

## **Data Engineering Capstone: Predicting Future NHL Season Performance**

**Tarek Clarke**

### **Research Question**

In the years leading up to the 2004-2005 National Hockey League (NHL) season, negotiations with the National Hockey League Player's Association (NHLPA) reached an impasse, leading to the cancellation of the season. Before the beginning of the following season, the NHL and NHLPA reached an agreement, which brought significant changes to the league's structure, which are still in effect today. Notably, the introduction of a salary cap, which limits the salary expenditure for each franchise to a uniform limit per season, and the replacement of ties with shootouts had a significant impact on parity and seasonal rankings. This new era of NHL season brought about a natural disconnect from the previous era, making it difficult to compare to previous years. (Burnside, 2005)

For my Capstone, I will be predicting the outcome of future NHL seasons based on team performance from the 2005-2006 season onward. Before conducting analysis, some considerations must be made.

1. The 2012-2013 season began late due to another partial season cancellation due to a strike, limiting the games played to 48.
2. The 2019-2020 season ended early due to sudden COVID lockdowns, and teams did not all play the same number of games.
3. The 2020-2021 season was again shortened due to COVID, but international travel restrictions limited teams to playing within their own divisions for the regular season, which does not have the same structure as the remaining seasons since the 2004-2005 lockout.

These three seasons will be considered in my analysis due to their atypical structure. I will be focusing on complete, 82 game seasons where each team plays every team at least twice. Due to salary cap circumvention, I will not be including playoff performance in this model.

There are some additional structural aspects of the league that are important to be considered.

1. In 2010, the Atlanta Thrashers relocated to Winnipeg, retaining their history, roster, staff and records. For this capstone, I will be treating them as one franchise. They will be referred to as Winnipeg in the data and paper moving forward.

2. In 2014, the Phoenix Coyotes were renamed the Arizona Coyotes. In 2024, they relocated to Salt Lake City, and were rebranded as the Utah Mammoth, retaining their history, roster, staff and records. This franchise will be referred to as the Utah Mammoth moving forward.
3. In 2017, the NHL awarded Las Vegas an expansion team, the Vegas Golden Knights. Their roster was built via an expansion draft, where unprotected players from the previously existing teams are selected by the expansion team to build a roster.
4. Similarly, in 2021, the NHL awarded Seattle an expansion team, the Seattle Kraken, which was also constructed via an expansion draft.
5. Beginning with the 2013-2014 season, a realignment of the existing divisions was executed, and wild card seeding was introduced for playoff rankings. As playoff ranking will not be modelled, it does not impact the model.
6. Team scoring is uniform from 2005 onwards, with a win awarding a team 2 points, an overtime loss or shootout loss awarding a team 1 point, and a regulation loss awards a team no points.

The null hypothesis for this analysis project is that team point totals can be accurately predicted using secondary variables with 90% accuracy.

The alternative hypothesis is that historical point totals cannot be used to accurately predict performance.

### **Data Collection**

The NHL API was used to collect data on NHL team performance since the 2005-06 season. The NHL does not have an exhaustive set of documentation, but the NHL statistics community has created documentation after exploring the API (Zmalski, 2025). To begin, I set my AWS access keys in RStudio to access my AWS bucket. The step where the keys were defined are redacted for security.

The first step in accessing the API was declaring the end dates for each season, as shown in the screenshot that follows:

```
# set bucket name  
  
bucket_name <- "nhld610"  
  
# season end dates  
seasons <- list()  
  "20052006" = "2006-04-18",  
  "20062007" = "2007-04-08",  
  "20072008" = "2008-04-06",  
  "20082009" = "2009-04-11",  
  "20092010" = "2010-04-11",  
  "20102011" = "2011-04-10",  
  "20112012" = "2012-04-01",  
  "20122013" = "2013-04-27",  
  "20132014" = "2014-04-13",  
  "20142015" = "2015-04-11",  
  "20152016" = "2016-04-10",  
  "20162017" = "2017-04-09",  
  "20172018" = "2018-04-07",  
  "20182019" = "2019-04-06",  
  "20192020" = "2020-03-11",  
  "20202021" = "2021-05-19",  
  "20212022" = "2022-04-29",  
  "20222023" = "2023-04-13",  
  "20232024" = "2024-04-18",  
  "20242025" = "2025-04-17"  
()
```

Afterwards, the API was called for season standings by year. Every team's final season totals represent a row and are exported to the S3 bucket as a JSON file.

```

39 ▼ for (season_code in names(seasons)) {
40   end_date <- seasons[[season_code]]
41   url <- sprintf("https://api-web.nhl.com/v1/standings/%s", end_date)
42   response <- GET(url)
43 ▼ if (status_code(response) == 200) {
44   standings <- content(response, as = "text", encoding = "UTF-8")
45   local_file <- sprintf("standings_%s.json", season_code)
46   writeLines(standings, local_file, useBytes = TRUE)
47   s3_object <- sprintf("nhl/standings/standings_%s.json", season_code)
48 ▼ result <- tryCatch({
49   put_object(
50     file = local_file,
51     object = s3_object,
52     bucket = bucket_name
53   )
54 ▼ }, error = function(e) {
55   message(sprintf("S3 upload failed for %s: %s", season_code, e$message))
56   FALSE
57 })
58 ▼ if (isTRUE(result)) {
59   message(sprintf("Uploaded standings for %s to S3.", season_code))
60 ▼ } else {
61   warning(sprintf("Failed to upload standings for %s to S3.", season_code))
62 }
63 # Optionally, remove local file after upload
64 # file.remove(local_file)
65 ▼ } else {
66   warning(sprintf("Failed to fetch standings for %s (HTTP %d)", season_code, status_code(response)))
67 }
68 }

```

## Data Extraction and Preparation

The JSON files were then bound converted to data frames and bound to create a time series for each team from the beginning of the 2005-06 season to the end of the 2024-25 season.

```
12 # List all objects in the S3 folder
13 s3_objects <- get_bucket("nhld610", prefix = "nhl/standings/", max = Inf)
14 json_files <- vapply(s3_objects, function(x) x$key, character(1))
15 json_files <- json_files[grepl("\\.json$", json_files)]
16
17 json_list <- lapply(json_files, function(key) {
18   obj <- get_object(object = key, bucket = "nhld610")
19   fromJSON(rawToChar(obj), flatten = TRUE)
20 })
21 names(json_list) <- basename(json_files)
22
23
24
25
26 season0506<-as.data.frame(json_list[[1]])
27 season0607<-as.data.frame(json_list[[2]])
28 season0708<-as.data.frame(json_list[[3]])
29 season0809<-as.data.frame(json_list[[4]])
30 season0910<-as.data.frame(json_list[[5]])
31 season1011<-as.data.frame(json_list[[6]])
32 season1112<-as.data.frame(json_list[[7]])
33 season1213<-as.data.frame(json_list[[8]])
34 season1314<-as.data.frame(json_list[[9]])
35 season1415<-as.data.frame(json_list[[10]])
36 season1516<-as.data.frame(json_list[[11]])
37 season1617<-as.data.frame(json_list[[12]])
38 season1718<-as.data.frame(json_list[[13]])
39 season1819<-as.data.frame(json_list[[14]])
40 season1920<-as.data.frame(json_list[[15]])
41 season2021<-as.data.frame(json_list[[16]])
42 season2122<-as.data.frame(json_list[[17]])
43 season2223<-as.data.frame(json_list[[18]])
44 season2324<-as.data.frame(json_list[[19]])
45 season2425<-as.data.frame(json_list[[20]])
```

```

57 #drop "standings.conferenceAbbrev" "standings.conferenceName" from everything but 2021
58 season0506<-select(season0506,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
59 season0607<-select(season0607,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
60 season0708<-select(season0708,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
61 season0809<-select(season0809,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
62 season0910<-select(season0910,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
63 season1011<-select(season1011,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
64 season1112<-select(season1112,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
65 season1213<-select(season1213,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
66 season1314<-select(season1314,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
67 season1415<-select(season1415,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
68 season1516<-select(season1516,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
69 season1617<-select(season1617,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
70 season1718<-select(season1718,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
71 season1819<-select(season1819,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
72 season1920<-select(season1920,-standings.conferenceAbbrev,-standings.conferenceName)
73 season2021<-select(season2021,-standings.clinchIndicator)
74 season2122<-select(season2122,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
75 season2223<-select(season2223,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
76 season2324<-select(season2324,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName)
77 season2425<-select(season2425,-standings.clinchIndicator,-standings.conferenceAbbrev,-standings.conferenceName,-standingsDateTimeUtc)

78
79
80 all_seasons<-rbind(season0506,season0607,season0708,season0809,season0910,season1011,season1112,season1213,season1314,season1415,
81   |season1516,season1617,season1718,season1819,season1920,season2021,season2122,season2223,season2324,season2425)
82

```

With the data consolidated into one data frame, I added some important identifiers. I added a column with the current name for each franchise, and added division names based on season to allow for potential playoff seeding to be added to the model if needed. I also removed less useful variables, such as French names, home/road games played and ties, which are not relevant to this model. The following screenshot shows this process:

```

83 #remove French and useless variables
84 names(all_seasons) <- sub("standings.", "", names(all_seasons))
85
86 all_seasons<-select(all_seasons,-teamCommonName.fr,-teamName.fr,-placeName.fr,-homeTies,-l10GamesPlayed,-date,-gameTypeId,, -homeGamesPlayed,
87   |-l10Ties,-roadGamesPlayed,-ties,-placeName.default,-teamCommonName.default,-roadTies,-teamLogo,-wildCardIndicator)
88
89 all_seasons$modernName<-all_seasons$teamName
90 all_seasons$modernName[all_seasons$teamName == "Phoenix Coyotes"] <- "Utah Mammoth"
91 all_seasons$modernName[all_seasons$teamName == "Arizona Coyotes"] <- "Utah Mammoth"
92 all_seasons$modernName[all_seasons$teamName == "Atlanta Thrashers"] <- "Winnipeg Jets"
93
94 all_seasons$modernAbbrev<-all_seasons$teamAbbrev
95 all_seasons$modernAbbrev[all_seasons$teamAbbrev == "PHX"] <- "UTA"
96 all_seasons$modernAbbrev[all_seasons$teamAbbrev == "ARI"] <- "UTA"
97 all_seasons$modernAbbrev[all_seasons$teamAbbrev == "ATL"] <- "WPG"
98
99 #new divisions
100 all_seasons$conference[all_seasons$divisionAbbrev == "CEN"] <- "Western"
101 all_seasons$conference[all_seasons$divisionAbbrev == "PAC"] <- "Western"
102 all_seasons$conference[all_seasons$divisionAbbrev == "ATL"] <- "Eastern"
103 all_seasons$conference[all_seasons$divisionAbbrev == "M"] <- "Eastern"
104
105 #old divisions
106 all_seasons$conference[all_seasons$divisionAbbrev == "A"] <- "Eastern"
107 all_seasons$conference[all_seasons$divisionAbbrev == "NE"] <- "Eastern"
108 all_seasons$conference[all_seasons$divisionAbbrev == "SE"] <- "Eastern"
109 all_seasons$conference[all_seasons$divisionAbbrev == "C"] <- "Western"
110 all_seasons$conference[all_seasons$divisionAbbrev == "NW"] <- "Western"
111 all_seasons$conference[all_seasons$divisionAbbrev == "P"] <- "Western"
112
113 #covid divisions
114 all_seasons$conference[all_seasons$divisionAbbrev == "NTH"] <- "Eastern"
115 all_seasons$conference[all_seasons$divisionAbbrev == "WST"] <- "Western"
116 all_seasons$conference[all_seasons$divisionAbbrev == "EST"] <- "Eastern"
117 all_seasons$conference[all_seasons$divisionAbbrev == "CEN"] <- "Western"
118
119 cols_first<-c("seasonId","conference","divisionAbbrev","divisionName","teamAbbrev.default","teamName.default","modernAbbrev","modernName",
120   "gamesPlayed","wins","losses","otLosses","points")
121

```

Afterwards, I added an index for each season and removed the 2012-13, 2019-20 and 2020-21 seasons because they are not used in this model. This is shown in the following screenshot:

```
123 all_seasons<-all_seasons[,c(cols_first, setdiff(names(all_seasons), cols_first))]  
124  
125 all_seasons$streak<-paste0(all_seasons$streakCount, all_seasons$streakCode)  
126 all_seasons<-select(all_seasons, -streakCount, -streakCode)  
127 all_seasons$seasonId<-as.character(all_seasons$seasonId)  
128  
129 all_seasons_82<-all_seasons %>%  
130   filter(!seasonId %in% c('20122013', '20192020', '20202021'))  
131  
132 all_seasons_82$seasonId<-as.numeric(all_seasons_82$seasonId)  
133  
134 all_seasons_82$season<-all_seasons_82$seasonId  
135  
136 all_seasons_82 <- all_seasons_82 %>%  
137   mutate(seasonId = case_when(  
138     season == 20052006 ~ 1,  
139     season == 20062007 ~ 2,  
140     season == 20072008 ~ 3,  
141     season == 20082009 ~ 4,  
142     season == 20092010 ~ 5,  
143     season == 20102011 ~ 6,  
144     season == 20112012 ~ 7,  
145     season == 20122013 ~ 8,  
146     season == 20132014 ~ 9,  
147     season == 20142015 ~ 10,  
148     season == 20152016 ~ 11,  
149     season == 20162017 ~ 12,  
150     season == 20172018 ~ 13,  
151     season == 20182019 ~ 14,  
152     season == 20192020 ~ 15,  
153     season == 20202021 ~ 16,  
154     season == 20212022 ~ 17,  
155     season == 20222023 ~ 18,  
156     season == 20232024 ~ 19,  
157     season == 20242025 ~ 20,  
158     TRUE ~ NA_real_  
159   ))
```

Afterwards, I wrote the dataset to the S3 bucket and exported a csv to my local drive for quick analysis, as shown in the following screenshot.

```

161 #write all_seasons to the bucket
162 s3write_using(
163   all_seasons,
164   FUN = write.csv,
165   object = "nhl/all_seasons.csv",    # S3 key (path/filename in the bucket)
166   bucket = "nhld610",
167   row.names = FALSE
168 )
169
170 s3write_using(
171   all_seasons_82,
172   FUN = write.csv,
173   object = "nhl/all_seasons_82.csv",    # S3 key (path/filename in the bucket)
174   bucket = "nhld610",
175   row.names = FALSE
176 )
177
178
179 write.csv(all_seasons,"all_seasons.csv",row.names=FALSE)
180
181 write.csv(all_seasons_82,"all_seasons_82.csv",row.names=FALSE)
182
183

```

I then organized the dataset by season and franchise, and created a training dataset with the season data.

```

161 #write all_seasons to the bucket
162 s3write_using(
163   all_seasons,
164   FUN = write.csv,
165   object = "nhl/all_seasons.csv",    # S3 key (path/filename in the bucket)
166   bucket = "nhld610",
167   row.names = FALSE
168 )
169
170 s3write_using(
171   all_seasons_82,
172   FUN = write.csv,
173   object = "nhl/all_seasons_82.csv",    # S3 key (path/filename in the bucket)
174   bucket = "nhld610",
175   row.names = FALSE
176 )
177
178
179 write.csv(all_seasons,"all_seasons.csv",row.names=FALSE)
180
181 write.csv(all_seasons_82,"all_seasons_82.csv",row.names=FALSE)
182
183

```

For this data extraction and cleaning process, I decided to use RStudio, as I use R professionally over Python, but the process can be replicated in Python as well. I used AWS as I was most comfortable with it, but any cloud storage solution can be used. Although I am more comfortable with R, this code could be written in Python and run on Amazon Redshift, which would allow me to avoid using local processing power. The extraction process was quite slow for R, and if my computer was interrupted, the process would not complete. If I used Pyhton and Redshift, I could rely on AWS' processing power to run the process off site. It is also possible to install RStudio on AWS EC2, but it had potential additional costs. The benefit of using RStudio locally allowed me to avoid paying for EC2, but it would be useful for a larger data extraction process. AWS was an easy off site data warehousing solution, but it does carry a cost as well. In this case, my storage cost was \$0.01 USD, but it can be significant as the project increases in scale.

## Analysis

In order to process the data, I created a test dataset containing all remaining NHL seasons in the S3 bucket. To begin, I grouped the rows by season and team, and created a new data frame called train to use as the training set.

```
214 all_seasons_82 <- all_seasons_82 %>%
215   arrange(modernAbbrev, seasonId) %>%
216   group_by(modernAbbrev) %>%
217   mutate(across(
218     .cols = -c(seasonId, season, conference, divisionAbbrev, divisionName, teamAbbrev.default, teamName.default),
219     .names = "{.col}_prev"
220   )) %>%
221   ungroup()
222 
223 #training and test sets
224 train<-all_seasons_82
225
```

To ensure the model works correctly, I removed the first occurrence of each team (2005-06 for existing/relocated franchises and 2017 for Vegas and 2021 for Seattle, respectively. I then created a random forest model using the previous year's season totals. The variables I used are:

- Points
- Wins
- Losses
- Overtime Losses
- Goals For

- Goals Against
- Goal Differential
- The streak the team ended the season on

I then repeated the process for home and road games.

The data up to 2024-25 was used, so I could predict the results of the season that had most recently concluded, and compare the validity of the model.

```

198 #remove SEA & VGK's first seasons
199
200 train <- all_seasons_82 %>%
201   filter(seasonId < 19) %>% # use seasons 1-18 [2005-2023]
202   filter(!is.na(points_prev))
203
204 #prep prediction for 2024-25
205 predict_teams <- all_seasons_82 %>%
206   filter(seasonId == 18) %>% # use 2023-24 data
207   mutate(seasonId = 19) %>%
208
209   mutate(
210     points_prev = points,
211     wins_prev = wins,
212     losses_prev = losses,
213     otLosses_prev = otLosses,
214     goalFor_prev = goalFor,
215     goalAgainst_prev = goalAgainst,
216     goalDifferential_prev = goalDifferential,
217     streak_prev = streak,
218     homeWins_prev = homeWins,
219     homeLosses_prev = homeLosses,
220     homeOtLosses_prev = homeOtLosses,
221     homeGoalsFor_prev = homeGoalsFor,
222     homeGoalsAgainst_prev = homeGoalsAgainst,
223     homeGoalDifferential_prev = homeGoalDifferential,
224     roadWins_prev = roadWins,
225     roadLosses_prev = roadLosses,
226     roadOtLosses_prev = roadOtLosses,
227     roadGoalsFor_prev = roadGoalsFor,
228     roadGoalsAgainst_prev = roadGoalsAgainst,
229     roadGoalDifferential_prev = roadGoalDifferential
230   )
231

```

I trained the model with a random forest distribution and predicted the data for the 2024-25 season using the following code:

```
232 #train random forest model
233
234 model <- randomForest(points ~
235   points_prev +
236   wins_prev +
237   losses_prev +
238   otLosses_prev +
239   goalFor_prev +
240   goalAgainst_prev +
241   goalDifferential_prev +
242   streak_prev +
243   homeWins_prev +
244   homeLosses_prev +
245   homeOtLosses_prev +
246   homeGoalsFor_prev +
247   homeGoalsAgainst_prev +
248   homeGoalDifferential_prev +
249   roadWins_prev +
250   roadLosses_prev +
251   roadOtLosses_prev +
252   roadGoalsFor_prev +
253   roadGoalsAgainst_prev +
254   roadGoalDifferential_prev,
255   data = train,
256   ntree = 500)
257
258 #generate predictions
259 predictions <- predict(model, newdata = predict_teams)
260
261 #rank the teams
262 standings_2024 <- predict_teams %>%
263   mutate(points = round(predictions, 1)) %>%
264   select(modernAbbrev, points) %>%
265   arrange(desc(points)) %>%
266   mutate(league_rank = row_number())
267
268 standings_2024$points<-as.integer(standings_2024$points)
269
```

The model predicted the standings to be as follows:

	modernAbbrev	points	league_rank
1	BOS	128	1
2	CAR	112	2
3	NJD	112	3
4	TOR	111	4
5	VGK	110	5
6	EDM	109	6
7	DAL	108	7
8	COL	107	8
9	NYR	107	9
10	LAK	103	10
11	MIN	102	11
12	SEA	99	12
13	TBL	98	13
14	WPG	95	14
15	NYI	93	15
16	CGY	92	16
17	FLA	92	17
18	NSH	92	18
19	PIT	90	19
20	BUF	89	20
21	OTT	85	21
22	VAN	82	22
23	STL	80	23
24	WSH	80	24
25	DET	79	25
26	PHI	74	26
27	UTA	69	27
28	MTL	67	28
29	SJS	60	29
30	CBJ	58	30
31	CHI	58	31
32	ANA	57	32

Comparing the actual season results from NHL.com:

Team	Season	P
Winnipeg Jets	20242025	116
Washington Capitals	20242025	111
Vegas Golden Knights	20242025	110
Toronto Maple Leafs	20242025	108
Dallas Stars	20242025	106
Los Angeles Kings	20242025	105
Colorado Avalanche	20242025	102
Tampa Bay Lightning	20242025	102
Edmonton Oilers	20242025	101
Carolina Hurricanes	20242025	99
Florida Panthers	20242025	98
Ottawa Senators	20242025	97
Minnesota Wild	20242025	97
St. Louis Blues	20242025	96
Calgary Flames	20242025	96
New Jersey Devils	20242025	91
Montréal Canadiens	20242025	91
Vancouver Canucks	20242025	90
Columbus Blue Jackets	20242025	89
Utah Hockey Club	20242025	89
Detroit Red Wings	20242025	86
New York Rangers	20242025	85
New York Islanders	20242025	82
Anaheim Ducks	20242025	80
Pittsburgh Penguins	20242025	80
Buffalo Sabres	20242025	79
Seattle Kraken	20242025	76
Philadelphia Flyers	20242025	76
Boston Bruins	20242025	76
Nashville Predators	20242025	68
Chicago Blackhawks	20242025	61
San Jose Sharks	20242025	52

(NHL, 2025)

Comparing the data to the real 2024-25 season, the model had some significant differences. Notably, the top team in the model, Boston was the 4<sup>th</sup> worst performing team in the actual 2024-25 season. The top team in the real 2024-25 season, Winnipeg was

ranked 14<sup>th</sup> in the model. This model does not satisfy the null hypothesis. Notably, some teams were very close to their prediction, such as Chicago, who were 3 points off.

Some attributes that can be incorporated into this model are per game rows, with an aggregation to generate the team totals. For example, the NHL API includes game logs, which include player statistics per game, which can shed light on high performers. Additionally, the model does not incorporate player transactions, which can have a major impact on the data. For example, Boston lost many significant high performing players in the offseason before the 2024-25 season, which are not reflected in the data. Additionally, due to the relatively short careers of NHL players, using data dating back to 2005 has very little continuity from 2005 to 2024, as very few players from those seasons remain in today's NHL. A rolling average with a more limited range of seasons may be beneficial for future iterations of this project.

As mentioned before, running this process on a local computer can be very slow, and implementing an instance of RStudio on EC2 to import player and team data from 2005 onwards can save a considerable amount of time, as it took over 2 hours to import game log data on a personal computer.

#### Sources:

Burnside, S. (2005, October 4). Remember the rules: A look at NHL's changes. ESPN.  
<https://www.espn.com/nhl/preview2005/news/story?id=2180808>

Zmalski. (2025). NHL-API-Reference Computer software. GitHub.  
<https://github.com/Zmalski/NHL-API-Reference>

National Hockey League. (2025). Team stats. NHL. <https://www.nhl.com/stats/teams>