An Analysis of Team Success in relation to Player Qualities in the National Hockey League

edX HarvardX: PH125.9x - Data Science: Capstone

Craig Hamlin

August 5, 2021

Contents

1 INTRODUCTION

In this project a publicly available dataset was employed to create machine learning models which predicted the rank of a National Hockey Leagure players team.

1.0.1 National Hockey League Overview

The sport of ice hockey in the National Hockey League (NHL) consists of 32 teams of professional athletes that compete against each other within a points based match system. The results of each match produces a winner and a loser and the objective of every team is to win as many games as possible. A season of hockey consists of roughly a one year time period whereby the points system resets and the teams once again start a new season. The teams that obtain the highest winning percentages in each season can be considered to be in the elite, top stratum of the NHL. Imposed upon every team equally is a salary cap which is the total amount of money that a team can pay its players for a season. The salary cap has a dual purpose: to create parity in the league so rich teams cannot financially dominate over smaller market teams and to prevent player salaries from being inflated beyond league sustaining levels. Players that make up the team can play one of 5 different positions: Center, Left Wing, Right Wing, Defence and Goalie. For the sake of this project the position of Goalie was not included in the dataset as the performance statistics are not comparable with the other positions. Using 10 separate features unique to every player observation in our dataset, team rank was predicted through the machine learning models of Recursive Partitioning, Robust Linear Discriminate Analysis and eXtreme Gradient Boosting. The analysis of the data and results of the machine learning models are contained within this report.

2 Methods and Analysis

2.1 Data Wrangling and Cleaning

After importing the nhl player and team data with permission from hockeystatssupplier on github, the data was then cleaned and engineered to best suit the project. Of note: only players with a minimum of 20 games played per season were included in the player datasets. This minimum games threshold was important to implement in order to provide a larger sample size for player performance on a seasonal basis.

2.1.1 Download Player Data and coerce into useable dataframe

Player data consists of two separate dataframes. Each contain similar columns such as player names, position, etc, however one dataframe contains salary cap based information such as cost per point, and the other dataframe contains other in-depth game based information like ice time per game and power play points totals. The two separate dataframes were modified with regex to create a more workable resource by removing dollar signs, trimming white space and removing accents from letters. Also, the features cap hit and cost per point were weighted to account for the salary cap inflation that occurs yearly. The two data frames were then joined to be utilized later alongside the team data.

Player	AGE	CAP.HIT	X..P	GP	TEAM	Pos	Season	G	A	P	PPP	TOI.GP
Jamie Benn	22	798523	12675	71	DAL	L	20112012	26	37	63	10	1084
John Tavares	21	1140747	14083	82	NYI	C	20112012	31	50	81	25	1234
Frans Nielsen	27	665435	14158	82	NYI	C	20112012	17	30	47	15	1047
Erik Karlsson	21	1109059	14218	81	OTT	D	20112012	19	59	78	28	1519
Adam Henrique	21	728810	14290	74	NJD	C	20112012	16	35	51	8	1090
Jordan Eberle	21	1109059	14593	78	EDM	R	20112012	34	42	76	20	1056

2.1.2 Download Team Data and coerce into useable dataframe

Team data consists of team performance data from years 2011-2020. The accumulated points of each team by year is displayed in a column within the dataframe. From this data a three tiered ranking system was developed which split teams into three groupings: 1 (top third percentile), 2 (middle third percentile), and 3 (bottom third percentile). Each team in each season now has a performance ranking attribute applied which has reflective of its total points obtained in that season.

	PTS	Team	rank
26	80	Anaheim	1
8	102	Boston	3
20	89	Buffalo	2
18	90	Calgary	2
24	82	Carolina	1
11	101	Chicago	3

2.1.3 Combine the Player and Team Data

The final dataframe utilized in the models was created by combining the Player and Team dataframes. Several steps were needed to fit the two groupings together: city names in the team data needed to be converted to abbreviations to match the player data. The number of teams and team city names aren't exactly the same for all seasons so this needed to be addressed in the coding. The final step was to bind the two dataframes, convert variables to numeric where necessary and edit out any unwanted columns that would not be used as features in the model.

Player	AGE	CAP.HIT	X..P	GP	TEAM	Pos	Season	G	A	P	PPP	TOI.GP	rank
Jamie Benn	22	798523	12675	71	DAL	L	20112012	26	37	63	10	1084	elite
John Tavares	21	1140747	14083	82	NYI	C	20112012	31	50	81	25	1234	elite
Frans Nielsen	27	665435	14158	82	NYI	C	20112012	17	30	47	15	1047	elite
Erik Karlsson	21	1109059	14218	81	OTT	D	20112012	19	59	78	28	1519	elite
Adam Henrique	21	728810	14290	74	NJD	C	20112012	16	35	51	8	1090	elite
David Desharnais	25	1077372	17955	81	MTL	C	20112012	16	44	60	20	1104	elite

Each observation in the dataset consists of one player's yearly (seasonal) results in 11 categories. The variables that were selected to remain in the final dataframe were:

- Response variable:
  - Rank: team rank of player (Elite, Bubble, Weak)
- Features:
  - AGE: age of player
  - CAP.HIT: adjusted salary of player
  - X..P: points divided by adjusted salary of player
  - GP: Games played in season by player
  - Pos: Position of Player (D,L,R,C)
  - P: number of points scored by player
  - TOI.GP: Average playing (ice) time per game by player
  - PPG: Power Play Goals scored in total by player
  - PPPG: Power Play Points per game by player
  - PPP: Power Play Points scored in total by player

2.1.4 Create training and test sets

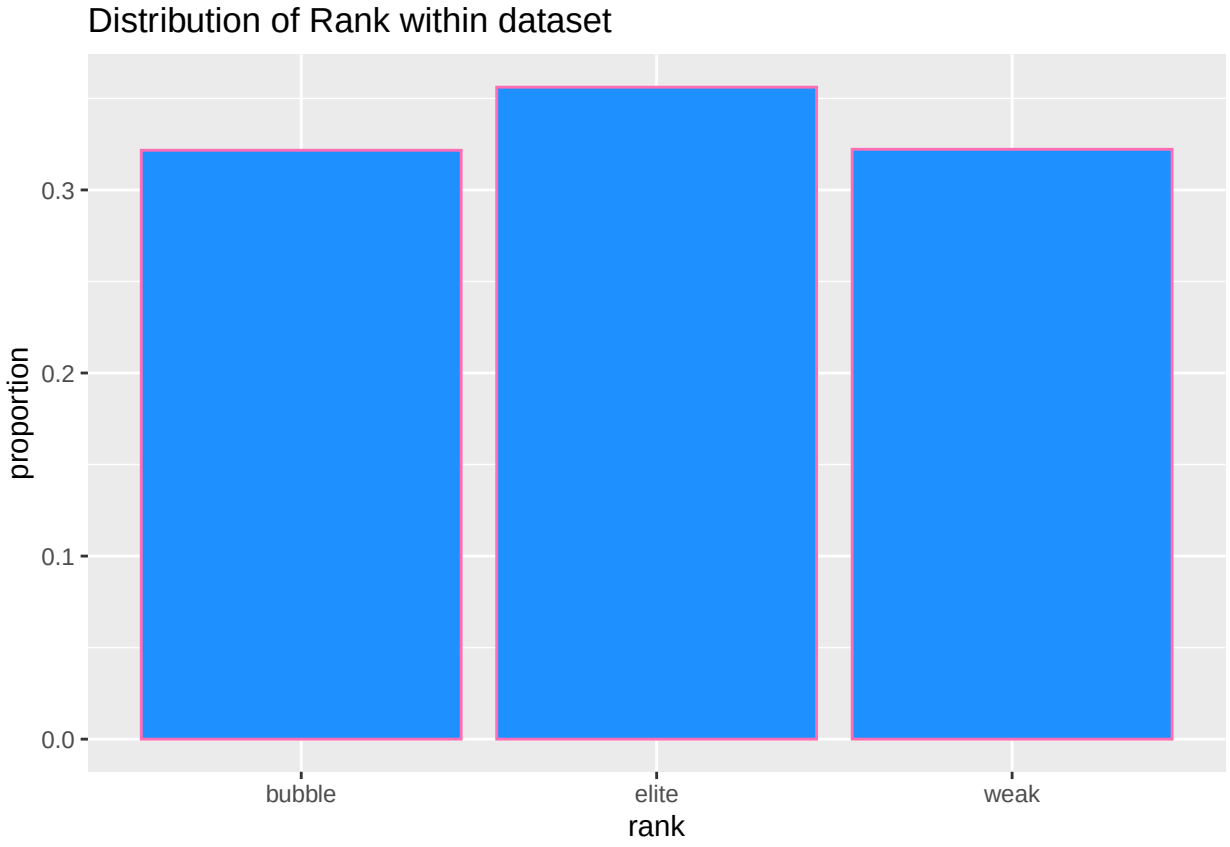
The player and team data obtained for this project covers the years 2011 until 2020. For modeling purposes the decision was made to designate the data from years 2011-2019 as the training set and the data from year 2020 as the test set. The training set consists of 5571 player values and the test set consists of 597 player values.

2.2 Exploratory Data Analysis

Exploratory Data Analysis is the initial process of analyzing a dataset to try and gain insight and obtain a summary of the relevant characteristics. In this project the usage of graphs and summary statistics were conducted in order to gain maximum insight into both the player and team data.

2.2.1 Distribution of Rank within Dataset

The data is separated into three rankings (Elite, Bubble, and Weak) which are based off of a 10 year sample of all players in the league and the team results for these seasons.

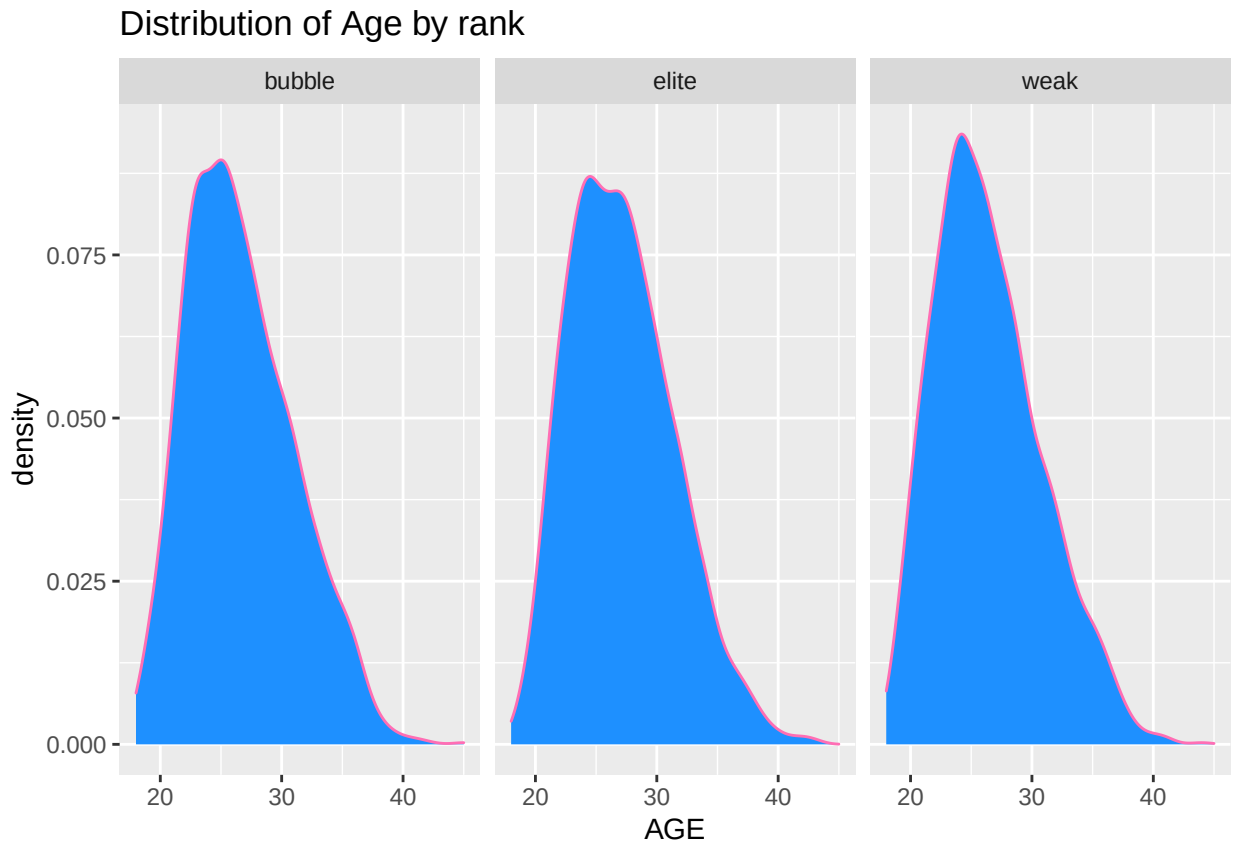


The bargraph above shows that the highest proportion of the training data is elite and the lowest proportion is weak. We have divided our team standings equally by percentile to create a non biased representation of team so why are we seeing a higher proportion of players that play for elite rather than weak teams? The answer would be found in the selection criteria for the dataset, in particular the minimum games played selector in the player data. As our cut off level for games played is 20 this would indicate that weak teams have more players with under 20 games played per season than elite or bubble teams. From this interpretation we can deduce that elite teams keep a roster of players that remains fairly constant through an entire ~80

game season. Weak teams, on the other hand, tend to have a roster that isn't firmly set and could feature young prospects and fill in players where the expectation level and number of games played is lower on average.

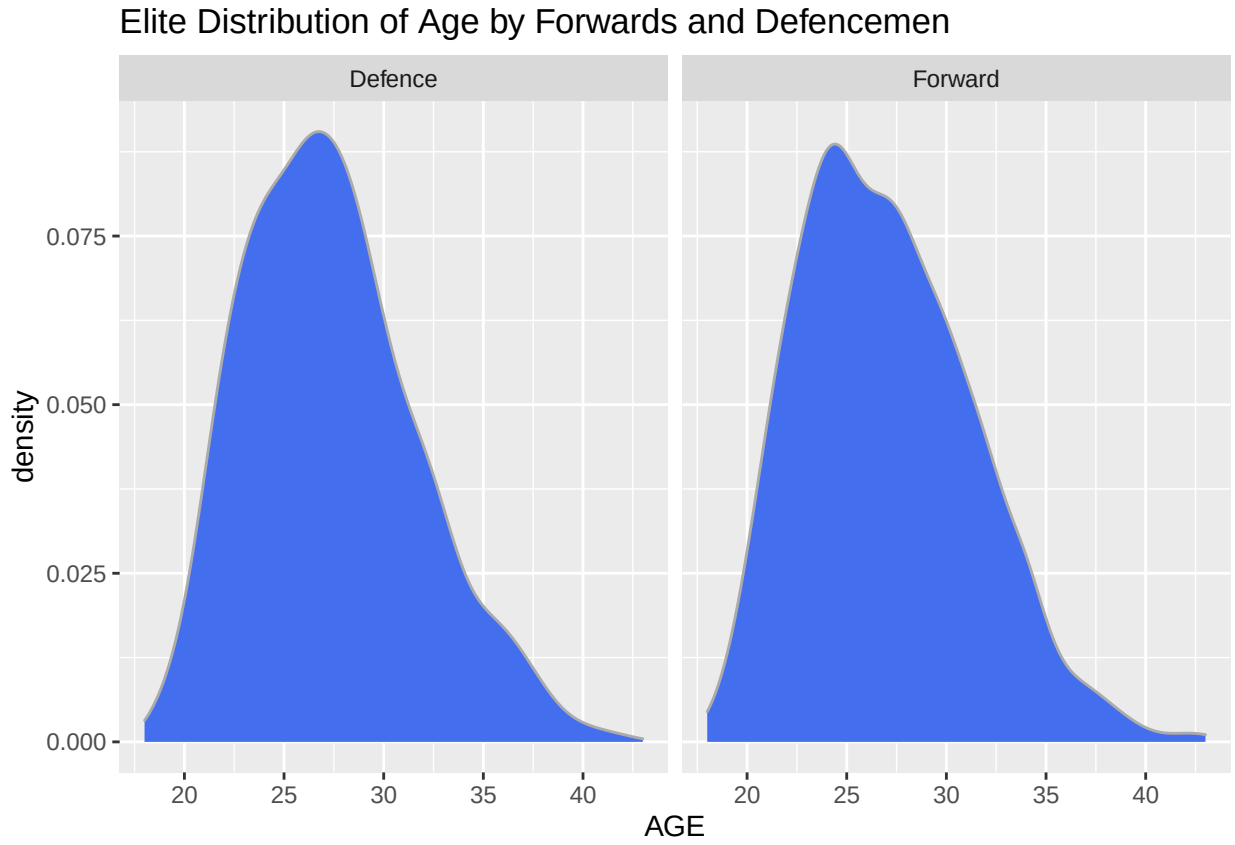
2.2.2 Distribution of Player Age by Team Rank

In order to search for noticeable difference between ranks we will select the player age feature and visualize the three separate distributions.



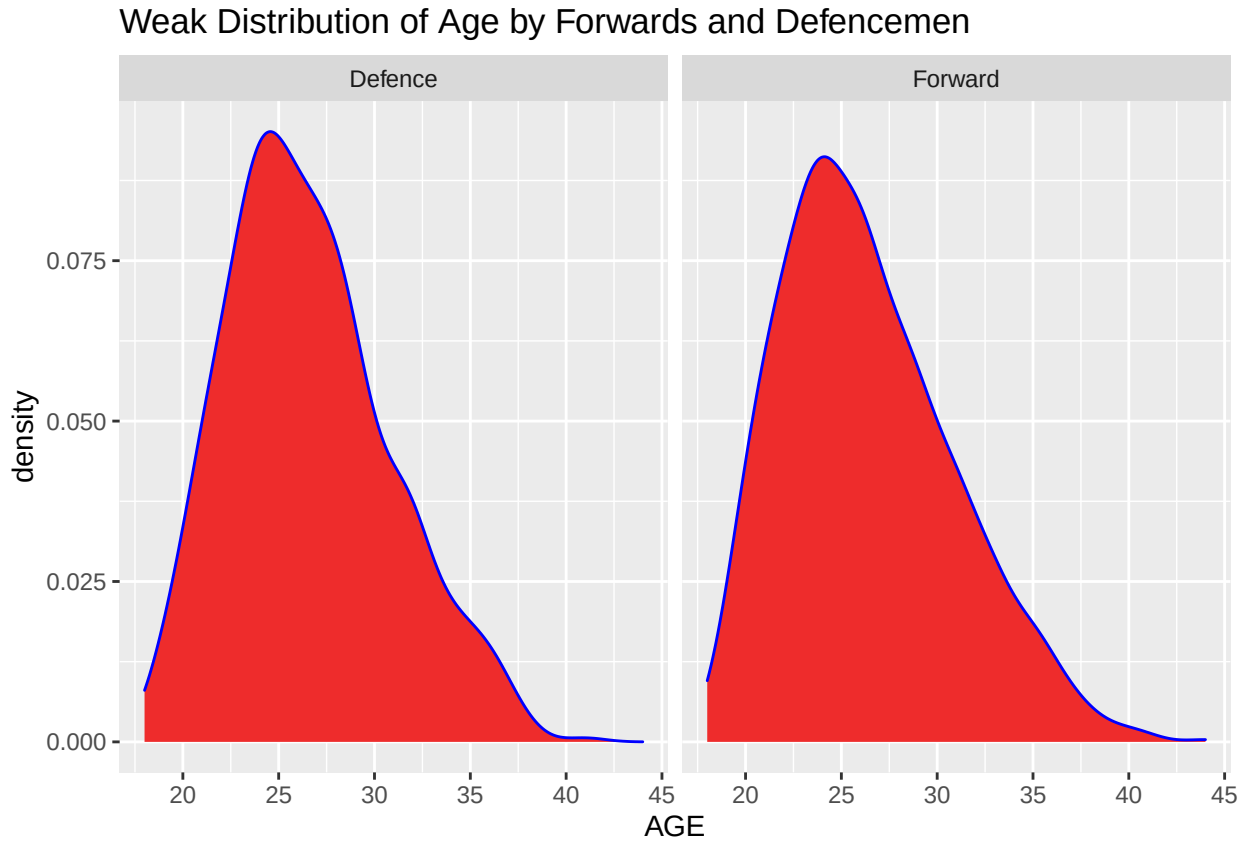
We see in the density plots expressed above that there is visible difference between the team rankings in accordance to player age distribution. The age distribution of the weak teams shows a steep peak that is centered at approximately 24 years of age. The bubble teams age distribution peaks at 25 years of age. The elite teams have bimodal age distribution peaks of 24.5 and 27 years of age. Doing some math it appears that the these bimodal values are tied to Position. The highest portion of players for elite forward positions (C,LW,RW) fall within one year difference around the first peak of 24.5 while th highest proportion of players for elite defence position (D) fall with in one year difference of the second peak 27 years. This phenomenon can be seen in the following density plot example.

2.2.3 Elite Distribution of Age by Forwards and Defencemen



The visualization above confirms that there is a distinct bimodal distribution of age for the elite teams. The distinction between age and position is quite clear for an elite team. How might this look for the weakly ranked teams?

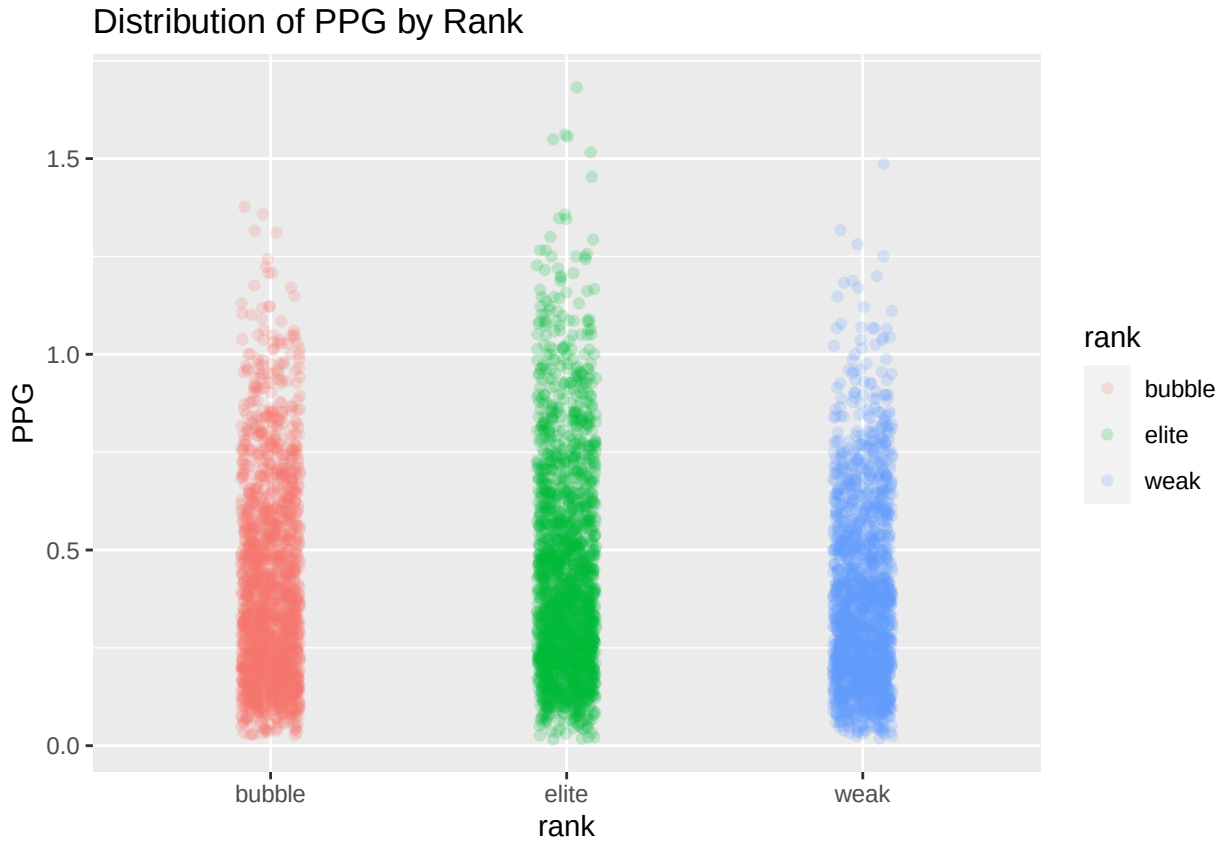
2.2.4 Weak Distribution of Age by Forwards and Defencemen



As we can see in the distribution the age separation for the two positions is not nearly as distinct as with the elite teams. From this information we can infer a correlation between elite teams and age distinction of players by position. An elite team is will generally have its highest distribution of forwards around 24.5 years of age and its highest distribution of defencemen around 27 years of age;

2.2.5 Distribution of Points per Game determined by Rank

As the objective of hockey is to score more points than the other team in order to win the match, a player with a high point per game total will be valuable to team success. By using a jitter plot we can see the differences in data between the three rankings of teams.

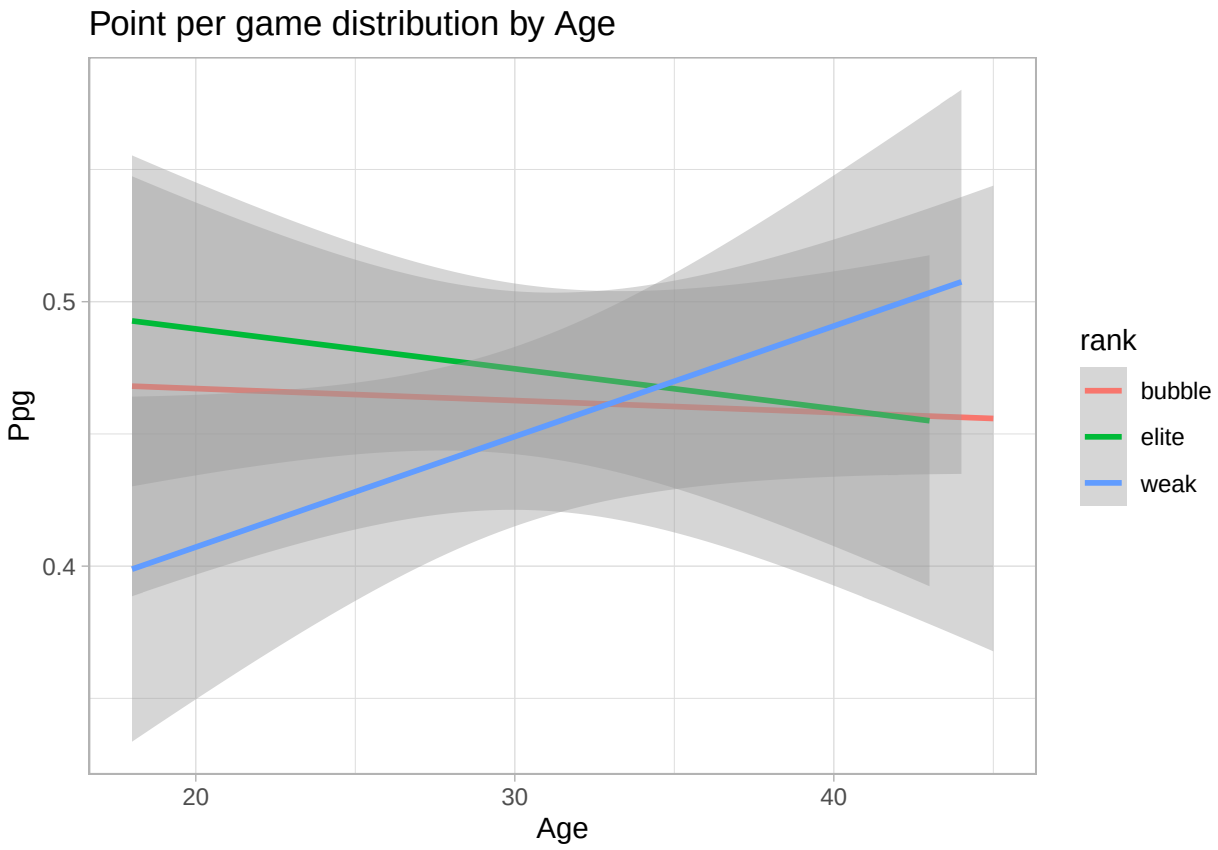


From the visual we can see that though the weak ranking contains the highest individual value for ppg, the elite column has a much higher overall total of ppg values. Summarizing the mean ppg per ranking shows elite with the highest average and weak with the lowest average.

rank	avg
elite	0.4344039
bubble	0.3996136
weak	0.3696536

2.2.6 Point distribution by Age

We have seen how elite teams have the highest mean age and seem most specific in the relevance of age distribution. How might the PPG be influence by Age for the three rankings? For this analysis we only will focus on the forward positions (C,R,L) as forwards traditionally are utilized to score points while the defencemen (D) traditionally used to defend against scoring.



The above graph displays ppg by age regression lines with confidence bands for each rank of team. As we can see from the linear interpretation of the data, weak teams are much more reliant upon older players to carry their scoring whereas the elite and bubble teams have a much more even distribution of scoring by age.

2.3 Model and Tuning Exploration

In this project we utilized several models in an attempt to obtain the best accuracy. These models consisted of Classification and Regression Tree (rpart), Robust Linear Discriminate Analysis (Linda), and eXtreme Gradient Boosting (xgbTree). As there is considerable variability in the time it takes to run these different models, time is included in the results as a consideration of efficiency. The models were performed on both the training set and test set in order to show the similarities and differences in the accuracy obtained for each dataset.

2.3.1 Step One: Classifier for Baseline Accuracy

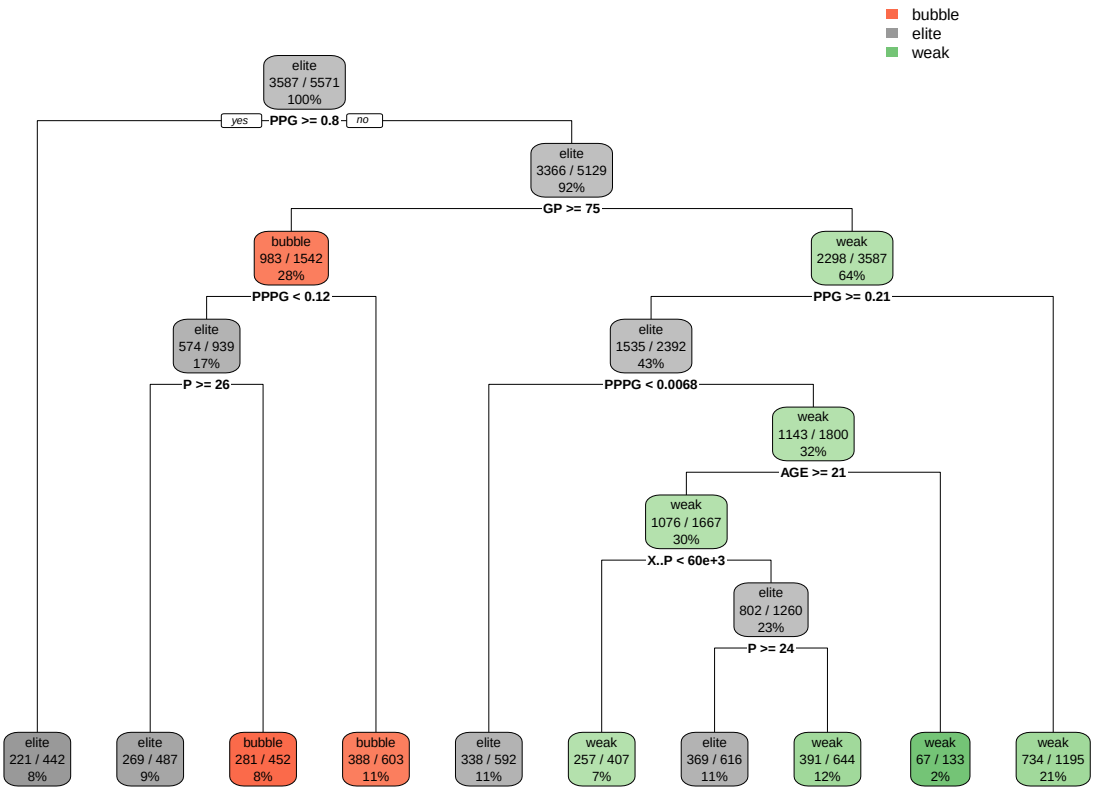
Guessing the accuracy can provide a baseline where results are obtained by simple random selection.

Predictor	Accuracy..train.	Accuracy..test.
Guess Model	0.2498654	0.2629816

Using a system of guessing we obtained an accuracy of 0.263 for the training data and 0.263 for the test data.

2.3.2 Step Two: Regression Tree

Regression Tree is the result of a tunable learning algorithm known as rpart. Using the statistical method of recursive partitioning, the rpart algorithm splits data into simpler versions of itself which then continue to split until the expectation of the algorithm parameters are met. The end result of the algorithm is a regression tree which can aid in the classification of data. Rpart contains one tunable hyperparameter, the complexity parameter, which is set at a default value of 0.01. A tunegrid dataframe containing a sequence of values between 0 and 0.05 was utilized to replace the default value.



Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
Rpart Model	0.4049542	0.3919598	38.09

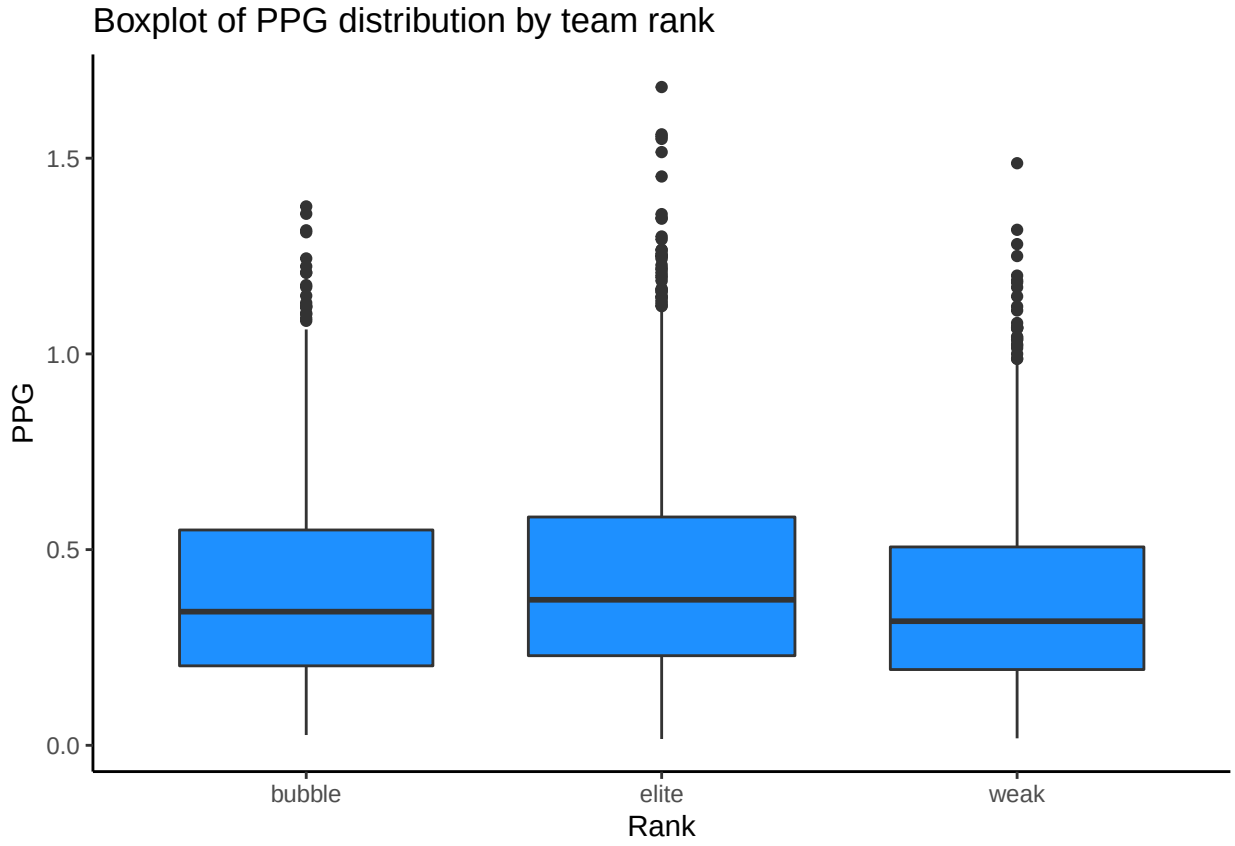
Each node shows the predicted rank, the predicted probability of a player being of that rank, and the percentage of observations in the node. The best tune of the complexity parameter was 0.004 there is a slight overfit as the train set performs better than the test set. For the training and test sets we obtained an accuracy of close to 40% which is much higher than the guessing model which is only around 25%. The accuracy obtained isn't particulary high but the nodes of the classification tree can provide insight on the data by determining which features were highlighted in the recursive modeling process. We see PPG, P, PPPG, GP, AGE, and X..P included as the relevant arguments in the decision flow of the classification tree. PPG appears to be important to all rankings while X..P is more relevant in defining the differences between the elite and weak ranking.

2.3.3 Step Two: Linda

While the rpart method contained one tunable hyperparameter (cp) there are many algorithms which perform without a tuning option, some examples include: Bayesian Generalized Linear Model, Gaussian Process, and Linear Discriminant Analysis. The benefit of these models with no tunable hyperparameters is that results are generally obtained quickly and simply. After testing several models with non-tunable parameters the highest accuracy was found using Robust Linear Discriminant Analysis (Linda).

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
Linda model	0.3972357	0.4036851	39.22

In considering why Linda outperformed the other non tunable algorithms, the Linda model being robust indicates that it is resistant to outliers and thus minimizes error to create a higher accuracy.



As shown in the above boxplot there are noticable outliers in the feature of ppg. Outliers are also consistently evident in the featured variables of PPPG, X..P, CAP.HIT and PPP.

2.3.4 Step Four: xgbTree

As Linda does not implement tunable hyperparameters the next step was to find a model that is similarly robust but also tunable. An algorithm with tunable hyperparamters can produce model specific optimization when training the data. XGBoost and boosting in general are very sensitive to outliers, this is because boosting builds each tree on previous trees' residuals/errors. Outliers will have much larger residuals than non-outliers, so boosting will focus a disproportionate amount of its attention on those points. In order to maximize the available training data and avoid overfitting, all xgbTree models were fit with a train control consisting of cross validation with 3 folds.

2.3.4.1 xgbTree with default parameters

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Default	0.455753	0.4455611	75.36

The xgbTree model using the default parameters resulted in a significant increase in accuracy in both training and test sets over the Linda and rpart models. The default tuning parameters for the algorithm are: nrounds = 150, max_depth = 1, eta = 0.3, gamma = 0, colsample_bytree = 0.8, min_child_weight = 1 and subsample = 0.75.

2.3.4.2 xgbTree with tuned hyperparameters By tuning the hyperparameters the optimal levels which produce the highest accuracy can be isolated. For this process each parameter was tuned individually with the best result then carried over to the next version of the model. The first step in tuning was to alter the default number of rounds (150) to instead cover a sequence of 100 to 1000 rounds increased by intervals of 50. The learning rate was also expanded to include 0.1, 0.2, 0.3, and 0.4

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Tune1	0.4573685	0.4371859	37.47

We see a silght change in the accuracy of the model once the initial hyperparameter tuning was performed. There is minimal increase in accuracy for both the training set and a slight decrease for the test set. The best tune for learning rate was 0.2.

2.3.5 Maximum Tree Depth

To search for a better fit the maximum tree depth count was then tuned to include all whole numbers between 1 and 5.

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Tune2	0.4758571	0.4355109	120.86

In this model training accuracy has increased and the test set accuracy is slightly decreased. The best tune for maximum tree depth was 1.

2.3.5.1 Tuning minimum child weight For the next step in tuning, the minimum child depth was expanded to include a range of values centered around the default value. In addition, the max depth was once again expanded upon by creating a whole number range of +/- around the best tuning found in the previous model.

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Tune3	0.6883863	0.4288107	131.28

With this tuning variation there was a substantial increase in the training accuracy while the test accuracy decreased slightly. The best tune for minimum child weight in this model was 1 and the best tune for max depth was 2.

2.3.5.2 Tuning subsample ratio of columns and observations In the next step, the best tune values of learning rate, max tree depth and minimum child weight were added to a new tuning grid which also contained multiple values for subsample ratio of columns and training observations. This process creates different random samples of the training data prior to growing trees which can help prevent overfitting.

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Tune4	0.5740442	0.440536	53.62

After tuning these hyperparameters there was a decrease in training accuracy but the test accuracy has been slightly elevated. The results are consistent in the expectation that training accuracy would decrease from reduced overfitting but in turn the test accuracy should be boosted. The best tune for column subsample in this model was 0.8 and the best tune for subsample ratio of training observations was 0.5

2.3.5.3 Tuning the Gamma hyperparameter Gamma is a regularization parameter for xgbTree which represents how much the loss will be reduced by for the model to perform a split resulting in a tree.The higher the level of Gamma the higher the degree of regularization in the model. In this step the model was performed with a degree of Gamma sequenced from 0 to 1 by an interval of 0.5.

Predictor	Accuracy (train)	Accuracy (test)	Time (secs)
xgbTree - Tune5	0.541016	0.4187605	47.38

By altering the Gamma values the best tuned model had a decrease in the training and test set accuracy.

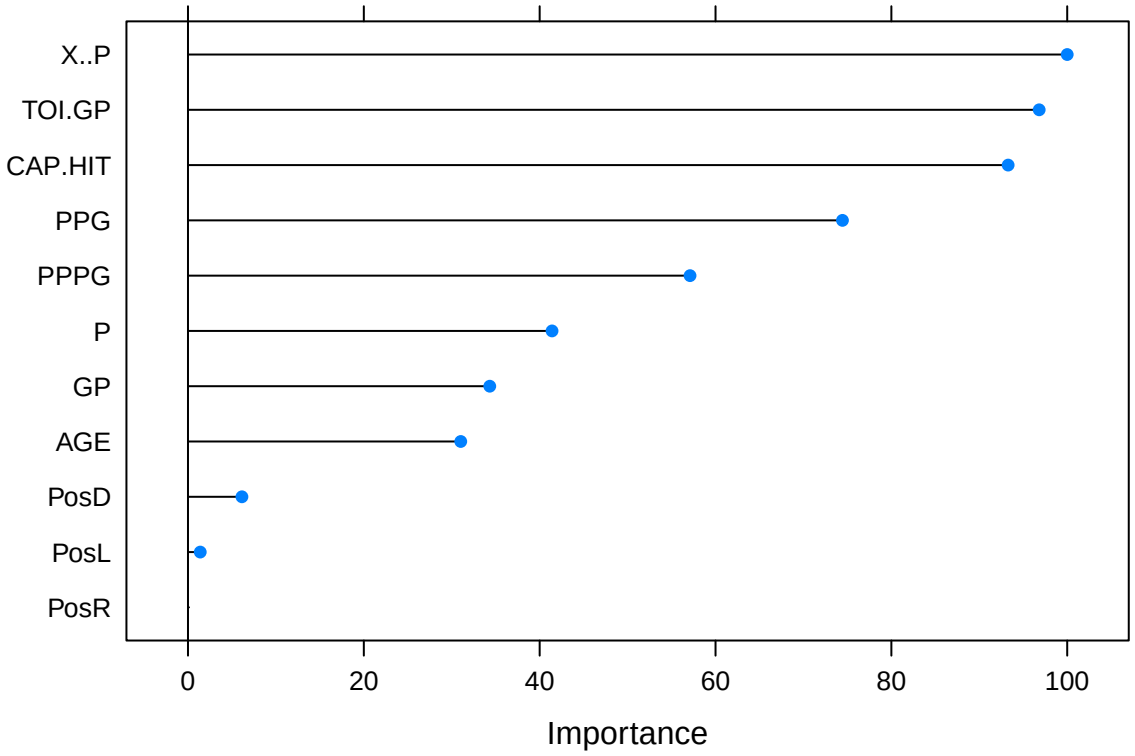
3 Results

The table below shows our results through all explored versions of our model.

Predictor	Accuracy..train.	Accuracy..test.	Time.secs.
Rpart Model	0.4049542	0.3919598	38.09
Linda model	0.3972357	0.4036851	39.22
xgbTree - Default	0.4557530	0.4455611	75.36
xgbTree - Tune1	0.4573685	0.4371859	37.47
xgbTree - Tune2	0.4758571	0.4355109	120.86
xgbTree - Tune3	0.6883863	0.4288107	131.28
xgbTree - Tune4	0.5740442	0.4405360	53.62
xgbTree - Tune5	0.5410160	0.4187605	48.79

The third tuning of xgbTree hyperparameters produced the best results for the training set accuracy and the best test accuracy was achieved in the initial version of xgbTree. The best model has surpassed the training set classifier for baseline accuracy by 0.439

The Variable Importance for the best training model is represented as:



Four top variables really stand out for importance (X..P, TOI.GP, CAP.HIT, PPG), while four variables are middling (PPPG, P, GP, AGE) and two variables are low importance (PosD, PosL). PosR is also listed but the value is essentially negligible.

A team that is seeking to attain elite status should consider focusing resources and attention accordingly on these top features. If you want your team to be elite a positive strategy would be to model ice time of players in the same way as other elite teams, with an extra emphasis on Defence and Left Wing positions. This strategy could entail spending extra money to hire the best coach who is in charge of defencemen. A team could also consider spending above budget to obtain the best free agent Left Wing player who fits closely with the peak age of the elite forwards. You want to have a Cost per Point (X..P) similar to the elites and you can evaluate your players when offering salary contract extensions in that regard.

As the accuracy certainly isn't perfect we can't say for certain that a team will have more success if they focus on these areas, however we have found a significant correlation between team performance and these features. By studying the features of the elite teams, a performance expectation of the players can be more clearly defined. Further testing could be conducted between variables to gain more insight. For example, if a correlation between age and average ice time for elite teams vs weak teams is identified, a weight potentially could be applied to the model.

*R version 4.1.0 was used for the modeling in this project in RStudio for Windows. Slightly different accuracy was found in xgbTree modeling when using earlier versions of R in Rstudio for MacOS. To be sure results precisely match please use R version 4.1.0.

4 Conclusion

In this project publicly available nhl player and team data was imported, engineered, analyzed through graphs and summary statistics and then utilized to make predictions through machine learning models. In comparing the models that were used, two separate tuned versions of eXtreme Gradient Boosting provided the best results for training and test accuracy. The most important feature to predict rank in the training data was the cost per point variable. By examining the results, particularly the variable importance, a National Hockey League team can find insights on how to achieve greater success. The accuracy obtained in the project doesn't guarantee success in adaptation of the analysis presented, however further examination of correlation between features may provide even better predictive results.