0.676.

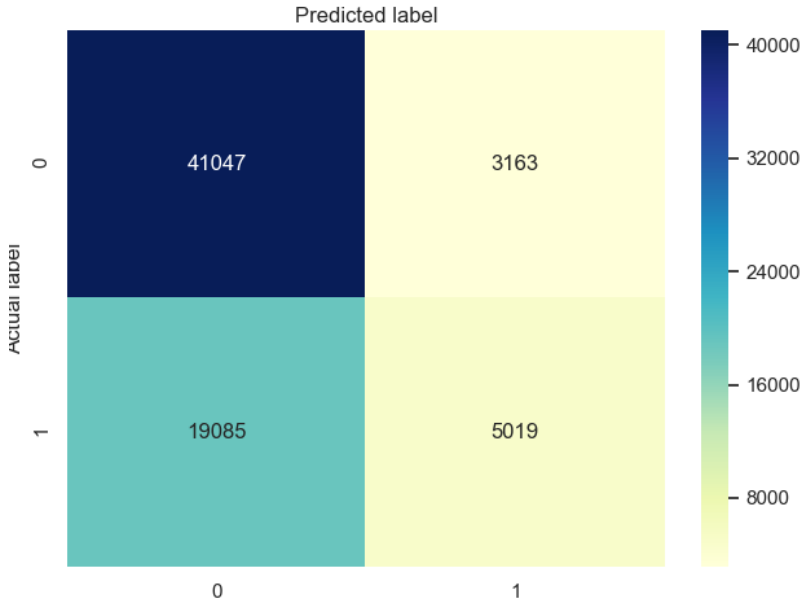



**Figure 2:** SVC Confusion matrix. Total predictions = 68314. Correct predictions: category 0 (no points) = 41047 Top left, category 1 (points) = 5019 bottom right. Incorrect predictions: false negatives = 19085 bottom left, false positives = 3163 top right.

## 4. Discussion

### *4.1 Feature Engineering and Choosing Features*

For our model's features, we analyzed time on ice (timeOnIce), shots, power play time on ice (powerPlayTimeOnIce), points per game (ppg), the number of games from the last seven games in which a player received at least a point (pts y/n l_7) the number of games from the last three games in which a player received at least a point (pts y/n l_3), number of games in the last twenty one days (games l_21), and number of games in the last seven days (games l_7) (Table 1, Appendix I-III). Of the features that we manually engineered, only ppg had improved our model (score = 0.658 without vs score = 0.676 with). Although pts y/n l_7 and pts y/n l_3 appeared to have a positively correlated association with points, the model could not create a decision boundary when they were both included – so they were removed. Because of games l_21 and game l_7's very weak positive correlation with points, we opted not to use them. It was hypothesized that perhaps a better prediction could be made if the model could capture more subtle aspects of player performance: phenomenon such as momentum/point streaks, consistency, and fatigue. Specifically, the feature pts y/n l_x (x = number of games) was meant to track if a player was on a point streak. The more games a player had with points, the more likely he was on a hot streak, and therefore they could ride their momentum into the next game. Ppg was a feature meant to track a player's consistency. The more points a player had per game could indicate how likely the player is to get a point in the next game – a player with a higher ppg is more likely than a player with a lower ppg. Finally, games l_d (games in the last d days) was meant to represent how fatigued a player could be. It was hypothesized that a player who plays more games in the last d days would be more fatigued than another player who has played less games over those same d days. Thus, this could result in a poorer player performance for the game being predicted.

Here lies a limitation to our model: we assumed that all features had a linear relationship with points. However, in the absence of a linear relationship (~ 0 correlation) – such as games l_d – there could be a more sophisticated relationship between fatigue and points: one that is not linear.

### *4.2 Support Vector Classifier Prediction*

As mentioned, the model has a training score of 0.676 and a test score of 0.676. The support vector classifier was used, given that in Figure 1 there appears to be an almost linear decision boundary between games that had a player get a point versus those that did not. Thus, we decided to use SVC with the linear kernel. In Figure 1, it is a nontrivial task to discern if the categories 0 or 1 (games without a point vs games with a point) are in fact different groups. From Figure 1, it does not appear that they are distinctly different, given the overlap. However, it would be remiss not to draw attention to the fact that at the extremes of both the purple group (games with a point) and red group (games without a point) (bottom left quarter vs top right corner, respectively) there is unequivocally more purple dots in the bottom left, and there unequivocally more red dots in the upper right corner. Also, from Figure 1 and 2 it certainly

appears that the model more accurately predicts whether a player does not get a point. A possible explanation for this is that if a player has a lower time on ice, lower shots per game, and lower power play time on ice he will certainly have less opportunity to score. However, the opposite may strictly not be true. If a player has more time on ice, more shots per game, and more power play time on ice, this does not necessarily guarantee that he will get a point or that he is improving his opportunities of scoring. This can be highlighted by the more subtle aspects when analyzing these features. For instance: how much time on ice is the player playing in the offensive zone? Where is the player taking his shots from? Or how good is a team's power play? Fundamentally, this is a quality versus quantity argument per se. Thus, while we have naively assumed that points linearly correlate with our chosen features, this may not be entirely true. Looking at the scatterplot for timeOnIce, for example, we can see that there is not a strong correlation between timeOnIce and points (Appendix II). As timeOnIce increases, points do not necessarily follow.

Another limitation to our prediction is that we use every forwards' data to create one model, as opposed to creating a separate model for each player. In theory, a model for each player would be more finely tuned for the changes in values for each feature and could create a model that is player-specific. For instance, take two individual models for player X and player Y: call them model X and model Y. Now suppose that for player X, changes in time on ice correlates more positively to points compared to player Y. Simply put, if player X and player Y play the same amount of time, player X is more likely to get a point as opposed to player Y. However, our model assumes that all correlations between features and points is uniform. Therefore, if we strictly had testing/holdout data for a specific player, and we used our model to predict whether a specific player was going to get a point in any number of the testing/holdout games we may get more or less than 0.676 (our model's testing score).

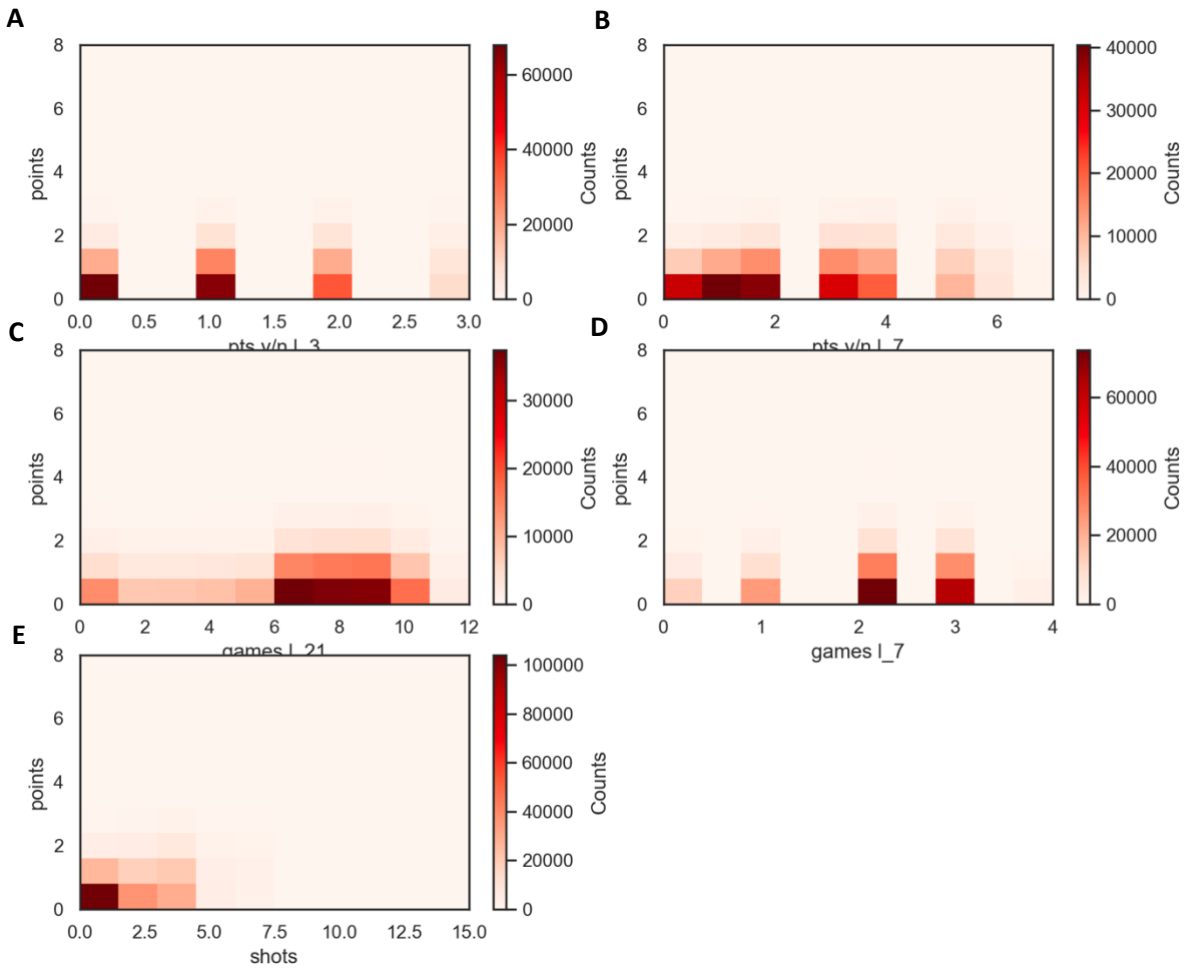
As an aside, another significant limitation of our prediction is that we predict whether a player gets a point based on data for features that already exist. Specifically: if we truly wanted a model that could make predictions, we would need to use values for features that did not already exist. Likely, we would need much more advanced techniques such as forecasting and time series analysis to make our predictions.

The prediction of whether a player gets a point in a certain game is tantalizing for any hockey enthusiast. If a model can accurately and reliably make such a prediction, such information can give an advantage to any analyst. Yet, it should be clear that these findings are not without their limitations. These findings should be taken at face value and it would be optimistic, even naïve, to expect accurate and precise predictions without drastic improvements and refinements to the model.

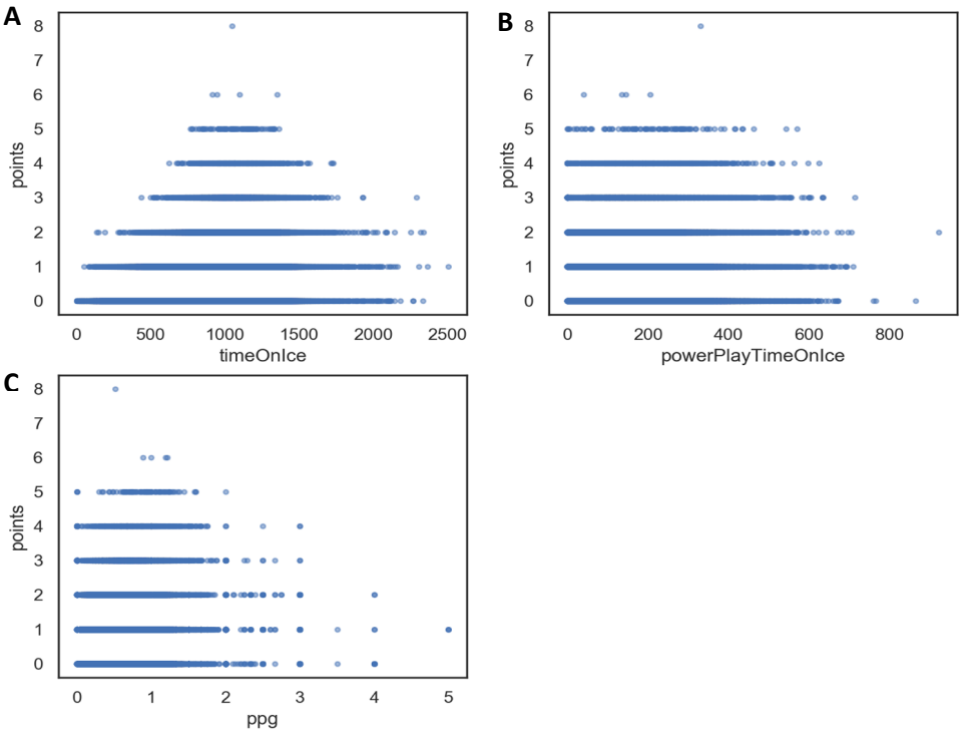
## References

1. Kaggle. (2018) NHL Game Data Game, team, player and plays information including x,y coordinates. Retrieved from https://www.kaggle.com/martinellis/nhl-game-data

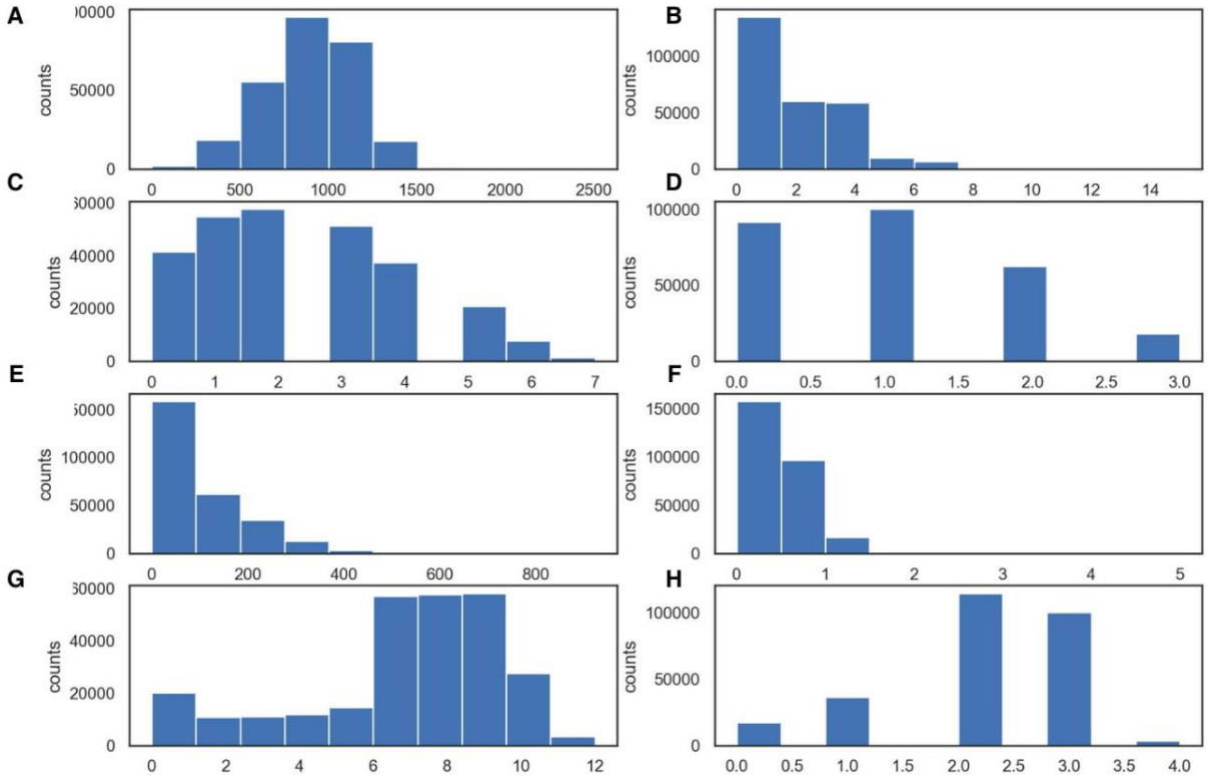
## Appendix




**Appendix I:** Heatmap of forwards' features. **A)** pts y/n l_3 vs points correlation = 0.173. **B)** pts y/n l_7 vs points correlation = 0.217. **C)** games l_21 vs points correlation = 0.0579. **D)** games l_7 vs points correlation = 0.0393. **E)** shots vs points correlation = 0.312.




**Appendix II:** scatterplots of forward's features. **A)** timeOnIce vs points correlation = 0.286. **B)** ppg vs points correlation = 0.272. **C)** powerPlayTimeOnIce vs points correlation = 0.252.




**Appendix III:** histogram of forwards. **A)** timeOnIce. **B)** shots. **C)** pts y/n l_7. **D)** pts y/n l_3. **E)** powerPlayTimeOnIce. **F)** ppg. **G)** games l_21. **H)** games l_7

# Project Experience

Matthew:

- Implemented an efficient python script that joins several .csv files together using pandas library. Inputted .csv files contained over four hundred thousand rows, thus needed to chunk data when joining to prevent memory overload. Resulting dataset was used for our machine learning model.

- Engineered machine learning features by drawing meaningful relationships between data within dataset. Features were engineered by creating python methods that, at times, relied upon python's datetime and pandas library. Engineered features helped increase accuracy of machine learning model by 2%.

- Analyzed dataset by creating scatterplots, histograms, and performing correlative statistics to identify candidate features for our model, using python's matplotlib and scipy libraries. Chosen features resulted in the ML model being able to predict 67.6% accuracy if a player gets a point or not.

- Wrote a report explaining the results from our investigation. Report included formal presentation of problem statement, methods, results, and a discussion of the results and limitations of findings.

Alex:

- Created and optimized machine learning models, SVC() and K-nearest via GridSearchCV() to achieve the best score possible for the machine learning model.
- Visualized the results obtain from the data analyses and machine learning outputs, specifically created histograms, heatmaps, scatterplots for better interpretation of the results.
- Wrote a report explaining the results from our investigation. Report included formal presentation of problem statement, methods, results, and a discussion of the results and limitations of findings.
- Was in charge of maintaining the GitLab repository, such as approving merge requests so that no conflicts arised