DOKUMENTATION DATENBESCHAFFUNG

NHL Spielerverbindungen

FHNW – Modul SNA
Joël Winter, Manuel Riedi, Janick Hürzeler

1 Einleitung

Die Daten für unsere Soziale Netzwerkanalyse stammen von einer von der NHL zur Verfügung gestellten, jedoch nicht dokumentierten API (https://statsapi.web.nhl.com/api/v1).Um trotzdem eine Übersicht über den Umfang der API zu erhalten, wurde auf ein Community-Projekt zurückgegriffen, welches Teile der API dokumentiert hat (https://gitlab.com/dword4/nhlapi/blob/master/stats-api.md).

Als zentraler Auswertungspunkt möchten wir herausfinden, welche Spieler über längere Zeit mit denselben Mitspieler zusammengespielt haben. Auf dieser Grundlage möchten wir dann die einzelnen Analysen, wie sie im Meilenstein aufgelistet sind, durchführen.

Da die Daten von offizieller Seite der NHL kommen, gehen wir grundsätzlich von deren Korrektheit aus. Auch schien uns ein Vergleich mit alternativen Datenquellen unnötig, da solche wahrscheinlich auch auf Daten der NHL beruhen. Nichtsdestotrotz haben wir die Kader der Teams mit den Inhalten auf der NHL.com-Webseite stichprobenartig abgeglichen. Die Webseite (z.B. https://www.nhl.com/wild/roster/2018 für den Kader der «Minnesota Wild» in der Saison 2018/2019) bietet Filter für Team und Saison und listet die Spieler auf. Für den Vergleich transformierten wir die JSON-Daten in KNIME zu Tabellen. Dabei konnten wir keine Differenzen feststellen. Eine Bestätigung für einen wichtigen Punkt lieferte uns der Vergleich dennoch: Spieler, die während einer Saison das Team wechselten, werden jeweils in beiden Kaderlisten aufgelistet (z.B. der Tausch von Nino Niederreiter und Victor Rask in der Saison 2018/2019 zwischen den Carolina Hurricanes und den Minnesota Wild).

2 Datenbeschaffung

Der grobe Aufbau unserer Datenbeschaffung sieht wie folgt aus: Wir laden die Daten über eine API der offiziellen NHL-Seite (https://statsapi.web.nhl.com/api/v1) herunter. Mit einem «API-Fetcher», der in Python programmiert wurde, sprechen wir die API an und speichern die Daten im JSON-Format. Mit KNIME werden die Daten so transformiert, dass die Spieler-Knoten aufbereitet und die Kanten anhand der gemeinsamen Saison berechnet werden können. Diese Daten exportieren wir als CSV um sie für die Soziale Netzwerkanalyse in Gephi zu importieren.

2.1 Jupyter Notebook als API-Fetcher

In einem ersten Schritt wurde ein API-Fetcher in Python programmiert, der in Form eines Jupyter Notebooks die benötigten API-Calls pro Jahr durchführt. Die geladenen Daten einer Anfrage werden dabei als JSON-Datei (UTF8-Encoding) gespeichert, damit diese zu einem späteren Zeitpunkt nicht erneut geladen werden müssen.
Die folgenden Schritte im API-Fetcher beziehen sich jeweils auf die Daten von einem Jahr. Die entsprechenden Aufrufe wurden dann für die Jahre 2004 bis 2018 ausgeführt.
Zuerst wurden alle in dieser Saison in der NHL teilnehmenden Teams und deren weiteren Informationen, wie z.B. das Spielerkader, geladen. Dazu wird die Methode “callYearWithAllTeams” verwendet. Dabei sammeln wir alle Team-IDs für dieses Jahr und anhand dieser IDs laden wir die Teamdaten.
Als zweiter Aufruf laden wir die Spielerdaten. In der Methode “callAllPlayerDetailByYear” iterieren wir über alle Teams und rufen dabei “callAllPlayerDetailFromRoster” auf. Dort

iterieren wir über alle Spieler dieses Teams und laden mittels API-Call die detaillierten Spielerdaten.

Mit dieser Vorgehensweise haben wir sichergestellt, dass wir von keinem Spieler Daten geholt haben, der keinem Team zugeordnet ist.

Die JSON-Dateien werden in die Ordner player, team und year abgelegt. Nachfolgend sind die Inhalte der Ordner kurz beschrieben:

year
Eine JSON-Datei pro Saison, mit dem Dateinamen «*year*.json», wobei *year* jeweils dem «ersten» Jahr einer Saison entspricht (z.B. zur Saison **2014**/2015 gehört die Datei «**2014**.json»). Pro Datei sind alle in dieser Saison in der NHL teilnehmenden Teams aufgelistet.

team
Eine JSON-Datei pro Team und Saison, mit dem Dateinamen «*year_teamId*.json», wobei *year* dem «ersten» Jahr einer Saison entspricht und *teamId* der eindeutigen Id eines Teams gemäss den Abfragen im year-Ordner.

player
Eine JSON-Datei pro Spieler, mit dem Dateinamen «*playerId*.json», wobei *playerId* der eindeutigen Id eines Spielers entspricht, gemäss dem Kader («roster») aus den Abfragen im team-Ordner.

Zur Benutzung des Jupyter-Notebook wurde ein README erstellt und im Ordner “Abgabe/Source” abgelegt.

2.2 KNIME

Mit KNIME werden die gesammelten Daten aus den JSON-Files zu einem Node- und einem Edge-Table transformiert, sodass sie in Gephi optimal eingesetzt werden können.

Hinweis: Folgend beschriebener Knime-Workflow wird im Anhang (siehe Kapitel 4.2) bereitgestellt. Um ihn auszuführen müssen zuerst die 3 Ordner vom Fetcher-Output: team, player und year in den Knime-Workflow-Ordner kopiert werden. Anschliessend werden alle Nodes über den Ausführungs-Button ausgeführt.

Node-Table
Das Ziel für den Node-Table ist jeweils ein NHL-Spieler je Datensatz zu erzeugen.
In einem ersten Schritt werden die Spielerdaten über die einzelnen JSON-Files mittels Loop eingelesen. Als Ergebnis erhalten wir eine Tabelle, welche pro Datensatz den jeweiligen JSON-Inhalt eines Spielers beinhaltet (Abbildung 1).

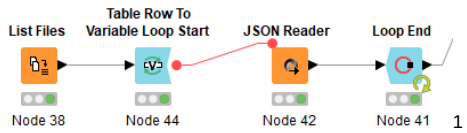

Abbildung 1: Spielerdaten einlesen

[1]

Mit dem JSON-Path-Node werden diese Inhalte dann zu einzelnen verwertbaren Columns bzw. Attributen extrahiert und mit geeigneten Datentypen versehen. Mit dem Column Filter werden unnötige Attribute wieder entfernt, um schliesslich den fertigen Node-Table in ein CSV-File zu speichern (Abbildung 2). Das CSV-File wird UTF-8 encodiert und im Workflow-Ordner als „nodetable.csv“ abgespeichert.

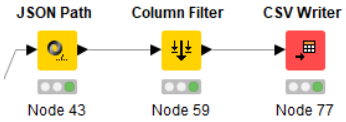

Abbildung 2: Spielerdaten in Attribute speichern

Edge-Table

Mit dem Edge-Table sollen die Spieler so miteinander in Beziehung gestellt werden, dass diese die Information enthalten, wer mit wem zur selben Zeit im selben Team gespielt hat.

Auch hier werden die Teams jeweils über die einzelnen JSON-Files mit einem Loop eingelesen. Da in den jeweiligen Inhalten der JSON-Files keine Informationen zu Team und Season vorhanden sind, müssen diese in einem zusätzlichen separaten Strang (Abbildung 3, Strang oben) extrahiert werden. Die resultierenden Attribute werden in einer Tabelle als Columns gespeichert.
Mit dem unteren Strang bzw. dem JSON-Reader werden gleichzeitig die Inhalte jedes JSON-Files ausgelesen und ebenfalls als Column in einer Tabelle festgelegt.
Die Ergebnisse beider Stränge werden mit dem Cross Joiner vereint, sodass schliesslich der Loop wieder geschlossen werden kann.
Das Ergebnis ist eine Tabelle, die je Datensatz genau die Information enthält, welche Spieler in welcher Team- Seasonkombination gespielt haben.

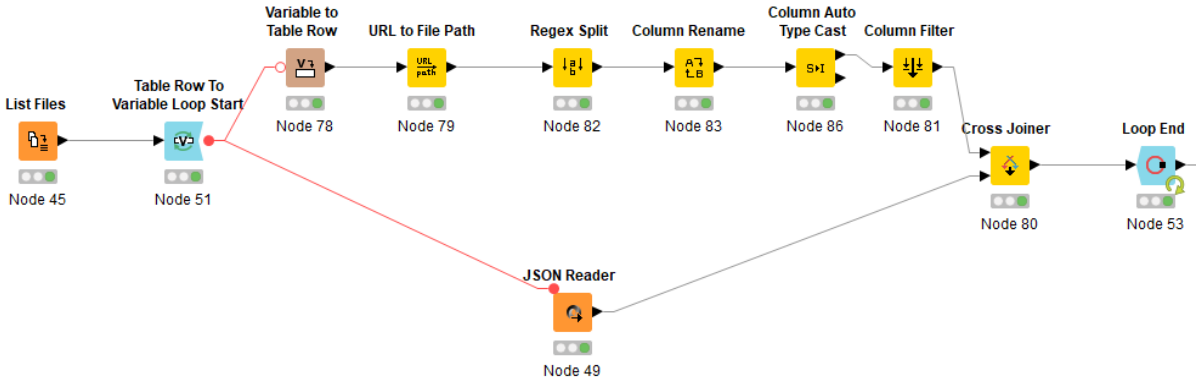

Abbildung 3: Einlesen von Teamdaten

Im nächsten Schritt wird die zuvor erstellte JSON-Column mit dem Ungroup-Node so zerlegt, dass jeweils nur noch ein Spieler je Datensatz erscheint. Weiter wird mit dem JSON-Path-Node die Spieler_ID extrahiert und als Integer gespeichert (Abbildung 4).

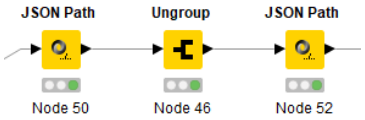

Abbildung 4: Spieler_ID aus Teamdaten extrahieren

Folgender Loop (Abbildung 5) hat schliesslich zum Ziel, zwischen all denen Spielern, welche je Datensatz die gleiche Team- Seasonkombination aufweisen, eine Beziehung herzustellen. Dies wird erreicht indem jede Team-Seasonkombination isoliert im Loop bearbeitet wird.

Jeder Spieler wird vorbereitend mit einer iterativen Nummer versehen. Mit dem Cross Joiner wird jeder Datensatz mit jedem kombiniert (Kartesisches Produkt). Die Rule-based Row Filter entfernen Beziehungen, die auf sich selbst zeigen sowie solche, welche bereits in die eine Kantenrichtung gelistet wurden. Damit werden die Beziehungen so reduziert, dass das Ergebnis dem gesuchten Edge-Table entspricht.

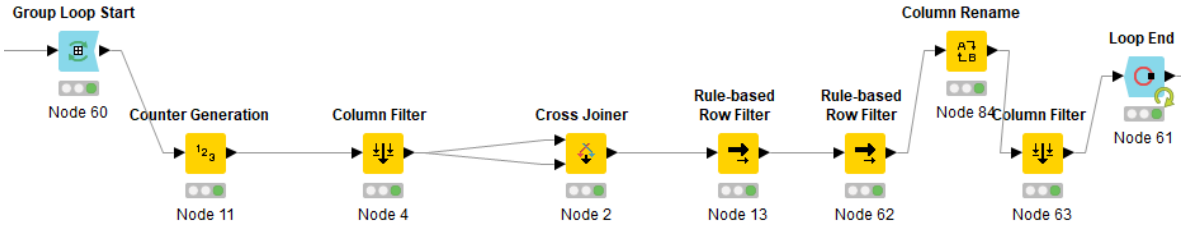

Abbildung 5: Beziehungen zwischen Spielern herstellen

Um den Edge-Table später in Gephi besser Filtern zu können werden noch zusätzliche Team-Informationen als Attribute angefügt. Die Vorgehensweise erfolgt dabei wieder nach dem gleichen Prinzip wie beim Einlesen der Spieler bzw. Teams. Die Ergebnistabelle wird schliesslich mittels Joiner-Node an den Edge-Table geknüpft (Abbildung 6). Das CSV-File wird UTF-8 encodiert und im Workflow-Ordner als „edgetable.csv“ abgespeichert.

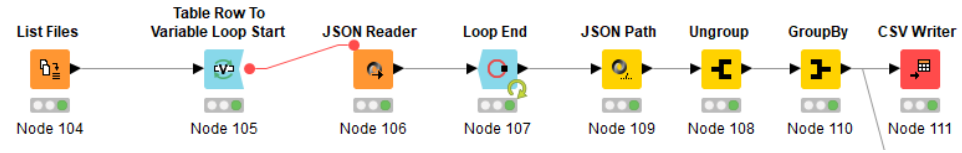

Abbildung 6: Zusätzliche Team Attribute erstellen

3 Daten-Analyse

3.1 Vorgehen

Die Datenanalyse wurde im selben KNIME-Workflow ausgeführt, wie zuvor die Datentransformation. Dies hat den Vorteil, dass durch das Starten des Transformations-Workflow auch gleich die Nodes für die Statistiken und Analyse ausgeführt werden. Somit kann die Datenanalyse jederzeit wieder automatisch durchgeführt werden, unter anderem dann, wenn sich die Daten noch verändern sollten.

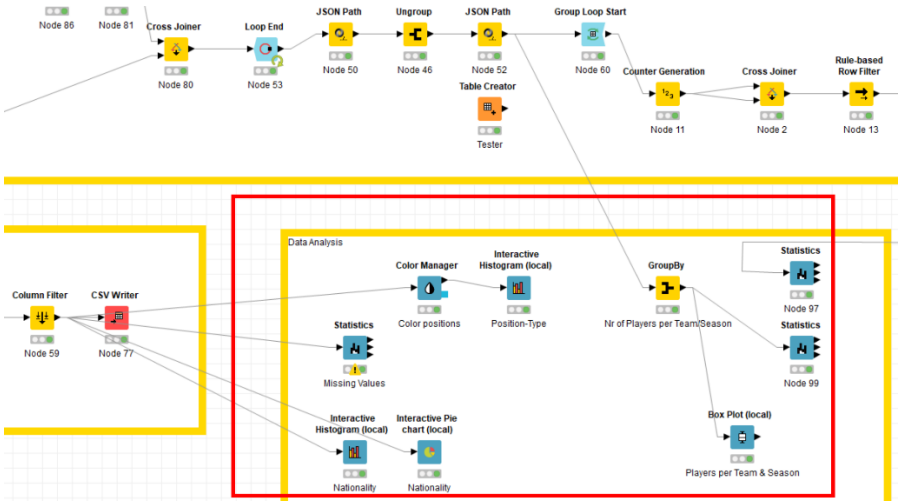

Abbildung 1: Im roten Rahmen die Nodes zur Datenanalyse mit Daten-Inputs aus den Transformationsschritten

3.2 Fehlende Werte

In KNIME wurde über den «Statistics»-Node die «Nominal Histogram Table» berechnet, welche unter anderem Auskunft über fehlende Werte enthält.
Bei der Nationalität wurde ersichtlich, dass lediglich für zwei Spieler keine erfasst wurde. Dies ist ein sehr kleiner Wert, womit diese Information trotzdem für unsere Analysen verwendet werden kann.

Anders sieht es bei der (Haupt-)Trikotnummer der Spieler aus («primaryNumber»). Diese Information wurde von uns nicht explizit für eine Analyse vorgesehen, wir wollten diesen Werte jedoch trotzdem einmal beibehalten. Das Fehlen von 61 solchen Werten hat uns jedoch dazu bewogen, diese Information bei der Transformation zu ignorieren. Glücklicherweise fehlen bei den restlichen Spalten keine Werte, womit wir u.a. auch die Position eines Spielers für die Analysen verwenden können.

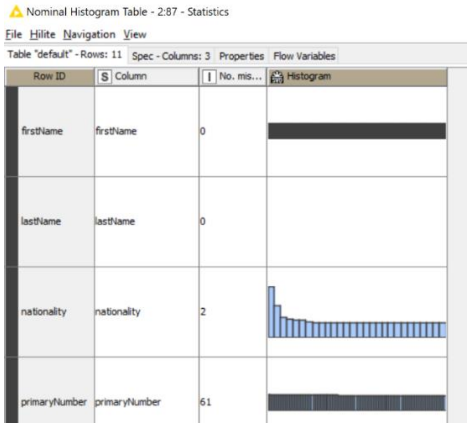

Abbildung 2: Histogramm-Tabelle der Spielerinformationen aus dem KNIME-Workflow

3.3 Realistische Datenmenge

Um herauszufinden, ob die Datenmenge realistisch ist, wurden zuerst die Schätzungen aus dem Meilenstein-Dokument hinzugezogen. Bei der Anzahl Knoten ist schon eine erhebliche Differenz auszumachen (geschätzt: 4'500, effektiv: 2’918). Einen Teil der Differenz lässt sich dadurch erklären, dass wir effektiv nur für 14 Saisons Daten sammeln konnten und nicht, wie bei der Berechnung im Meilenstein angedacht, 15. Weil wir zusätzlich auch die durchschnittliche Anzahl Spieler im Kader überschätzt haben, wurde die Knotenanzahl zur Plausibilisierung mit einer Statistikseite der NHL abgeglichen.
Für die Feldspieler (Verteidiger und Stürmer) wurde der Bericht «Skater Summary» ausgewählt (Link). Dies ergab aufsummiert 2’640 Einträge für die Feldspieler. Um die Anzahl der Torhüter herauszufinden, wurde der Bericht «Goalie Summary» ausgewählt, mit welchem weitere 285 Einträge geladen wurden (Link). Werden diese Zahlen addiert, kommt man auf ein Total von 2'925. Dieser Wert entspricht fast genau unserer Anzahl Knoten. Die minime Differenz dabei ist zu vernachlässigen und liegt sehr wahrscheinlich daran, dass die Berichte jeweils nur die «Regular Seasons» ohne Playoffs berücksichtigen.

Die Anzahl Kanten wurden im Meilenstein etwas genauer geschätzt. Jedoch basierte die Schätzung auf der geschätzten Anzahl Knoten, welche eben nicht der Realität entsprach. Teilt man die Anzahl Kanten (253’217) durch die Anzahl Knoten, so kommt man zum Schluss, dass jeder Spieler durchschnittlich mit knapp 87 anderen Spieler zusammengespielt hat. Bei einer durchschnittlichen Kadergrösse von 34.8 scheint diese Zahl immer noch realistisch.

3.4 Realistische Daten

3.4.1 Nationalität der Spieler

Um die Verteilung der Nationalitäten der Spieler grafisch darzustellen, wurde in KNIME der «Interactive Histogram»-Node verwendet. Daraus wurde ersichtlich, dass Kanada mit Abstand die meisten Spieler stellt, gefolgt von den USA. Da es sich bei der NHL um eine Nordamerikanische Liga handelt, und Kanada als «Mutterland» des Eishockeys gilt, scheinen diese Daten plausibel.

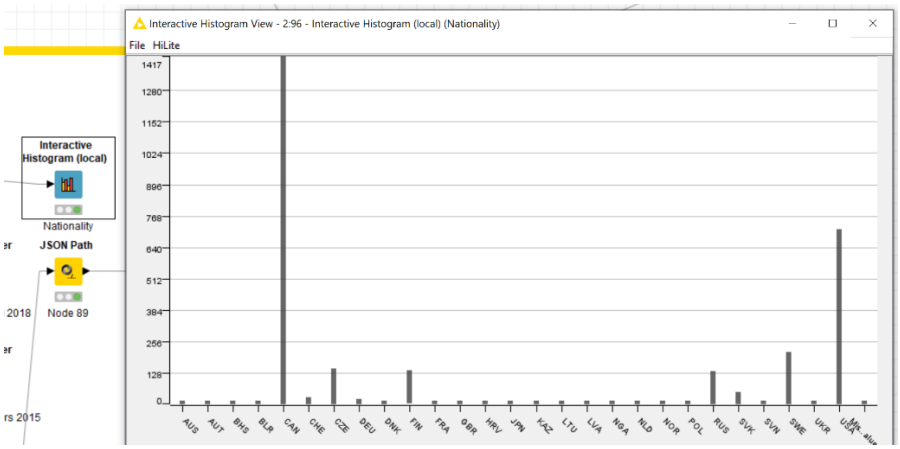

Abbildung 3: Histogramm der Nationalitäten der Spieler mit Kanada als absoluten Spitzenreiter

Vergleicht man unsere Werte mit der relativen Anzahl aus der Saison 2018/2019 (Quelle: quanthockey.com) wird ersichtlich, dass die Anteile für die Nationalitäten sehr eng beieinander liegen. Tendenziell hat Kanada über die Dauer der letzten 14 Jahre einen etwas höheren Anteil, als in aktuellen Saisons, da es eine Tendenz gibt, dass immer mehr ausländische Spieler in der NHL aktiv sind (Quelle: Tweet von eliteprospects).

3.4.2 Anzahl Spieler pro Team und Saison

Die Anzahl Spieler pro Team und Saison wurde mit einem Box Plot ausgewertet. Um diese Zahlen interpretieren zu können, ist es wichtig, zuerst ein paar Regeln zur Kadergrösse eines NHL Teams zu kennen. Zu Saisonbeginn ist der Kader auf maximal 23 Spieler begrenzt, wobei verletzte Spieler nicht dazuzählen. Das absolute Minimum sind 20 Spieler. Während der Saison bleibt das Limit von 23 Spielern im aktiven Kader bestehen, jedoch können Spieler vom aktiven Kader mit Spielern von der Reserven-Liste ausgetauscht werden (z.B. bei Verletzungen, etc.). Insgesamt dürfen aber nicht mehr als 50 Spieler bei einem Team unter Vertrag stehen.
Im Box Plot wird ersichtlich, dass sich sämtliche Werte in diesem Bereich zwischen 23 und 50 Spieler bewegen. Dies lässt unsere Daten schon ziemlich Plausibel erscheinen.

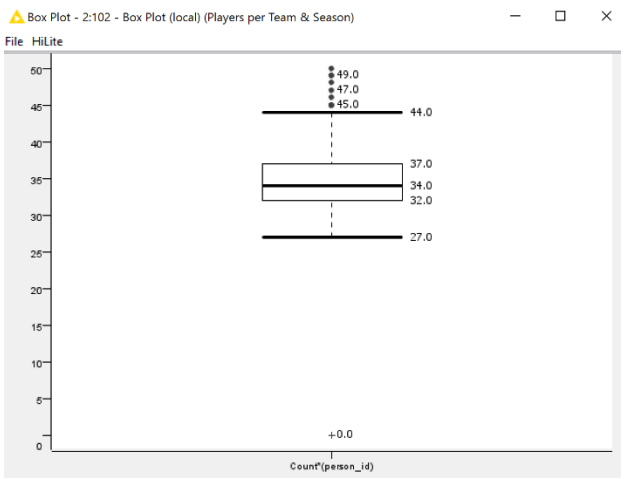

Abbildung 4: Box Plot der Anzahl Spieler pro Team und Saison

Mit der «HiLite»-Funktion wurden die sechs nach oben ausreissenden Werte markiert und damit in der tabellarischen Darstellung graphisch hervorgehoben. Daraus wurde ersichtlich, zu welcher Kombination aus Team und Saison die Ausreisser gehören. Für diese Ausreisser wurde nochmals ein Vergleich mit der Anzahl der Spieler auf der jeweiligen Kaderwebsite gemacht (siehe Kapitel 1), welcher diese grossen Zahlen bestätigte.

4 Anhang

4.1 Excel-Dokument für Rohdatenanalyse

Das hier verlinkte Excel-Dokument wurde verwendet, um die Resultate der API-Abfragen mit der NHL-Webseite zu vergleichen:

Datenanalyse.xlsx

4.2 KNIME Workflow

Der KNIME Workflow zur Datentransformation und -analyse ist einerseits im Sourcecode im Unterordner «Knime» zu finden, da wir ihn auch in die Quellcodeverwaltung aufgenommen haben. Zusätzlich ist er aber hier als Anhang verlinkt:

SNA_Projekt.knwf