

Game of Thrones - Warum Machtspiele so brutal sind

Lea Mayer

Oct 21. 2021



Einleitung in die Thematik

Kaum einer hat seit dem Serienstart 2011 noch nicht von **Game of Thrones** gehört. Für die Wenigen, die noch nichts von den sieben Königsländern in Westeros mitbekommen haben, handelt es sich dabei um eine amerikanische Fantasy-Fernsehserie, nach den Büchern von George R. R. Martin. Die Serie handelt von den Machtkämpfen um den eisernen Thron, weiteren Bedrohungen, Morden, Intrigen und Verrat.

Ohne an dieser Stelle zukünftige Fans zu spoilern, hebt sich die Serie vor allem durch ihre Brutalität, Unverblümtheit und dem Tod des ein oder anderen Hauptcharakters von anderen ab. In diesen durchweg brutalen Zuständen wird auch der ein oder andere vermeindliche Held zum Mörder. Doch was bringt ihn dazu? Sind die Beziehungen zu anderen Charakteren und die eigene Position ausschlaggebend für dieses Verhalten?

Im Folgenden soll also der Zusammenhang zwischen der Bereitschaft zu töten und der Interaktionen der einzelnen Charaktere, sowie dessen Position zwischen den anderen Charakteren wissenschaftlich untersucht werden. Dazu wird eine Social Network Analyse durchgeführt, diese erfasst Beziehungen und Beziehungsstrukturen, bildet diese durch Knoten und Kanten ab, die anschließend grafisch visualisiert werden können. Diese Visualisierung und Analyse der Daten soll helfen die Fragestellung zu beantworten und Zusammenhänge hervorheben.

Benötigte Packages laden

Zunächst müssen alle Packeges installiert werden, die im Folgenden für die Social Network Analyse benötigt werden.

```
# Output nicht anzeigen
# mit "message = FALSE" kann das printen des Packageladens verhindert werden

# Benötigte Packages:
# Alle Packeges werden nur einmal Installiert und dann auskommentiert

# install.packages("knitr", dependencies = TRUE) # für schöne Tabellen
# install.packages("tidyverse", dependencies = TRUE) # falls ein Paket noch nicht installiert ist
# install.packages("gapminder", dependencies = TRUE) # für die gapminder Daten
# install.packages("magick", dependencies = TRUE) # für die gapminder Daten
# install.packages("ggthemes", dependencies = TRUE) # Themes für Datenvisualisierung
# install.packages("ggridge", dependencies = TRUE) # dynamisches Animations Package
# install.packages("gifski", dependencies = TRUE) # Maschine um Gifs zu rendern
# install.packages("gridExtra", dependencies = TRUE) #
# install.packages("plotly", dependencies = TRUE) #

library("tidyverse") # Laden von tidyverse um damit in diesem 'Environment' zu programmieren
```

Daten importieren

Über Game of Thrones existiert eine Vielzahl an Daten. Für die Analyse werden Daten über die Charaktere und deren Beziehung untereinander, sowie Informationen über andere Charaktere, die von diesem Charakter getötet wurden, benötigt. Bei den Beziehungen soll es sich nicht um Daten wie Freundschaften oder Verwandschaft handeln, sondern um die Interaktionen zwischen den Charakteren ungeachtet deren Motive. Damit können auch negative Beziehungen abgebildet werden, um die Stellung eines Charakters im sozialen Netzwerk zu analysieren.

Besonders gut eignet sich dafür die Beziehungsliste, die von einem User über Github bereitgestellt wurde (data-game-of-thrones-nodes&edges). Eine Beziehungsliste stellt die Beziehung zwischen den Knoten spaltenweise dar und zählt beispielsweise deren Häufigkeit. Der User hat aus verschiedenen, von Fans erstellten Drehbuchskripte der Serie herausgelesen, wann und zwischen wem wie oft eine Interaktion stattfindet und diese in der Beziehungsliste gespeichert.

Da diese Daten, durch das Analysieren der Drehbuchskripte, extra für den Zweck einer Analyse herausgelesen wurde, handelt es sich um custom-made Daten. Es handelt sich dabei um eine Gesamterhebung, da hier nicht auf einen Ausgangspunkt fokussiert wird. Ebenfalls handelt es sich nicht um ein Schneeball Netzwerk, da die Charaktere nicht nacheinander abgearbeitet werden und auf sich gegenseitig verweisen, sondern die Beziehungen beidseitig immer wieder zeitlich versetzt auftreten können. Namensgeneratoren sind dabei die Charaktere, die in den Skripten genannt werden. Mit einer Nennung hat ein Charakter auch automatisch eine Beziehung/Interaktion mit einem weiteren Charakter und wird damit ins Netz aufgenommen. Bei den Namensinterpretatoren handelt es sich um die Beziehungen der einzelnen Knoten. Diese werden dann über den Alogrithmus des Git-Users aus den Skripten ermittelt, wenn zwei Namen von Charakteren miteinander auftauchen. Wie genau sich eine Beziehung definiert und wann diese gezählt wird, wird später genauer beschrieben, wenn auf die Kanten des Netzes eingegangen wird.

Adressiert werden die Charaktere über eine individuelle ID. Der User stellt neben der Beziehungsliste ebenfalls eine Liste der Charaktere der Serie bereit. Diese wird im Folgenden verwendet, da die identische ID-Bezeichnung die Verbindung der Datensätze, zum erstellen eines Netzes, erleichtert. Allerdings müssen diese Daten noch durch einen anderen Datensatz, um Informationen über die Anzahl getöteter Charaktere

ergänzt werden. Dazu werden Daten verwendet, die ein anderer User über Github zu Verfügung stellt (data-game-of-thrones-deaths). Hierbei handelt es sich ebenfalls um custom-made Daten, da sie extra für diesen Zweck durch eine Gesamterhebung ermittelt wurden.

Mit dem Befehl `read_csv()` werden die Daten aus dem Data-Ordner mit relativen Pfaden importiert. Die relativen Pfade sorgen dafür, dass jeder Rechner auf die Daten zugreifen kann.

```
library('tidyverse') # basic data wrangling
library('tidygraph') # SNA
library('tidylog') # Verbose tidy code
library('ggraph') # Network Data Viz

# read_csv() importiert Nodes und Edges aus csv files:

df_deaths <- read_csv("data/game-of-thrones-deaths-data.csv") # erstellt aus geladener Datei ein Objekt

edgesS1 <- read_csv("data/got-s1-edges.csv")
edgesS2 <- read_csv("data/got-s2-edges.csv")
edgesS3 <- read_csv("data/got-s3-edges.csv")
edgesS4 <- read_csv("data/got-s4-edges.csv")
edgesS5 <- read_csv("data/got-s5-edges.csv")
edgesS6 <- read_csv("data/got-s6-edges.csv")
edgesS7 <- read_csv("data/got-s7-edges.csv")
edgesS8 <- read_csv("data/got-s8-edges.csv")
nodesS1 <- read_csv("data/got-s1-nodes.csv")
nodesS2 <- read_csv("data/got-s2-nodes.csv")
nodesS3 <- read_csv("data/got-s3-nodes.csv")
nodesS4 <- read_csv("data/got-s4-nodes.csv")
nodesS5 <- read_csv("data/got-s5-nodes.csv")
nodesS6 <- read_csv("data/got-s6-nodes.csv")
nodesS7 <- read_csv("data/got-s7-nodes.csv")
nodesS8 <- read_csv("data/got-s8-nodes.csv")
```

First Look auf die Daten

Um die Datensätze verbinden und analysieren zu können, müssen die Daten zuerst inspiziert werden.

```
# Aufbau der Daten analysieren um Cleaning-Bedarf festzulegen

df_deaths

## # A tibble: 6,887 x 11
##   order season episode character_killed killer    method  method_cat reason
##   <dbl>  <dbl>   <dbl> <chr>           <chr>    <chr>   <chr>      <chr>
## 1     1      1       1 Waymar Royce    White W~ Ice sw~ Blade   Unknown
## 2     2      2       1 Gared          White W~ Ice sw~ Blade   Unknown
## 3     3      3       1 Will           Ned Sta~ Sword ~ Blade   Deserting ~
## 4     4      4       1 Stag           Direwolf Direwo~ Animal  Unknown
## 5     5      5       1 Direwolf        Stag     Antler  Animal  Unknown
## 6     6      6       1 Jon Arryn       Lysa Ar~ Poison Poison  Petyr Bael~
## 7     7      7       1 Dothraki man  Dothrak~ Arakh  Blade   A Dothraki~
## 8     8      8       1 Catspaw assassin Summer  Direwo~ Animal  Attempting~
## 9     9      9       1 Mycrah         Sandor ~ Unknow~ Unknown Joffrey ha~
```

```

## 10      10      1      2 Lady          Ned Sta~ Knife     Blade      Robert Bar~
## # ... with 6,877 more rows, and 3 more variables: location <chr>,
## #   allegiance <chr>, importance <dbl>

```

edgesS1

```

## # A tibble: 549 x 4
##   Source      Target Weight Season
##   <chr>       <chr>   <dbl>  <dbl>
## 1 NED        ROBERT    192     1
## 2 DAENERYS   JORAH     154     1
## 3 JON         SAM       121     1
## 4 LITTLEFINGER NED     107     1
## 5 NED        VARYS     96      1
## 6 DAENERYS   DROGO     91      1
## 7 ARYA       NED       90      1
## 8 CATELYN    ROBB      90      1
## 9 BRONN      TYRION    86      1
## 10 CERSEI     NED      86      1
## # ... with 539 more rows

```

nodesS1

```

## # A tibble: 126 x 2
##   Id      Label
##   <chr>    <chr>
## 1 ADDAM_MARBRAND Addam
## 2 AEGON      Aegon
## 3 AERYS      Aerys
## 4 ALLISER_THORNE Allister
## 5 ARYA       Arya
## 6 ASSASSIN   Assassin
## 7 BAELOR     Baelor
## 8 BALON      Balon
## 9 BARRISTAN  Barristan
## 10 BENJEN    Benjen
## # ... with 116 more rows

```

#edgesS2

#nodesS2

#Inspektion der benötigten Spalten

#Überprüfen auf na_werte bei killern
`#sum(is.na(df_deaths$killer))`

#Überprüfen ob jeder Name eine ID hat
`#sum(is.na(nodesS1$Label))`
`#sum(is.na(nodesS1$ID))`

#Überprüfen ob Kanten ins leere führen
`#sum(is.na(edgesS1$source))`
`#sum(is.na(edgesS1$target))`

Im Folgenden werden für die Analyse zum Beantworten der Forschungsfrage Informationen über die Charaktere benötigt, wie deren Namen, die Anzahl deren Opfer und die Beziehungen zu den anderen Charakteren. Auf alle anderen Informationen kann verzichtet werden. Da diese Daten keine fehlenden Informationen oder andere Unstimmigkeiten aufweisen, können die benötigten Informationen im nächsten Schritt ohne weiteres Cleaning verbunden werden.

Merching data 1

Aus der Inspizierung geht hervor, dass die Edge und Node Dfs in ihrer Bezeichnung identisch sind, sodass diese einfach über einen Join-Befehl zusammengebracht werden können. Durch das verbinden der einzelnen Staffeln, kann die gesammte Serie analysiert werden. Ebenfalls sind die Daten so ausgewählt, dass sie über die identischen Ids einfach zu einem Netzwerk hinzugefügt werden können.

```
### Wie kann ich nur den code anzeigen aber nicht den Output
df_nodes = nodesS1 %>% # zusammenfassen der einzelnen Node dfs zu einem großen df mit join -> ermöglicht
  full_join(nodesS2) %>%
  full_join(nodesS3) %>%
  full_join(nodesS4) %>%
  full_join(nodesS5) %>%
  full_join(nodesS6) %>%
  full_join(nodesS7) %>%
  full_join(nodesS8)

df_edges = edgesS1 %>% # zusammenfassen der einzelnen Edge dfs zu einem großen df mit join
  full_join(edgesS2) %>%
  full_join(edgesS3) %>%
  full_join(edgesS4) %>%
  full_join(edgesS5) %>%
  full_join(edgesS6) %>%
  full_join(edgesS7) %>%
  full_join(edgesS8)
```

Merching data 2

Die Nodes enthalten nun alle Namen Charaktere der 8 Staffeln und eine ID, mit der die Charaktere in der Egdelist bezeichnet werden. Für die Analyse müssen die Knoten allerdings darüber hinaus noch die informationen enthalten wie viele Charaktere von dem entsprechenden Charakter im laufe der 8 Staffeln getötet worden sind. Aus den Einblick geht ebenfalls hervor, dass die Datensätze aus den unterschiedlichen Github-Datensätzen nicht über eine identische Spalte verbunden werden können. Beispielsweise wird Arya in den Nodes als Arya bezeichnet und in dem Datensatz dessen Informationen den einzelnen Charakteren zugeordnet werden sollen als Arya Stark. Um die Datensätze über eine identische bezeichnung zu verbinden wird die Spalte des Namens in vor und Nachname aufgeteilt. Über den Vornamen eines Charakters wird dann die berechnete Rate der getöteten Charaktere dem Knoten zugeordnet.

```
# Anpassung des Labels, damit die Daten über eine identische Spalte zu dem entsprechenden Knoten zugeordnet werden
library(tidyr)

df_deaths <- df_deaths %>% separate(killer, c('Label', 'Surname')) #Aufteilung der Spalte killer in Label und Surname

#Zählen der getöteten Charaktere
```

```

killrate <- df_deaths %>%
  count(Label) # Morde mit dem gleichen Label (Vorname des Killer) werden zusammengefasst und deren Anzahl
  killrate # Überprüfung der Ergebnisse

## # A tibble: 129 x 2
##   Label      n
##   <chr>    <int>
## 1 Accident     1
## 2 Allister     3
## 3 Amory       1
## 4 Arryn      12
## 5 Arthur       3
## 6 Arya      1278
## 7 Baelish      1
## 8 Baratheon    56
## 9 Barristan    16
## 10 Benjen      23
## # ... with 119 more rows

colnames(killrate) <- c("Label", "kills") # Umbenennung der Spalten
df_nodes_wk <- merge(df_nodes, killrate, by="Label") # Killrate wird über identische Labelspalte zu dem entsprechenden NodeID hinzugefügt

df_nodes_wk # Überprüfung der Ergebnisse

```

	Label	Id	kills
## 1	Allister	ALLISER_THORNE	3
## 2	Amory	AMORY	1
## 3	Arthur	ARTHUR	3
## 4	Arya	ARYA	1278
## 5	Barristan	BARRISTAN	16
## 6	Benjen	BENJEN	23
## 7	Beric	BERIC	30
## 8	Brienne	BRIENNE	28
## 9	Bronn	BRONN	27
## 10	Catelyn	CATELYN	1
## 11	Cersei	CERSEI	199
## 12	Daario	DAARIO	25
## 13	Daenerys	DAENERYS	24
## 14	Dickon	DICKON	1
## 15	Eddison	EDDISON_TOLLETT	3
## 16	Ellaria	ELLARIA	2
## 17	Euron	EURON	12
## 18	Gendry	GENDRY	9
## 19	Gregor	MOUNTAIN	10
## 20	Grenn	GRENN	2
## 21	Hodor	HODOR	4
## 22	Ilyn	ILYN_PAYNE	1
## 23	Jaime	JAIME	23
## 24	Janos	JANOS	1
## 25	Jaqen	JAQEN	4
## 26	Jeor	JEOR	1
## 27	Joffrey	JOFFREY	1

```

## 28      Jon        JON  116
## 29    Jorah      JORAH   65
## 30     Jory  JORY_CASSEL    3
## 31     Karl KARL_TANNER   2
## 32    Karsi      KARSI   9
## 33     Leaf      LEAF  37
## 34     Locke     LOCKE   2
## 35    Loras      LORAS   4
## 36   Lothar     LOTHAR   2
## 37   Lyanna     LYANNA   1
## 38     Lysa      LYSA   1
## 39   Magnar     MAGNAR   3
## 40    Meera     MEERA   6
## 41 Melisandre MELISANDRE  8
## 42    Meryn  MERYN_TRANT  1
## 43  Mossador MOSSADOR   1
## 44     Ned       NED   5
## 45  Nymeria    NYMERIA   1
## 46    Obara     OBARA   2
## 47   Olenna     OLENNNA   1
## 48     Olly     OLLY   1
## 49     Osha     OSHA   2
## 50    Petyr  LITTLEFINGER  1
## 51   Podrick    PODRICK   5
## 52   Polliver  POLLIVER   1
## 53    Qhorin    QHORIN   1
## 54    Qhorin  QHORIN_HALFHAND  1
## 55    Qyburn    QYBURN   2
## 56    Ramsay    RAMSAY  15
## 57     Rast      RAST   1
## 58     Robb      ROBB   3
## 59   Rodrik     RODRIK   2
## 60    Roose  ROOSE_BOLTON   1
## 61     Sally     SALLY   1
## 62     Sam       SAM   7
## 63    Sandor    HOUND  59
## 64     Sansa    SANSA   1
## 65    Selyse    SELYSE   1
## 66   Stannis   STANNIS   9
## 67     Styr      STYR   8
## 68    Theon     THEON  34
## 69   Tommen    TOMMEN   1
## 70   Tormund   TORMUND  66
## 71     Tyene    TYENE   5
## 72   Tyrion     TYRION   5
## 73     Yara      YARA  12
## 74   Ygritte   YGRITTE  14
## 75     Yoren     YOREN   5

```

```
df_edges
```

```

## # A tibble: 4,110 x 4
##   Source      Target  Weight Season
##   <chr>      <chr>    <dbl>  <dbl>

```

```

## 1 NED ROBERT 192 1
## 2 DAENERYS JORAH 154 1
## 3 JON SAM 121 1
## 4 LITTLEFINGER NED 107 1
## 5 NED VARYS 96 1
## 6 DAENERYS DROGO 91 1
## 7 ARYA NED 90 1
## 8 CATELYN ROBB 90 1
## 9 BRONN TYRION 86 1
## 10 CERSEI NED 86 1
## # ... with 4,100 more rows

```

Netz erstellen

Um aus diesen Daten ein soziales Netzwerk zu erstellen müssen als erstes die Akteure, die sogenannten Knoten des Netzes, festgelegt werden. Die Akteure sind in den Game of Thrones Daten die Charaktere wie Jon, Daenerys, Arya und Cersei. Diese sozialen Akteure haben verschiedene Merkmale wie ihren Namen oder Anzahl der von ihnen getöteten Personen. Die Beziehungen der Akteure, die sogenannten Kanten, verbinden diese. Die Kanten gehen aus den Daten hervor wie oft zwei Akteure miteinander interagiert haben. Eine Interaktion wurde verzeichnet wenn zwei Charaktere direkt miteinander sprechen, übereinander sprechen, kurz hintereinander genannt werden oder in der selben Szene auftauchen. Damit ist die Beziehung immer auf gegenseitigkeit bezogen und hat damit keine Richtung, sie gilt also als ungerichtet. Die Kante gibt also wieder wie oft die Knoten miteinander interagiert haben. Die Beziehungen werden durch die Kanten also vergleichend in ihrer Häufigkeit bewertet, damit wird die Kante als gewichtet bezeichnet.

Sind Kanten und Knoten nun definiert, kann das Netz erstellt werden.

```

net <- as_tbl_graph(df_edges) #Erstellt einen Table-Graph (Netzwerk) mit dem Namen net, dem Kanten zugehörigen Werten und den Knoten als Nodes
net <- net %>% left_join(df_nodes_wk %>% rename(name = Id), by = "name") #dem Netz werden ebenfalls die Knotenmerkmale hinzugefügt

```

Visualisierung des Netzes

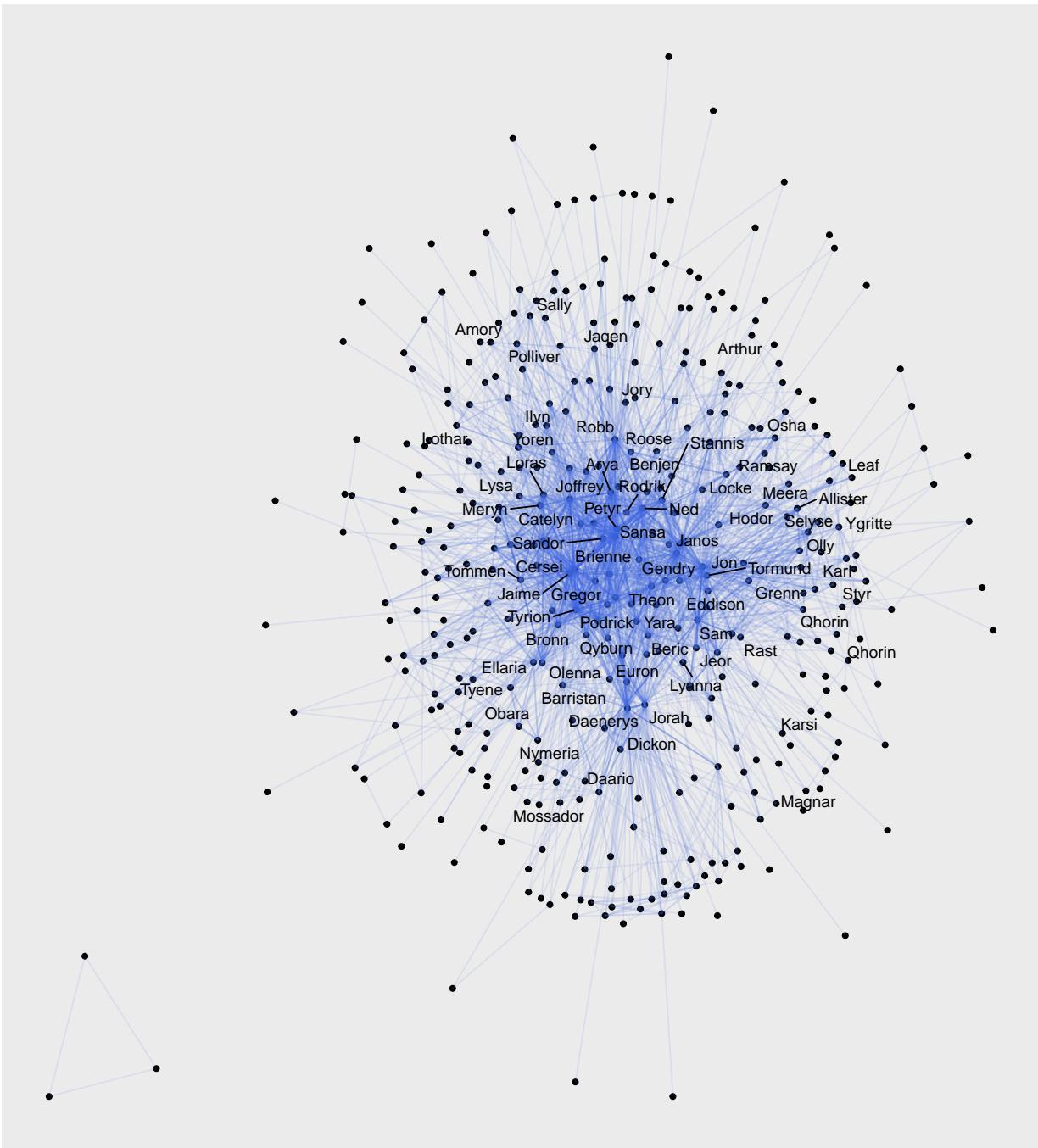
Das Netz, bestehend aus Kanten und Konten kann nun visualisiert werden. Anschließend wird die Visualisierung so gestaltet, dass die Attribute, wie in diesem Fall die getöteten Charaktere, ebenfalls abgebildet. Dazu sollen die Knoten sich mit steigender Anzahl vergrößern.

```

# ChunkOutput mittig ausgeben & große Plots
#Netzwerk nur mit Verbindungen plotten

#Netzwerk plotrten
net %>% # erste Visualisierung des Netzwerks durch ggraph
  ggraph( layout = "stress") +
  geom_node_point() +
  geom_edge_link( alpha = 0.1, edge_colour = "royalblue") +
  geom_node_text(aes(label = Label), repel = TRUE,
                 point.padding = unit(0.1, "lines"))

```



#Netzwerk mit Killattribut darstellen

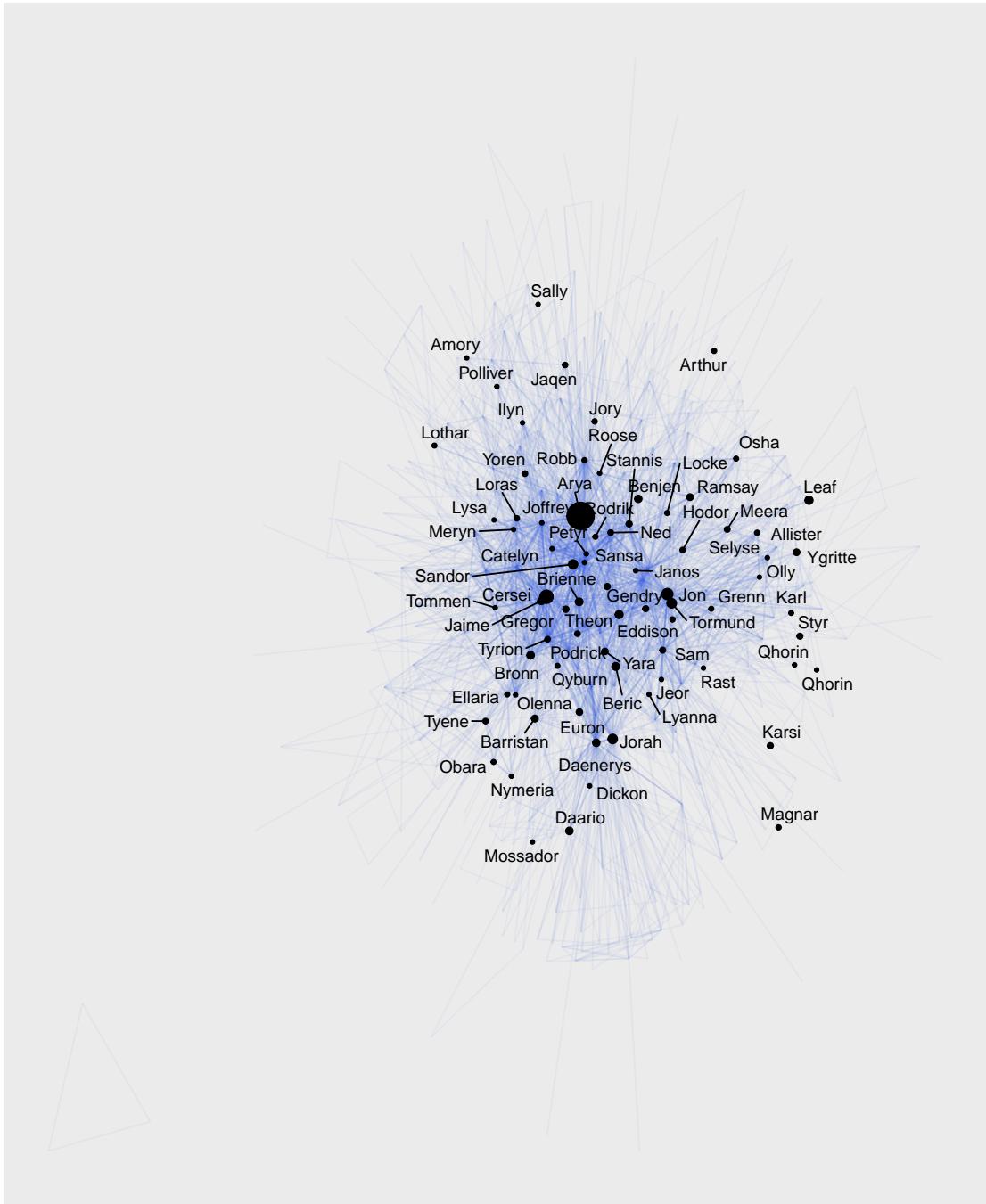
```
library(plotly)

net %>% # erste Visualisierung des Netzwerks durch ggraph
  ggraph( layout = "stress") +
  geom_edge_link( alpha = 0.05, edge_colour = "royalblue") +
  geom_node_point(
    aes(size = kills)) +
```

```

  scale_size(range = c(1,8)) + # currently under construction - send help
  geom_node_text(aes(label = Label), repel = T,
    point.padding = unit(0.1, "lines"))

```



```

#plot %>% ggplotly(tooltip="Label") # %>% toWebGL() -> funktioniert nicht mit ggraph, da geom_GeomEdgeP
# Daher wurde die größe angepasst um möglichst viele Labels zu lesen

```

In den Plots sind nun die Kanten, also die Charaktere aus der Serie Game of Thrones, und deren Interaktionen

untereinander, die Kanten visualisiert. Dabei Rücken Charaktere, die mit vielen anderen interagieren in die Mitte der Darstellung, während Charaktere die nur vereinzelte Konakte haben am Rand eingezeichnet sind. Die Größe der Knoten gibt anschließend Aufschluss darüber, wie viele andere Charaktere von einem Charakter getötet wurden. Der Charakter Arya hat demnach am meisten Opfer, gefolgt von Jon, Cersei und Jaime. Ebenfalls zu beobachten ist, dass sich die Charaktere mit den größten Opferzahlen in der Mitte des Netzes befinden und damit mit den meisten anderen in Kontakt stehen.

Dieses Netz kann nun weiter analysiert werden. Im Sinne der Sozial Kapital Theorie kann Kapital als Ressource wie Macht verstanden werden, zu der man über Beziehungen Zugang hat. Beziehungen können also als Ressource verstanden werden, auch in der Serie wird immer wieder deutlich wie nicht nur materielles eine Rolle spielt, sondern wie wichtig loyale Verbündete sind und wie selten diese mit jeder Staffel werden ... Analysiert man also die Beziehungen der Knoten, können wie Ergebnisse Aufschluss darüber geben wer die Autoritätspersonen sind oder wer die größte Kontrolle hat. Diese Zentralitätsmaße können gemessen und in die Darstellung des Netzes integriert werden. Man unterscheidet zwischen Degree (Wer hat die meisten Kontakte?), Closeness (Wer hat den direktesten Zugang zu anderen Gruppen?), Betweenness (Wer hat die größte Kontrolle?) und Authority (Wer hat die einflussreichsten Freunde?). All diese Aspekte sind im Machtspiel um den Eisernen Thron zentrale Erfolgsfaktoren, doch bringen diese Eigenschaften der Knoten auch die Brutalität mit sich andere zu töten? Genau um dieser Frage nachzugehen werden die Zentralitätsmaße in das Netz integriert um Zusammenhänge darzustellen.

Dazu müssen die Zentralitätsmaße für jeden Knoten berechnet werden. Für diese Berechnungen werden Formeln verwendet, die bestimmte Annahmen über das Netz treffen. Man unterscheidet verschiedene Arten wie Dinge durch das Netz fließen. Bei dem Game of Thrones Netz handelt es sich dabei um Interaktionen zwischen Charakteren. Ein Charakter kann mit verschiedenen Charakteren interagieren. Ein Knoten kann also mehrfach besucht werden. Ebenfalls kann eine Verbindung der Charaktere, also die Kante mehrfach durchlaufen werden, da eine Interaktion wie oben im Abschnitt über die Datenherkunft immer beidseitig ist. Ebenfalls kann eine Interaktion mehrmals erfolgen. Deshalb handelt es sich bei der Flowstruktur um einen Walk, bei dem sowohl Knoten als auch Kanten mehr als einmal besucht werden können. Für ein Netz mit Walk-Flows können mehrere Zentralitätsmaße berechnet werden, dazu zählen Closeness, Degree und Authority. Betweenness kann an dieser Stelle nicht verwendet werden, da die Flows hierbei immer dem kürzesten Pfad folgen müssten, dies ist bei den Interaktionen zwischen den Charakteren nicht der Fall.

Visualisierung der Zentralitätsmaße

Beispielhaft wird im Folgenden der Prozess durchlaufen, wie die Zentralitätsmaße berechnet und als Attribut der Knoten zum Netz hinzugefügt wird. Diese zusätzlichen Daten werden ebenfalls in der Visualisierung dargestellt, indem die Ausprägung des Zentralitätsmaßes über die Färbung der Knoten ablesen lässt.

```
### Wie kann ich nur den code anzeigen aber nicht den Output

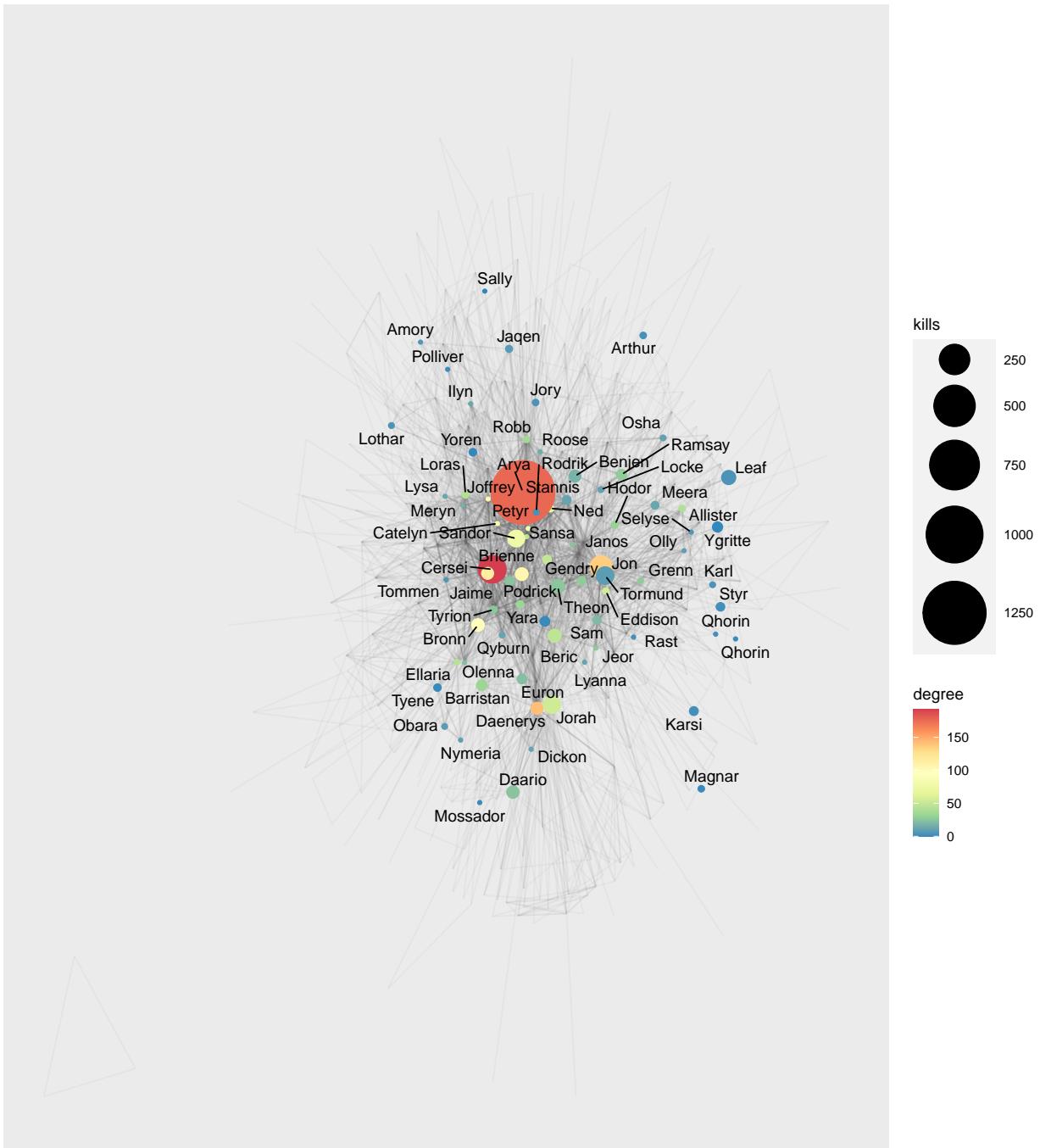
net_degree <- net %>% #Netz wird im Folgenden überarbeitet
  activate(nodes)%>% #Knoten werden aktiviert
  mutate(
    degree = centrality_degree() #Die authority-centrality des Netzwerkes wird für jeden Knoten berechnet
  )

net_degree %E>%
  as.data.frame() %>%
  select(from,to) %>%
  unique

# Beziehungen + Kills + Centrality visualisieren

net_degree_vis <- net_degree %>% # erste Visualisierung des Netzwerks durch ggraph
```

```
ggraph( layout = "stress") +  
  geom_edge_link( alpha = 0.05, edge_colour = "dimgray") +  
  geom_node_point(  
    aes(size = kills, colour = degree)) +  
    scale_size(range = c(1,20)) +  
    scale_colour_distiller(palette = "Spectral") +  
  geom_node_text(aes(label = Label), repel = TRUE,  
    point.padding = unit(0.1, "lines"))  
net_degree_vis
```



```
# nur die Killer und deren Position
```

```
net_degree_vis_red <- net_degree %>% # erste Visualisierung des Netzwerks durch ggraph
ggraph( layout = "stress") +  
  
geom_node_point(  
  aes(size = kills, colour = degree)) +  
  scale_size(range = c(1,20)) +  
  scale_colour_distiller(palette = "Spectral") +
```

```
geom_node_text(aes(label = Label), repel = TRUE,
               point.padding = unit(0.1, "lines"))
```

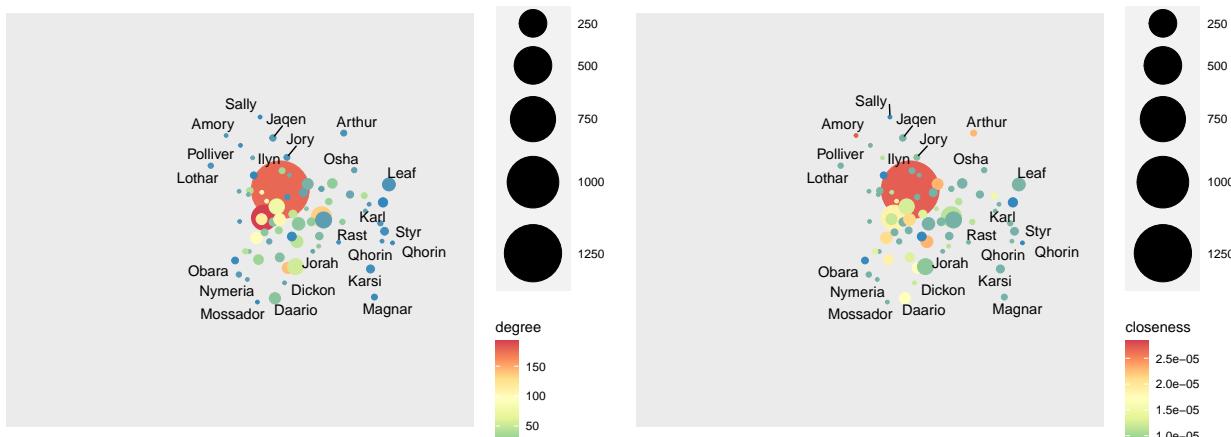
Die Visualisierung stellt nun die zuvor erstellten Beziehungen der Charaktere untereinander da, wie auch die Anzahl der getöteten Charaktere anhand der Größe der Knoten dar. Nun wurde die Farbe der Knoten mit einer Skala so angepasst, dass diese die Ausprägung des Zentralitätsmaßes Degree wiederspiegelt. Eine rote Färbung entspricht einer ausgeprägten Degree-Zentralität und bedeutet für den entsprechenden Knoten, dass dieser im Verhältnis zu den übrigen die meisten Kontakte hat. Die deutlichste rotfärbung ist bei den größten Knoten zu beobachten, was auf einen Zusammenhang zwischen der Anzahl der getöteten Charaktere und der Anzahl der Knoten des Mörders hinweist.

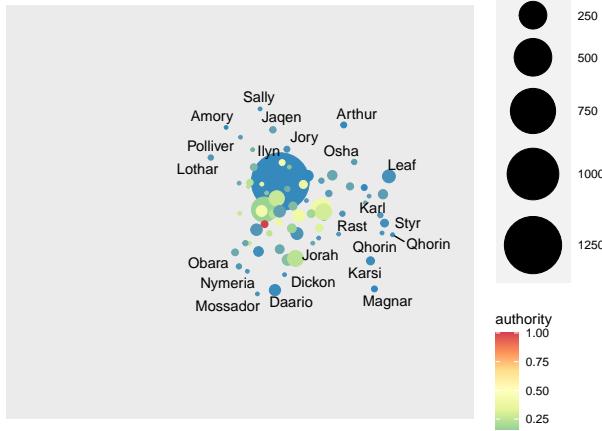
Vergleich der Ausprägung der Zentralitätsmaße

Nun können nach dem gleichen Vorgehen die anderen Zentralitätsmaße berechnet und visualisiert werden. Ausgangspunkt dafür ist immer das gleiche Netzwerk, weshalb auf die Darstellung der Kanten an dieser Stelle verzichtet wird. Anhand der Zentralitätsmaße und der Verteilung in der Visualisierung kann allerdings auf die Interaktionen geschlossen werden. Doch der Fokus soll auf dem Zusammenhang zwischen den getöteten Charakteren und den Zentralitätsmaßen liegen.

```
par(mar = c(4, 4, .1, .1)) #Darstellung von je 2 Plots nebeneinander

net_degree_vis_red
net_closeness_vis_red
#net_betweenness_vis_red
net_authority_vis_red
```





Hat ein Charakter nun die höchste Ausprägung eines Zentralitätsmaßes, ist dieses rot eingefärbt. Wenn ein Zusammenhang zwischen den getöteten Charakteren und dem Zentralitätsmaß bestehen würde, müsste sich die größten Knoten ebenfalls durch eine rote Färbung auszeichnen. Entsprechend müsste die Größe der Knoten identisch mit der Färbung verändern. Ähnlich wie bei der soweit betrachteten Degree-Zentralität lässt sich durch die großen roten Knoten ein Zusammenhang zwischen der Closeness-Zentralität und der Anzahl der getöteten Charaktere vermuten. Das würde bedeuten, dass die Killer mit den meisten Opfern den direktesten Zugang zu ihren Kontrahenten gegenüber anderen haben. Ein Zusammenhang zwischen der Authority-Zentralität und der Anzahl der Opfer ist nicht zu vermuten, da starke Ausprägungen des Zentralitätsmaßes sich nicht unter den größten Knoten befinden. Der Einfluss auf andere Charaktere ist demnach eine auszeichnende Eigenschaft der Killer aus Game of Thrones.

Berechnung der Korrelation

Um diesem Ansatz nun nochmal zu überprüfen und darzustellen, soll im folgenden die Kovarianz zwischen den Attributen der Knoten (Killrate und Zentralitätsmaß) betrachtet werden. Kovarianz ist ein Maß für die Assoziation zwischen zwei Variablen. Je mehr die Kovarianz von 0 (Null) abweicht, umso deutlicher ist die lineare Beziehung zwischen den beiden Variablen. (Quelle)

```
par(mar = c(4, 4, .1, .1)) #Darstellung von je 2 Plots nebeneinander

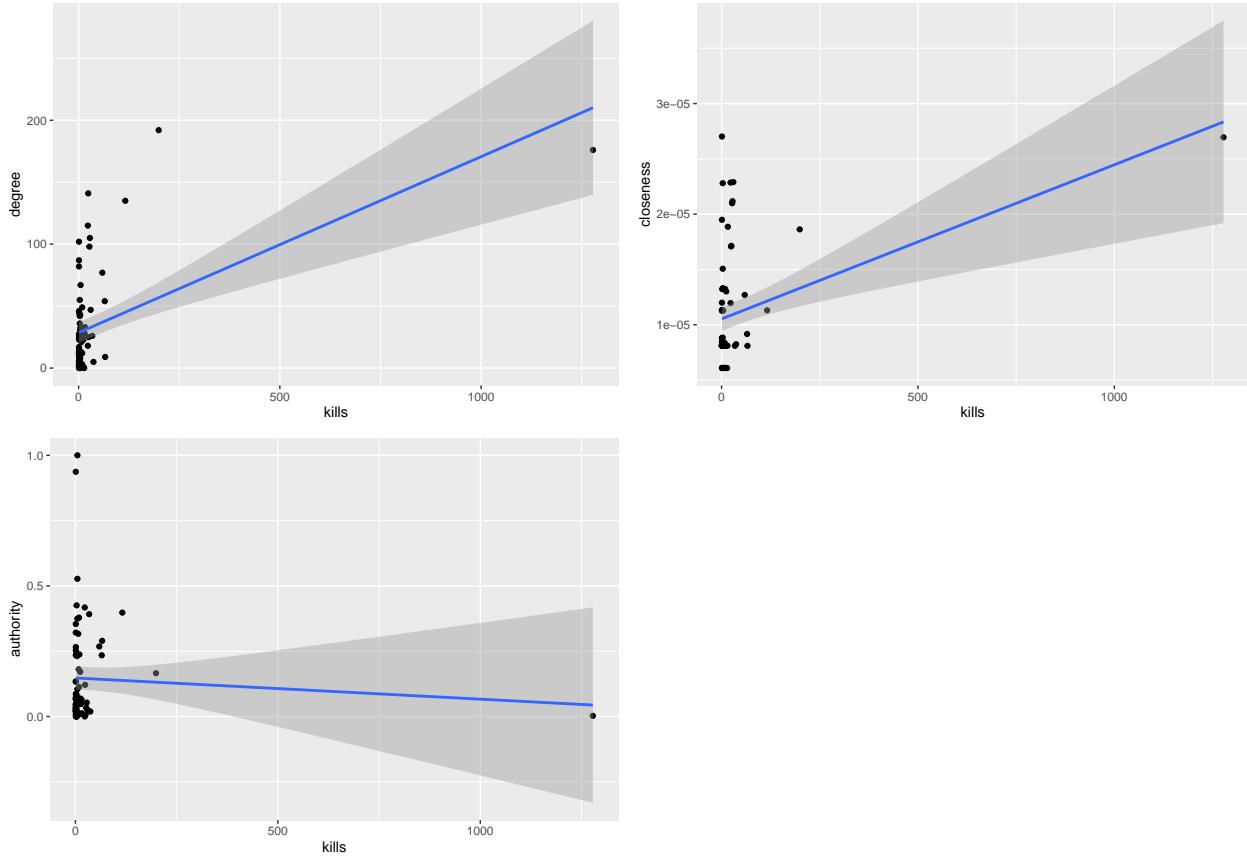
library(readr)
library(ggplot2)

ggplot(as.data.frame(net_degree), aes(kills, degree)) + geom_point() + geom_smooth(method=lm)

ggplot(as.data.frame(net_closeness), aes(kills, closeness)) + geom_point() + geom_smooth(method=lm)

#ggplot(as.data.frame(net_betweenness), aes(kills, betweenness)) + geom_point() + geom_smooth(method=lm)

ggplot(as.data.frame(net_authority), aes(kills, authority)) + geom_point() + geom_smooth(method=lm)
```



Je größer der Zusammenhang zwischen zwei Variablen, desto steiler ist die gezeichnete Funktion, die die Korrelation darstellt. Ist die Steigung positiv, besteht ein positiver Zusammenhang, das ansteigen der einen Variable wirkt sich ebenfalls auf die andere aus. Ein solcher positiver Zusammenhang ist wie vermutet bei den Zentralitätsmaßen Degree und Closeness zu erkennen. Bei der Authority dagegen ist nur eine leichte negative Steigung zu beobachten und schließt daher nicht auf einen Zusammenhang der Variablen.

Zusammenfassend lässt sich durch die dargestellten Untersuchungen sagen, dass sich die Killer mit den höchsten Opferzahlen aus Game of Thrones durch viele und enge Kontakte zu anderen Charakteren auszeichnen. Es konnte beobachtet werden, dass eine stärkere Ausprägung der Eigenschaften mit einer höheren Opferzahl in Verbindung gebracht werden kann. Im Gegenteil dazu beeinflusst das Einflussreichtum eines Charakters die Höhe dessen Opfer nicht. Hiermit lässt sich nun auch die zuvor gestellte Forschungsfrage beantworten, die Beziehungen zu anderen Charakteren und die Stellung des Charakters/Knotens im Netz ist ausschlaggebend für dessen Verhalten und dessen Bereitschaft anderen Charakteren ein Ende zu setzen.