Als ersten Schritt musste ich zuerst ein passendes Datenset finden. Da ich in diesem Semester die Tinder for Movies Challenge absolvieren, nahm ich direkt dieses Datenset. Die Daten kann man direkt von der Homepage mittels einer API herunterladen.

Hierbei handelt es sich um eine Datenbank, welche Filme zwischen dem Erstellungsjahr 2010 und 2020 beinhaltet. Dies ist mit seinen rund 2.7Mio Einträgen ein sehr grosses Datenset. Es handelt von 18493 Filmen welche von 60759 verschiedenen Usern bewertet wurden. Dies geschah mit einer Wertung zwischen 0.5 und 5. Insgesamt beinhaltet das Datenset elf verschiedene Spalten. Darunter neun welche Infos zum Film geben.

Jegliche gezeigten Visualisierungen, in diesem Bericht, wurden eigenhändig hergestellt. Der Herstellungscode wird unter meinem Github repository(<https://github.com/daviddedicfhwn/ivi>) aufzufinden sein.

1. LO1: Performance
   1. Wie viele Daten sind zu viel?

Es ist schwierig zu sagen, wie viele Daten zu viel für eine interaktive Visualisierung in R Studio sind, da es von verschiedenen Faktoren abhängt, wie zum Beispiel dem verwendeten Hardware und der Bandbreite, der Art der Visualisierung und der Verwendung von Leistungsverbesserungen. In der Regel können interaktive Visualisierungen in R Studio mit großen Datenmengen umgehen, aber es ist möglich, dass Leistungsprobleme auftreten, wenn die Daten sehr groß sind oder die Visualisierung sehr komplex ist. Eine Möglichkeit, die Leistung von interaktiven Visualisierungen in R Studio zu verbessern, besteht darin, Leistungsmanagement-Lösungen wie Tiling und Level of Detail Management zu verwenden und die Verwendung von GPUs zu nutzen. Es ist auch ratsam, die Daten zu priorisieren und hierarchisch zu organisieren, um die Leistung zu optimieren.

* 1. Welche Hardware und Bandbreite werden benötigt, um sicherzustellen, dass die Visualisierung reibungslos funktioniert?

Um sicherzustellen, dass eine interaktive Visualisierung in R Studio reibungslos funktioniert, benötigt man in der Regel einen leistungsstarken Computer mit genügend Arbeitsspeicher und Prozessorgeschwindigkeit, um die Visualisierungen zu berechnen und darzustellen. Eine schnelle Netzwerkverbindung und genügend Bandbreite sind auch wichtig, insbesondere wenn die Daten größer sind und von einem Remote-Server geladen werden müssen. Es ist schwierig zu sagen, welche genauen Hardware- und Bandbreitenanforderungen für eine reibungslose Visualisierung in R Studio erforderlich sind, da es von verschiedenen Faktoren abhängt, wie der Größe und Komplexität der Daten und der Art der Visualisierung. In der Regel ist es ratsam, einen leistungsstarken Computer und eine schnelle Netzwerkverbindung zu verwenden, um die besten Ergebnisse zu erzielen.

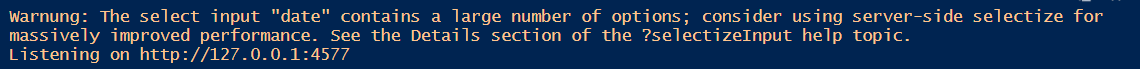
* 1. Meine Hardware und Bandbreite

Ich besitze einen relative schwachen Laptop, welcher sehr schnell an seine Grenze kommt, wenn es sich um grosse Datenmengen handelt. Hier folgen nun einige Spezifikationen zu meinem Laptop:

|  |  |
| --- | --- |
| Spezifikation | Bezeichnung |
| RAM Grösse | 8 GB |
| Prozessor | Intel Core I5 6. Generation |
| GPU | Intel HD Graphics 520 |
| Bandbreite zu Hause | Download: 100Mbit/s  Upload: 100Mbit/s |

Hier sieht man dass meine Inhalte nicht besonders Leistungsfähig sind. Ich freue mich jedoch zu versuchen das beste aus diesen zu bekommen!

* 1. Versuch Erstellung von einem interaktiven Plot vor der Bearbeitung der Effizienz

Wie bereits erwartet funktionierte der Plot nicht. Folgende Meldung kam dabei raus: 

Um dieses Problem zu umgehen kann man einige Dinge ändern.

* 1. Effizientes bearbeiten von Daten

Die Problembehandlug sollte als erstes bei der Wurzel beginnen, dem Einlesen. Es gibt viele Arten ein Dokument einzulesen. Durch das Internet konnte ich recherchieren dass die Funktionen ‘fread()’ und ‘read\_csv()’ am besten geeignet sind. Beide Funktionen sind effiziente Methoden zum Einlesen von Daten in R. Aber ‘fread()’ könnte für größere Datenmengen allgemein effizienter sein, während ‘read\_csv()’ möglicherweise für kleinere Datenmengen effizienter ist. Um herauszufinden, welche der beiden schneller ist, habe ich einen kleinen Test durchgeführt(im Github zu sehen). Da kam heraus dass in meine Fall ‘fread()’ besser ist. Danach sollte man die Daten mittels ‘Plotly’ und ‘LOD’ visualisieren. Level of Detail Management ist eine Technik, die verwendet wird, um die Leistung von Visualisierungen zu verbessern, indem sie die Anzahl der gezeigten Datenpunkte basierend auf der Größe und Auflösung des Bildschirms und der Anzahl der gezeigten Datenpunkte dynamisch anpasst. Dies kann dazu beitragen, dass interaktive Plots schneller ausgeführt werden, insbesondere wenn die Datenmenge groß ist.

LO2: Dashboard design principles

LO3: HCI Basics

LO4: Evaluation