

Projektbericht zum Modul Information Retrieval und
Visualisierung Sommersemester 2022

Smartphone Analyse Tool

Tom Schindler

24. August 2024

Inhaltsverzeichnis

1	Einleitung	3
1.1	Anwendungshintergrund	3
1.2	Zielgruppen	4
1.3	Überblick und Beiträge	4
2	Daten	6
3	Visualisierungen	9
3.1	Analyse der Anwendungsaufgaben	9
3.2	Anforderungen an die Visualisierungen	9
3.3	Präsentation der Visualisierungen	10
3.3.1	Scatterplot	11
3.3.2	Parallel-Koordinaten-Plot	12
3.3.3	Tree	13
3.3.4	Starplot	15
3.4	Interaktion	16
4	Implementierung	18
5	Anwendungsfälle	21
5.1	Anwendung Scatterplot	21
5.2	Anwendung Parallel-Koordinaten-Plot	21
5.3	Anwendung Tree	21
5.4	Anwendung Starplot	21
6	Verwandte Arbeiten	21
7	Zusammenfassung und Ausblick	21

1 Einleitung

In der wettbewerbsintensiven Welt der Smartphone-Technologie ist es für Unternehmen entscheidend, die Präferenzen ihrer Kunden genau zu verstehen. Die Analyse von Smartphone-Hardwaredaten bietet wertvolle Einblicke in das Nutzerverhalten und die Vorlieben der Kunden hinsichtlich verschiedener Gerätefunktionen. Diese Erkenntnisse können genutzt werden, um Marketingstrategien gezielt auszurichten und maßgeschneiderte Produkte und Dienstleistungen zu entwickeln, die den spezifischen Bedürfnissen der Nutzer entsprechen.

Ziel dieses Berichts ist es, ein speziell entwickeltes Analyse-Tool vorzustellen, das dazu dient, die Hardwaredaten von Smartphones zu visualisieren und zu analysieren. Im Fokus stehen dabei Fragestellungen wie: Welche Hardwaremerkmale sind für Kunden besonders wichtig? Wie beeinflussen diese Präferenzen die Kaufentscheidungen (Bewertung)? Und wie können diese Informationen genutzt werden, um gezielte Marketingstrategien zu entwickeln und Produkte zu optimieren?

Mithilfe von Techniken der Informationsvisualisierung wird im weiteren Verlauf des Berichts gezeigt, wie diese Fragestellungen beantwortet werden können. Dies unterstützt Unternehmen dabei, tiefere Einblicke in die Präferenzen ihrer Kunden zu gewinnen, um so den Erfolg auf dem Markt zu steigern und innovativere, auf die Bedürfnisse der Kunden abgestimmte Produkte zu entwickeln.

1.1 Anwendungshintergrund

Die Analyse von Smartphone-Hardwaredaten spielt eine zentrale Rolle, wenn es darum geht, Kundenpräferenzen zu verstehen und darauf basierend gezielte Marketingstrategien zu entwickeln. Moderne Smartphones bestehen aus einer Vielzahl von Hardwarekomponenten, wie Prozessoren, Speichereinheiten, Batterien und Bildschirmgröße, die alle unterschiedlich wahrgenommen und genutzt werden. Diese Daten können durch geeignete Analysetools extrahiert und visuell aufbereitet werden, um Muster und Trends zu erkennen, die für Unternehmen von großem Interesse sind.

Ein erfolgreiches Analyse-Tool muss daher in der Lage sein, diese komplexen und umfangreichen Daten in einer Form darzustellen, die für die Entscheidungsfindung nützlich ist. Dies erfordert nicht nur eine leistungsfähige Datenverarbeitung, sondern auch effektive Visualisierungstechniken, die es ermöglichen, wichtige Erkenntnisse schnell und klar zu erfassen.

Besonders relevant ist dabei, wie die Visualisierung von Hardwaredaten auf die Präferenzen der Kunden hinweist und wie sich diese Informationen in die Entwicklung maßgeschneiderter Produkte und Dienstleistungen umsetzen lassen. Dabei ist es entscheidend, dass die Visualisie-

rungen so gestaltet sind, dass sie sowohl detaillierte Einblicke ermöglichen als auch übersichtlich genug bleiben, um auf den ersten Blick verständlich zu sein.

1.2 Zielgruppen

Die Zielgruppe für das Smartphone Analyse Tool umfasst hauptsächlich Produktmanager, Marketingexperten und Unternehmensanalysten in der Smartphone Technologiebranche. Diese Personen sind daran interessiert, tiefere Einblicke in die Kundenpräferenzen zu gewinnen, um fundierte Entscheidungen bezüglich Produktentwicklungen und Marketingstrategien zu treffen. Darüber hinaus könnten auch Führungskräfte und Entscheidungsträger aus dem Bereich strategische Planung von den Ergebnissen profitieren, da sie auf Basis dieser Erkenntnisse marktgerechte Produkte entwickeln möchten.

Bei dieser Zielgruppe kann von einem soliden Vorwissen in den Bereichen Marktanalyse, Produktentwicklung und Kundenforschung ausgegangen werden. Viele der Anwender sind bereits mit grundlegenden Konzepten der Datenanalyse vertraut und haben ein Verständnis für die Bedeutung von Hardwaremerkmalen bei der Kaufentscheidung von Konsumenten. Sie benötigen jedoch spezialisierte Werkzeuge, die es ihnen ermöglichen, komplexe Daten auf intuitive Weise zu visualisieren und so gezielt Erkenntnisse über Kundenbedürfnisse und -verhalten zu gewinnen.

Die durch die Visualisierungen adressierten Informationsbedürfnisse betreffen insbesondere:

- **Identifizierung von Kundenpräferenzen:** Welche Hardwarekomponenten sind für die Kunden am wichtigsten, und wie wirken sich diese Präferenzen auf Kaufentscheidungen aus?
- **Produktentwicklung:** Welche Features sollten in zukünftigen Produktgenerationen stärker berücksichtigt oder optimiert werden, um die Wettbewerbsfähigkeit zu erhöhen?

Die Visualisierungen bieten den Anwendern somit eine klare und übersichtliche Darstellung der relevanten Daten, die es ihnen ermöglicht, ihre strategischen Ziele effektiver zu erreichen.

1.3 Überblick und Beiträge

Mit Hilfe des Analyse-Tools ist es möglich, aus einer großen Datenmenge[2], die zahlreiche moderne Smartphone-Modelle umfasst, gezielte Vergleiche und Analysen durchzuführen. Dabei werden unter anderem Daten zu den Hardware-Komponenten, wie verbauter Prozessor, vorhandener Speicher und Batteriekapazität, vergleichbar gemacht. Darüber hinaus werden Preis-Leistungs-Aspekte anhand von Kundenbewertungen veranschaulicht, was einen wichtigen Beitrag zur Identifizierung von Kundenpräferenzen leistet.

Die Daten werden in vier verschiedenen Visualisierungen dargestellt, um die Informationsbedürfnisse der Zielgruppe umfassend zu erfüllen. Diese Visualisierungen sind:

- **Scatterplot:** Stellt jeweils zwei numerische Daten auf zwei Achsen dar und ermöglicht so die Analyse von Korrelationen zwischen verschiedenen Variablen.
- **Parallel-Koordinaten-Plot:** Zeigt alle numerischen Daten der Datenmenge in einer mehrdimensionalen Visualisierung, um komplexe Zusammenhänge übersichtlich darzustellen.
- **Baum-Datstellung:** Visualisiert alle Smartphone-Hersteller und deren Modelle aus der Datenmenge in einer hierarchischen Struktur.
- **Starplot:** Stellt alle numerischen Daten zu einem Smartphone in einer sternförmigen Visualisierung dar und bietet so einen kompakten Überblick über die Leistungsmerkmale eines Geräts.

In den folgenden Abschnitten werden die verwendeten Daten, die erstellten Visualisierungstechniken und deren Umsetzung und deren Anwendungsfälle detailliert beschrieben.

2 Daten

Die verwendeten Daten stammen von Kaggle[2] und liegen im CSV-Format vor. Diese Datensammlung umfasst Informationen zu verschiedenen Hardware-Komponenten, Preisen, Bewertungen und weiteren Merkmalen einer großen Anzahl älterer und aktueller Smartphone-Modelle (insgesamt 984 Einträge in der CSV-Datei). Die Datenspalten der CSV-Datei[2] umfassen:

- Brand - Marke des Smartphones
- Model - Model des Smartphones
- Price - Price des Smartphones
- Rating - durchschnittliche Kundenbewertung
- 5G - hat 5G oder nicht
- NFC - hat NFC oder nicht
- IR Blaster - hat IR Blaster oder nicht
- Processor Name - Name des Prozessors
- Processor Brand - Marke des Prozessors
- NumCores - Kernanzahl
- Processor Speed - Anzahl der Zyklen pro Sekunde, die eine CPU ausführen kann
- Ram and internal storage (ram) - RAM and interner Speicher
- Battery - Kapazität der Battery
- Fast Charging - hat Fast Charging oder nicht, und falls ja Kapazität
- Internal Mem - interner Speicher des Smartphones
- RAM - RAM des Smartphones
- Screen Size - Bildschirmgröße des Smartphones
- Resolution - Auflösung des Smartphones
- Refresh Rate - Bildwiederholfrequenz des Smartphones
- Camera - Kameraqualität der Rück- und Frontkamera
- Card - Speicherkarte unterstützt oder nicht und falls ja Speicherkapazität
- OS - Betriebssystem des Smartphones

Die meisten Datenspalten aus der CSV-Datei wurden in die Datenvorverarbeitung der Anwendung integriert, die in Elm programmiert wurde. Ausgenommen davon ist die Spalte „Ram and internal storage (ram)“, da hierfür bereits separate Spalten vorhanden sind. Einige weitere Spalten wie „Camera“, „Card“, „OS“ und „Resolution“ wurden bei der Vorverarbeitung lediglich als Zeichenketten gespeichert und werden in den Visualisierungen der Anwendung nicht weiter

berücksichtigt. Diese Spalten weisen Inkonsistenzen in ihren Datenwerten auf (meist in Textform), was es schwierig machte, daraus numerisch vergleichbare Werte zu erzeugen. Diese Daten könnten in zukünftigen Arbeiten weiter aufbereitet und in die Visualisierungen integriert werden.

Alle anderen Daten fließen entweder direkt oder indirekt (z.B. als Filteroption oder Beschriftung) in die Visualisierungen ein. Bei der Vorverarbeitung wurden bei den meisten numerischen Werten bestimmte Zeichen oder Wörter entfernt, um den reinen numerischen Wert zu extrahieren (z.B. Screen Size = „6.6 inches“ zu „6.6“ oder Processor Speed = „2.2 GHz Processor“ zu „2.2“). Daten, die das Vorhandensein eines Merkmals beschreiben, wurden in Elm in Wahrheitswerte übersetzt (z.B. „5G“), welche meist als Filteroption dienen. Alle weiteren beschreibenden (textuellen) Daten wurden in ihrer ursprünglichen Form als Zeichenketten gespeichert und dienen hauptsächlich der Beschriftung oder als Filter. Fehlende Datenwerte wurden in Elm als Nothing-Werte behandelt.

Verarbeitungsschritte:

1. Auswahl der Daten nach ihrer Relevanz für das Projekt
2. Zuordnung der Daten zu Kategorien (numerische vergleichbare Werte, Wahrheitswerte, textuelle Beschreibungen)
3. Falls erforderlich, manuelle Korrektur einzelner fehlerhafter Werte
4. Parsen der Datenwerte aus der CSV-Datei:
 - Bei numerischen Werten in Zeichenketten: Extraktion durch Entfernen von Textinhalten und Umwandlung in den entsprechenden Datentyp
 - Bei numerischen Werten: Extraktion und Umwandlung in den entsprechenden Datentyp
 - Bei Wahrheitswerten: Übersetzung in Elm-Wahrheitswerte
 - Bei fehlenden Werten (unabhängig von der Art): Übersetzung in Elm-Nothing
 - Andernfalls: Speicherung als Zeichenkette
5. Speicherung der geparsen Daten im Model

Anmerkung: Da bei einigen wenigen Smartphone-Modellen in der CSV-Datei falsche Datenwerte für den RAM und den internen Speicher vorlagen, wurden diese Daten korrigiert oder ergänzt. Beispielsweise waren bei einigen Modellen RAM und interner Speicher vertauscht oder der interne Speicher wurde fälschlicherweise unter RAM eingetragen (z.B. RAM = „512GB“).

Die ausgewählten Daten eignen sich gut (bzw. ausreichend) für die Analyse der Smartphone-Daten in Bezug auf die Kundenzufriedenheit und erfüllen die Informationsbedürfnisse der definierten Zielgruppe. Aus den Daten lassen sich Zusammenhänge zwischen verbauter Hardware

und Preis sowie den Kundenbewertungen ableiten und auswerten. Diese Erkenntnisse ermöglichen es, Schwachstellen und Erfolge der verschiedenen Modelle zu identifizieren. Die gewonnenen Informationen können von Produktmanagern, Marketingexperten und Unternehmensanalysten in die zukünftige Produktentwicklung und -planung neuer Modelle einfließen, um wirtschaftlich erfolgreiche und kundenorientierte Smartphones zu entwickeln.

Leider fehlten in der ausgewählten Datenmenge zusätzliche Informationen wie das Erscheinungsdatum und die Anzahl der verkauften Einheiten. Diese Daten wären jedoch äußerst wertvoll, um die Informationsbedürfnisse der definierten Zielgruppe noch besser zu unterstützen. Das Erscheinungsdatum eines Smartphones könnte Rückschlüsse auf die in einem bestimmten Jahr oder einer bestimmten Zeit verwendete Hardware ermöglichen, was helfen würde, ältere Modelle bei der Produktentwicklung und -planung auszuschließen. Die Daten zu den verkauften Einheiten könnten zudem den tatsächlichen Erfolg eines Modells verdeutlichen, anstatt sich ausschließlich auf Kundenbewertungen zu stützen.

3 Visualisierungen

3.1 Analyse der Anwendungsaufgaben

Um das Zielproblem zu lösen, müssen die Anwender mehrere spezifische Anwendungsaufgaben bearbeiten. Diese Aufgaben umfassen die Analyse von Kundenpräferenzen, das Erkennen von Mustern in den Kundenbewertungen, die Bewertung des Preis-Leistungs-Verhältnisses verschiedener Smartphone-Modelle und das Ableiten von Empfehlungen für die Produktentwicklung und -planung.

Eine der zentralen Aufgaben besteht darin, Zusammenhänge zwischen verbauter Hardware und Kundenbewertungen zu erkennen. Dies erfordert, dass die Anwender in der Lage sind, die Beziehung zwischen technischen Spezifikationen, Preis und Kundenzufriedenheit zu verstehen und zu bewerten. Hierfür ist es hilfreich, ein mentales Modell zu entwickeln, das die Bedeutung und den Einfluss einzelner Hardwarekomponenten auf die Gesamtbewertung eines Smartphones erklärt. Beispielsweise müssen die Anwender verstehen, dass ein leistungsstarker Prozessor oder eine große Speicherkapazität für bestimmte Nutzergruppen besonders wertvoll sind und sich daher positiv auf die Bewertungen auswirken können.

Diese mentalen Modelle sind notwendig, um die Anwendungsaufgaben effektiv lösen zu können. Die Visualisierungen in der Anwendung tragen entscheidend dazu bei, diese Modelle aufzubauen und zu stärken. Durch die anschauliche Darstellung von Daten wie Hardwarekomponenten, Kundenbewertungen und Preisen werden die Beziehungen zwischen diesen Faktoren klarer und greifbarer, was es den Anwender ermöglicht, fundierte Entscheidungen zu treffen.

Insgesamt unterstützen die Visualisierungen den Aufbau und die Anwendung der notwendigen mentalen Modelle, die für eine erfolgreiche Bearbeitung der Anwendungsaufgaben unerlässlich sind. Die klaren und übersichtlichen Darstellungen erleichtern es den Anwender, komplexe Datenmuster zu verstehen und sinnvoll in ihren Entscheidungsprozessen zu nutzen.

3.2 Anforderungen an die Visualisierungen

Aus der Analyse des Zielproblems und der damit verbundenen Anwendungsaufgaben lassen sich mehrere Anforderungen an das Design der Visualisierungen ableiten, um die Informationsbedürfnisse der Anwender effektiv zu erfüllen.

- **Klarheit und Verständlichkeit der Darstellung:** Die Visualisierungen müssen so gestaltet sein, dass sie auch komplexe Daten einfach und verständlich vermitteln. Dies erfordert klare Achsenbeschriftungen, eine einfache auswertbare Visualisierung und eine intuitive Benutzeroberfläche die es den Anwender ermöglicht, auf den ersten Blick die relevanten Informationen zu erfassen.

- **Interaktive Filteroptionen:** Da die Anwender möglicherweise unterschiedliche Hardwaremerkmale oder Preis-Leistungs-Aspekte analysieren möchten, sollten die Visualisierungen interaktive Filteroptionen bieten. Diese ermöglichen es den Nutzer, gezielt bestimmte Daten auszuwählen und sich auf relevante Aspekte zu konzentrieren. Zum Beispiel könnten Nutzer spezifische Hardwarekomponenten oder Marken filtern, um detaillierte Analysen durchzuführen.
- **Vergleichbarkeit von Modellen:** Eine zentrale Anforderung ist die Fähigkeit, mehrere Smartphone-Modelle direkt miteinander zu vergleichen. Die Visualisierungen sollten es ermöglichen, verschiedene Modelle nebeneinander darzustellen, um Unterschiede in Hardware, Preis und Kundenbewertungen sichtbar zu machen. Dies könnte durch die Nutzung von Scatterplots, Parallelkoordinaten-Plots oder Starplots unterstützt werden.
- **Darstellung von Korrelationen:** Die Visualisierungen sollten in der Lage sein, Korrelationen zwischen verschiedenen Variablen, wie z.B. verbauter Hardware und Kundenbewertungen, deutlich zu machen. Scatterplots sind hierfür besonders geeignet, da sie die Beziehung zwischen zwei numerischen Werten visualisieren und so Korrelationen schnell erkennbar machen.
- **Hierarchische Strukturen:** Da die Daten aus verschiedenen Smartphone-Herstellern und ihren jeweiligen Modellen bestehen, sollten die Visualisierungen auch hierarchische Strukturen abbilden können. Eine Baum-Darstellung wäre hier sinnvoll, um die Organisation der Daten übersichtlich darzustellen und den Anwender die Navigation durch die verschiedenen Modelle zu erleichtern.
- **Einfache Handhabung und Benutzerfreundlichkeit:** Da die Anwender möglicherweise kein tiefgehendes technisches Wissen besitzen, sollten die Visualisierungen eine einfache und benutzerfreundliche Interaktion ermöglichen. Dies beinhaltet eine intuitive Bedienung sowie klare und verständliche Anweisungen oder Tooltips, die den Umgang mit den Visualisierungen erleichtern.

Durch die Umsetzung dieser Anforderungen kann sichergestellt werden, dass die Visualisierungen den Anwender wertvolle Einblicke bieten und sie bei der Erreichung ihrer Ziele effektiv unterstützen.

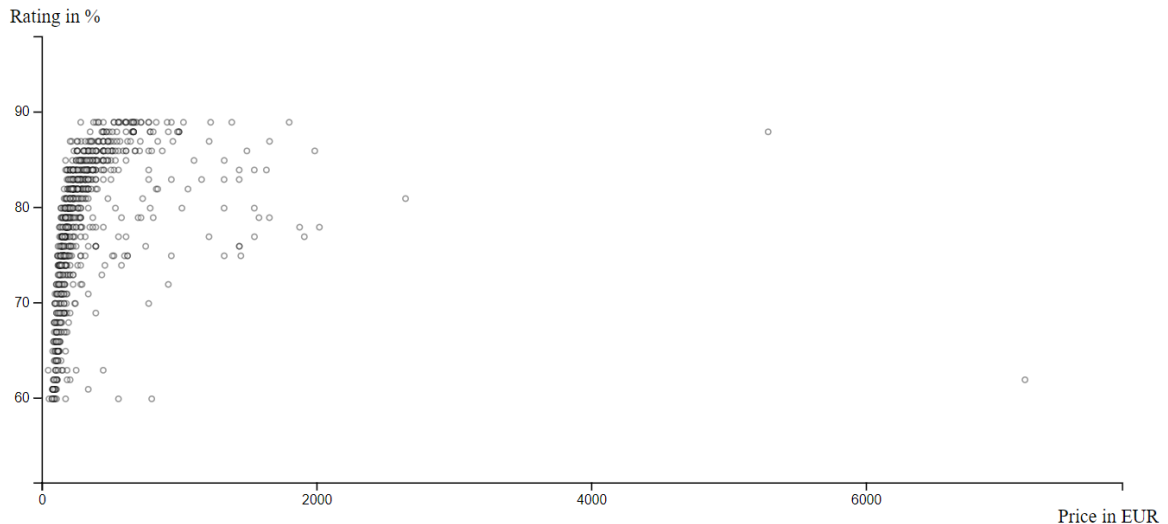
3.3 Präsentation der Visualisierungen

In den folgenden Abschnitten werden die vier Visualisierungen der Anwendung detailliert beschrieben.

3.3.1 Scatterplot

Hints: By clicking on a point, the plot switch to the star plot and show the point data. By hovering over a point, you get the model name, ID und the exactly x and y values.

x-Axis: Price y-Axis: Rating



Die erste Visualisierung ist ein Scatterplot, der Zusammenhänge zwischen verschiedenen Variablen entlang zweier Achsen darstellt. Die Datenwerte werden als Punkte in einem 2D - Koordinatensystem codiert und abgebildet. Jeder Punkt besitzt einen identischen Transparenzwert, wodurch überlappende Punkte leichter erkennbar sind: Einzelne Punkte erscheinen heller (hellgrau), während eine Ansammlung überlappender Punkte dunkler wirkt (dunkelgrau bis schwarz). Diese farbliche Gestaltung erleichtert es, viele Datenpunkte mit ähnlichen Werten von einzelnen Punkten zu unterscheiden.

Eine weitere implementierte Funktion ist die Auswahlmöglichkeit der Variablen für die beiden Achsen. Nutzer können vordefinierte Variablen auswählen und diese auf einer der beiden Achsen darstellen lassen, um Korrelationen zwischen verschiedenen Variablen wie verbauter Hardware, Kundenbewertungen oder Preis zu verdeutlichen.

Die Achsen sind linear dargestellt, was eine einfache und verständliche Vermittlung der Daten ermöglicht. Diese lineare Darstellung ist gut geeignet, da die ausgewählten Datenwerte in der Regel keine zu großen Abstände aufweisen.

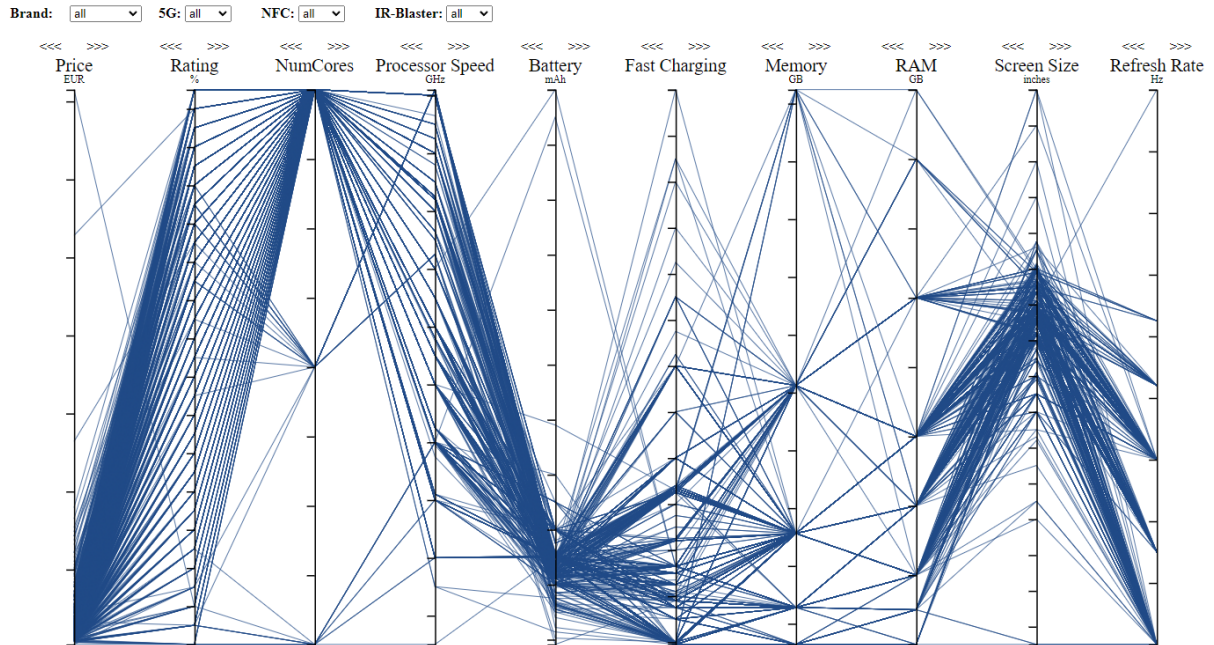
Zusätzlich wurden interaktive Funktionen zu den Punkten hinzugefügt. Wenn ein Punkt mit dem Mauszeiger überfahren wird, werden die genauen Datenwerte für beide Achsen sowie das Smartphone-Modell, das dieser Punkt repräsentiert, angezeigt. Beim Anklicken eines Punktes entsteht eine weitere Interaktion zwischen den verschiedenen Plots, die im Abschnitt „Interaktionen“ detaillierter beschrieben wird.

erfüllte Anforderungen:

- Klarheit und Verständlichkeit der Darstellung

- Vergleichbarkeit von Modellen (eingeschränkt, aber an zwei Variablen)
- Darstellung von Korrelationen
- Einfache Handhabung und Benutzerfreundlichkeit

3.3.2 Parallel-Koordinaten-Plot



Der Parallel-Koordinaten-Plot bildet die Datenmenge auf ein mehrdimensionale Datendarstellung ab. In der Anwendung werden zehn Achsen dargestellt, die jeweils einen der zehn numerischen Werte eines Smartphones repräsentieren (darunter Preis, Bewertung usw.). Jedes Smartphone wird durch eine Linie dargestellt, die alle Achsen des Plots durchläuft. Dabei werden die Werte an den jeweiligen Achsen bestimmt und für jedes Smartphone-Modell eingetragen. So wird es möglich, den Zusammenhang aller numerischen Attribute der Smartphone-Datenmenge innerhalb einer einzigen Visualisierung darzustellen, um effizient mögliche schwache oder optimale Konfigurationen von Hardware, Preis und Kundenbewertungen zu ermitteln.

Die Linienfarbe (blau) wurde so gewählt, dass sich die Beschriftungen an den Achsen (schwarz) gut davon abheben und deutlich lesbar sind, ohne visuell ineinander zu überlappen.

Wie bei der ersten Visualisierung sind die Achsen linear skaliert. Um eine übersichtlichere Darstellung für die Anwender zu erreichen, wurden die Wertebeschriftungen an den Achsen weggelassen. Optional kann die Beschriftung der Achsenwerte eingeblendet werden, wenn sich der Mauszeiger über der Achse oder deren Beschriftung befindet. Diese Einblendung der Werte kann für präzisere Analysen nützlich sein.

Eine weitere nützliche Interaktion bieten die «< und »> „Buttons“, mit denen die Reihenfolge der Achsen vertauscht werden kann. Dadurch können die Anwender benutzerdefinierte Reihen-

folgen der Achsen erstellen. Eine andere Anordnung der Achsen kann eine neue Perspektive auf die Daten eröffnen und möglicherweise bessere, zielgerichtete Ergebnisse liefern.

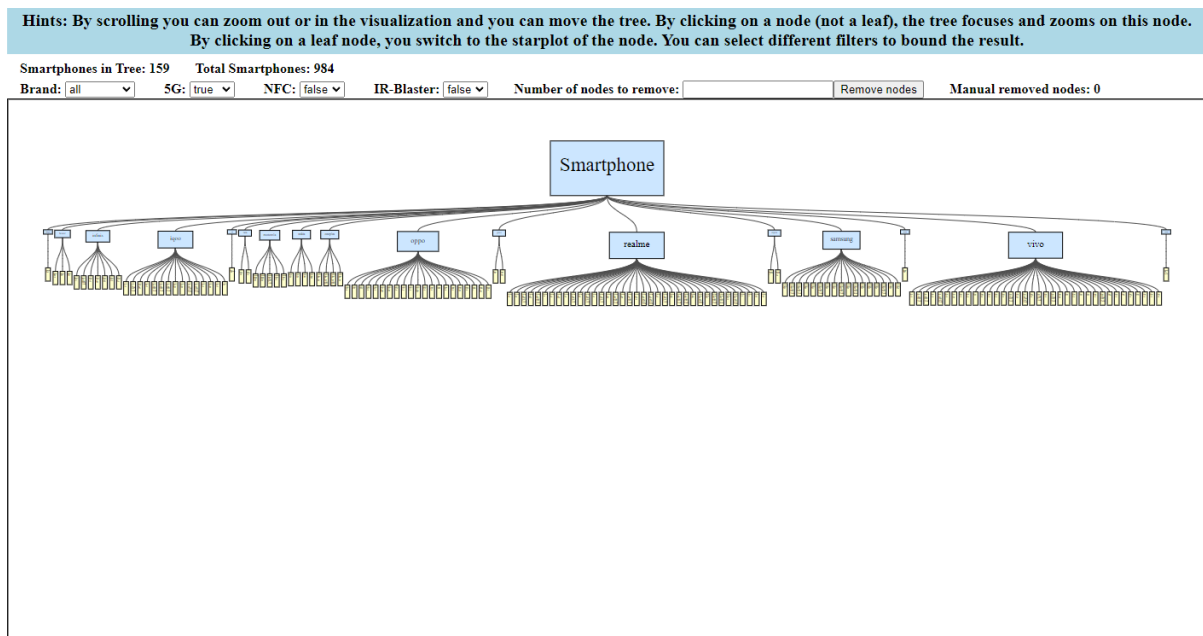
Zusätzlich bietet die Visualisierung Filteroptionen, mit denen die Ergebnismenge im Plot eingeschränkt werden kann, um detailliertere, anwendungsorientierte Analysen durchzuführen. Die Datenmenge kann nach Smartphone-Marken und dem Vorhandensein bestimmter Smartphone-Funktionen (z.B. 5G, NFC, IR-Blaster) gefiltert werden.

Beim Anklicken der Achsenbeschriftung über der Achse wechselt die Darstellung zu einem anderen Plot, was im Abschnitt „Interaktionen“ detaillierter beschrieben wird.

erfüllte Anforderungen:

- Klarheit und Verständlichkeit der Darstellung
- Interaktive Filteroptionen (Brand, 5G, NFC, IR-Blaster)
- Vergleichbarkeit von Modellen (anhand von zehn Achsen)
- Darstellung von Korrelationen
- Einfache Handhabung und Benutzerfreundlichkeit

3.3.3 Tree



Für die dritte Visualisierung der Anwendung wurde eine hierarchische Baumdarstellung (Tree) gewählt. Diese Visualisierung ordnet den verschiedenen Smartphone-Marken die jeweiligen Modelle zu und ermöglicht es den Anwender, einen umfassenden Überblick über alle Modelle eines Herstellers zu erhalten.

Dank benutzerfreundlicher Interaktionsmöglichkeiten und Navigationsoptionen können sich die

Anwender auf das Wesentliche konzentrieren und effizient, auch ohne technisches Wissen, Analysen wie z.B. Marktanalysen durchführen. Zu diesen Optionen gehören das Zoomen in und aus verschiedenen Bereichen des Baumes, das Fokussieren auf einen bestimmten Abschnitt durch Anklicken eines Baumknotens (nicht Blattknoten) sowie das Navigieren durch den Baum durch Bewegen und Gedrückthalten des Mauszeigers.

Die Knoten des Baumes sind nach ihren Bezeichnern sortiert (Hersteller, Model) und variieren in der Größe je nach Anzahl der ihnen untergeordneten Smartphone-Modelle. Diese Sortierung erleichtert das Auffinden spezifischer Knoten, während die unterschiedlichen Knotengrößen eine visuelle Darstellung der Anzahl untergeordneter Modelle bieten, wodurch die Übersichtlichkeit verbessert wird.

Zusätzlich bietet die Baumdarstellung Filteroptionen, mit denen die Knotenanzahl im Baum eingeschränkt werden kann, um spezifische Smartphone-Modelle zu finden und weitere Analysen durchzuführen. Die Datenmenge kann nach Smartphone-Marken und dem Vorhandensein bestimmter Smartphone-Funktionen (z.B. 5G, NFC, IR-Blaster) gefiltert werden. Anzeigen informieren zudem über die aktuelle Anzahl der im Baum dargestellten Smartphone-Modelle und die Gesamtzahl der Smartphones in der Datenmenge.

Die Datenmenge kann auch manuell reduziert werden, indem eine Anzahl zu entfernender Knoten angegeben wird. Diese Funktion wurde ursprünglich für Testzwecke eingeführt, um die Darstellung bei unterschiedlichen Knotengrößen im Baum zu testen. Es stellte sich jedoch heraus, dass eine Begrenzung des Baumes die Übersichtlichkeit erhöht, weshalb diese optionale Anpassungsmöglichkeit für die Anwender beibehalten wurde.

Durch das Anklicken eines Blattknotens (Smartphone-Modells) gelangt man zu einer detaillierteren Visualisierung, die im Abschnitt „Interaktionen“ näher beschrieben wird.

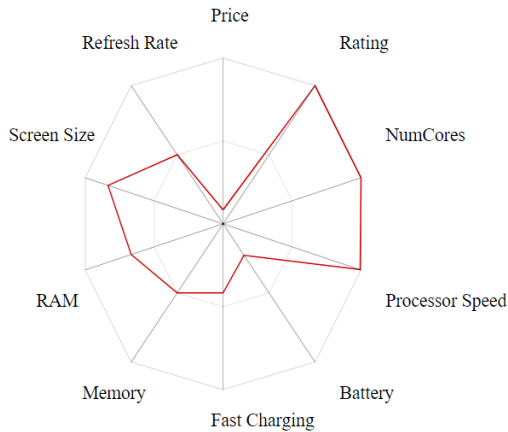
erfüllte Anforderungen:

- Klarheit und Verständlichkeit der Darstellung
- Interaktive Filteroptionen (Brand, 5G, NFC, IR-Blaster)
- Hierarchische Strukturen
- Einfache Handhabung und Benutzerfreundlichkeit

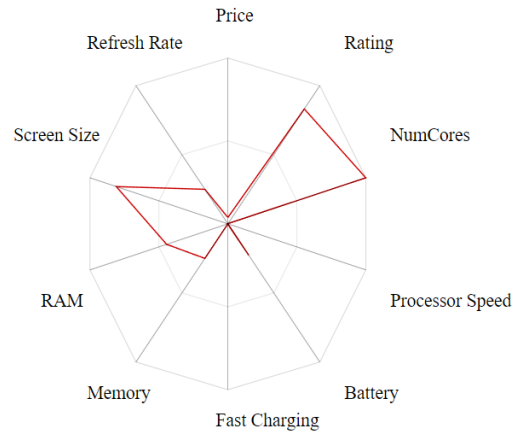
3.3.4 Starplot

Hints: You can compare two smartphones at the same time. Each of the two starplots has an input field where an ID from a smartphone can be entered to display it in the starplot. By hovering over a plot label, you can see the exactly value for the label.

Starplot1 | ID:0 | OnePlus 11 5G
Select ID1: Submit for SP1



Starplot2 | ID:1019 | Samsung Galaxy M52s 5G
Select ID2: Submit for SP2



Zusätzlich zu den drei bereits vorgestellten Visualisierungen wurde ein Starplot mit Vergleichsfunktion integriert, um die Möglichkeiten für effiziente Analysen durch weitere sinnvolle Interaktionen zu erweitern (die Interaktionen zwischen den Plots werden im Abschnitt „Interaktionen“ detaillierter beschrieben).

Die Starplot-Darstellung bietet der Zielgruppe die Möglichkeit, zwei Smartphones nebeneinander zu vergleichen. Ein Starplot ist eine sternförmige Darstellung mehrdimensionaler Daten, die in diesem Fall Informationen zu einem bestimmten Smartphone visualisiert. Die einzelnen Variablenwerte des Smartphones werden auf zehn Achsen (ähnlich dem Parallelkoordinaten-Plot) eingetragen und durch eine Linie miteinander verbunden, die sich zu einem Kreis schließt. Anhand dieser Linie können die Anwender an den Achsen ablesen, wie stark eine Hardwarekomponente, der Preis oder die Kundenbewertung im Vergleich zum Minimum und Maximum aller Smartphone-Modelle ausgeprägt ist.

Im Designprozess wurde entschieden, zwei Starplots nebeneinander darzustellen, um einen direkten Vergleich zu ermöglichen und Unterschiede sowie Gemeinsamkeiten zwischen den Modellen einfach erkennen zu können.

Die Starplots ändern sich je nach Eingabe der Smartphone-ID oder durch Interaktionen mit anderen Plots, sodass jeweils das entsprechende Modell dargestellt wird. Der exakte Wert, den ein Smartphone für eine bestimmte Variable aufweist, kann angezeigt werden, indem der Mauszeiger über den Achsenbezeichner bewegt wird.

erfüllte Anforderungen:

- Klarheit und Verständlichkeit der Darstellung
- Vergleichbarkeit von Modellen (zwei nebeneinander)

- Darstellung von Korrelationen (einigermaßen, da es sich auf ein Smartphone bezieht)
- Einfache Handhabung und Benutzerfreundlichkeit

3.4 Interaktion

Die plot-übergreifenden Interaktionen bieten eine effiziente und vorteilhafte Bedienung, die über die einzelnen Visualisierungen hinausgeht und zusätzliche Analysemöglichkeiten eröffnen. Innerhalb der Anwendung wurden drei Hauptinteraktionen dieser Art implementiert.

- **Scatterplot -> Starplot** : Beim Klicken auf einen Datenpunkt im Scatterplot wechselt die Anwendung zur Starplot-Darstellung und projiziert die Datenwerte des ausgewählten Smartphones auf das ältere (upgedatete) der beiden vorhandenen Starplots. Dadurch kann der Anwender bei Bedarf alle Variablen des Smartphones, das im Scatterplot ausgewählt wurde, einsehen und es gleichzeitig mit einem anderen Modell vergleichen. Diese Interaktion ist besonders nützlich, wenn der Anwender von einer groben Analyse mehrerer Datenwerte (z.B. zur Identifizierung optimaler Smartphone-Komponenten) zu einer detaillierten Analyse einzelner Modelle übergehen möchte. So können einzelne Smartphone-Modelle hinsichtlich ihrer Komponenten, Kundenbewertungen und Preise miteinander verglichen werden, um die eigene Produktionsplanung und -entwicklung zu optimieren.
- **Parallel-Koordinaten-Plot -> Scatterplot** : Beim Klicken auf einen Achsenbezeichner (über der Achse) im Parallel-Koordinaten-Plot wechselt die Darstellung zum Scatterplot und setzt die Variablen des angeklickten Bezeichners sowie des nachfolgenden Bezeichners als Datenwerte für die x- und y-Achse fest. So kann der Anwender die Linien zwischen zwei Achsen im Parallel-Koordinaten-Plot genauer aufschlüsseln und diese als Punkte in einem zweidimensionalen Koordinatensystem auswerten. Dies ermöglicht eine detailliertere Analyse von Korrelationen zwischen den beiden Variablen. Mit dieser Funktion können Smartphone-Komponenten im Verhältnis zu Preis oder Bewertung übersichtlicher im Scatterplot untersucht werden.
- **Tree -> Starplot** : Ähnlich wie bei der Interaktion im Scatterplot, führt das Klicken auf einen Blattknoten im Baumdiagramm zur Starplot-Darstellung, wobei die Daten des ausgewählten Smartphones im älteren der beiden Starplots aktualisiert werden. Diese Interaktion bietet eine umfassende Übersicht über alle numerischen Daten eines angeforderten Modells. Auf diese Weise ermöglicht die Baumdarstellung zunächst eine grobe Analyse der Konkurrenten (Hersteller) und ihrer Modelle mit optionalen Filtermöglichkeiten. Im zweiten Schritt kann dann eine detaillierte Modellanalyse durchgeführt werden, die wertvolle Informationen über die Stärken und Schwächen der Konfiguration eines Smartphone-Modells liefert. Diese Erkenntnisse können direkt in den Planungs- und Entwicklungsprozess neuer, optimierter Smartphone-Modelle für den Markt einfließen.

- **Buttons Plot Switch** : Die Anwendung bietet eine einfache Interaktion, bei der mittels Buttons zwischen den verschiedenen Plots gewechselt werden kann. Diese Funktion erleichtert die Bedienung der Anwendung und ermöglicht einen schnellen Wechsel zwischen den Visualisierungen.

4 Implementierung

Während des Projekts wurde eine Visualisierungsanwendung in Elm entwickelt, die in mehrere Module unterteilt ist. Für jede der vier Visualisierungen wurde ein eigenes Modul erstellt, das alle wichtigen Funktionen für die Gestaltung und Interaktionen der jeweiligen Darstellung bereitstellt (Scatterplot.elm, ParallelCoordinates.elm, TreeView.elm, Starplot.elm). Zur Organisation und Integration aller Visualisierungen dient das Modul Main.elm, welches als Startpunkt des Elm-Programms fungiert. Es beinhaltet die Initialisierung des Modells (Model.elm), die Update-Funktionen für Interaktionen sowie die allgemeine Auswahl, welche Visualisierung dargestellt werden soll (View). Darüber hinaus werden in Main.elm die Daten aus der CSV-Datei in eine bestimmte Datenstruktur geparsed, die im Modul SmartPhoneType.elm definiert ist. Um diese Aufgaben zu erfüllen, importiert Main.elm alle zuvor genannten Module sowie zusätzliche Elm-Pakete.

Die aus der CSV-Datei geparsen Daten werden im Modell (siehe Abb. 1(a)) als eine Liste vom Typ Smartphone gespeichert (siehe Abb. 1(b)). Diese Liste enthält alle möglichen Attribute eines Smartphones, die aus der CSV-Datei extrahiert werden konnten, wie z. B. Preis, Bewertung und verschiedene Hardware-Details.

```
4  type alias Smartphone =
5      { id : Maybe String
6        , brand : Maybe String
7        , model : Maybe String
8        , price : Maybe Float
9        , rating : Maybe Float
10       , g5 : Maybe Bool
11       , nfc : Maybe Bool
12       , ir_blaster : Maybe Bool
13       , processor_name : Maybe String
14       , processor_brand : Maybe String
15       , num_cores : Maybe Float
16       , processor_speed : Maybe Float
17       , battery : Maybe Float
18       , fast_charging : Maybe Float
19       , memory : Maybe Float
20       , ram : Maybe Float
21       , screen_size : Maybe Float
22       , resolution : Maybe String
23       , refresh_rate : Maybe Float
24       , camera : Maybe String
25       , card : Maybe String
26       , os : Maybe String
27     }
28
```

(a) SmartPhoneType

```
7  type alias Model =
8      { data : List Smartphone
9        , error : Maybe String
10       , csvdata : String
11       , scatterplotOptions : ScatterPlotOption
12       , parallelPlotOption : ParallelPlotOption
13       , treeOption : TreeOption
14       , starplotOption : StarPlotOption
15       , plotVisible :
16         { scatterPlot : Bool
17           , parallelCoordinatesPlot : Bool
18           , treeView : Bool
19           , starPlot : Bool
20         }
21     }
```

(b) Modellausschnitt

Abbildung 1: SmartPhoneType und Modellausschnitt

Das Modell wird hauptsächlich zur Verwaltung der verschiedenen Zustände der Visualisierungs-

Interaktionen und zur Speicherung der Smartphone-Daten verwendet. Das Attribut `data` enthält die Liste der geparkten Smartphones. Die weiteren Attribute dienen der Zustandsverwaltung der Darstellung, wie zum Beispiel `scatterplotOptions` (für Scatterplot-Optionen) und `plotVisible` (um anzugeben, welche Plots aktuell dargestellt werden sollen).

Die `ScatterPlotOptions` (siehe Abb. 2(a)) enthalten Informationen zu den aktuellen Attribut- und Einheitenbezeichnern an den Achsen sowie eine Liste der Werte für diese Attribute.

Die `ParallelPlotOption` (siehe Abb. 2(d)) speichert Zustandsinformationen zur Reihenfolge der Achsen (Bezeichner, Einheiten und Werte) sowie zusätzliche Filteroptionen.

Die `TreeOption` (siehe Abb. 2(c)) enthält Informationen über die Zoom-Zustände, ausgewählte Filteroptionen und manuell entfernte Knoten.

Die `StarPlotOption` (siehe Abb. 2(b)) speichert Informationen zu den zwei aktuellen Modell-IDs für beide Starplots, sowie Eingaben für die IDs und den Zustand, welches Starplot als nächstes bei einer Interaktion mit anderen Plots überschrieben werden soll.

```

24 type alias ScatterPlotOption =
25   { attribute1 : String
26     , attribute2 : String
27     , att1List : List (Maybe Float)
28     , att2List : List (Maybe Float)
29     , unit1 : String
30     , unit2 : String
31   }

```

(a) ScatterPlotOption

```

58 type alias StarPlotOption =
59   { modelID1 : String
60     , modelID2 : String
61     , input1 : String
62     , input2 : String
63     , nextChange : Int
64   }

```

(b) StarPlotOption

```

42 type alias TreeOption =
43   { zoomOption : Zoom
44     , dropCount : Int
45     , inputDropCount : String
46     , selectedFilter : FilterOptions
47   }

```

(c) TreeOption

```

34 type alias ParallelPlotOption =
35   { orderedList : List (Smartphone -> Maybe Float)
36     , orderedLabels : List String
37     , orderedUnitLabels : List String
38     , selectedFilter : FilterOptions
39   }

```

(d) ParallelPlotOption

Abbildung 2: Teile vom Model

Die `FilterOption` (siehe Abb. 3), die bei den `ParallelPlotOption` und `TreeOption` verwendet wird, enthält Zustände zu den ausgewählten Filtern. Dazu gehören der gewählte Hersteller und das Vorhandensein bestimmter Smartphone-Funktionen wie 5G oder NFC.

Das letzte Modul `ExtendedFunctions.elm` enthält zusätzliche erstellte Funktionen wie `map10` und `listmap10`. Diese ähneln den in Elm vorhandenen Funktionen, bieten jedoch Unterstützung für mehr Parameter. Die Funktion `map10` ist vergleichbar mit `Maybe.map`, und `listmap10` ähnelt `List.map`, nur dass beide Funktionen jeweils 10 Parameter verarbeiten können. Da diese Funktio-

```

50 type alias FilterOptions =
51     { brand : String
52       , g5 : String
53       , nfc : String
54       , ir_blaster : String
55     }

```

Abbildung 3: FilterOption

nen die Möglichkeiten der Sprache Elm erweitern und für die Parallelkoordinaten-Visualisierung benötigt wurden, um Attribute und Listen auf eine Funktion abzubilden, wurden sie in einem separaten Modul bereitgestellt.

Dank der Übungen gestaltete sich die Umsetzung einiger Plots einfacher als ursprünglich erwartet. Der Aufwand für den Scatterplot und den Parallelen-Koordinaten-Plot war relativ gering, da größere Teile aus den vorhandenen Übungsaufgaben übernommen werden konnten und nur kleinere Anpassungen, wie etwa neue Interaktionen oder Designänderungen, nötig waren.

Anders verhielt es sich bei den beiden anderen Plots (Tree und Starplot), die von Grund auf geplant und recherchiert werden mussten. Für den Tree-Plot war es erforderlich, nach neuen Paketen zu suchen, um die Navigation durch den Baum für den Anwender so einfach wie möglich zu gestalten und ein ansprechendes Design zu erzielen. Beim Starplot war zwar keine Einarbeitung in neue Pakete nötig, jedoch mussten aufwendigere Designs erstellt, Interaktionen hinzugefügt und das grundlegende Gerüst für den Starplot komplett neu entwickelt werden. Insgesamt lässt sich sagen, dass die Plots, die in den Übungen behandelt wurden, mit weniger Aufwand umgesetzt werden konnten als der Tree-Plot mit einem anderen Paket und der Starplot.

Im Folgenden werden einige wichtige Elm-Pakete, die bei der Implementierung verwendet wurden, genannt und kurz deren Einsatz erläutert:

- **BrianHicks/elm-csv** und **NoRedInk/elm-json-decode-pipeline**: Diese Pakete wurden für das Parsen der CSV-Daten verwendet.
- **elm-community/typed-svg**: Diente zur Erstellung von SVG-Grafiken in allen Plots.
- **elm-community/list-extra**: Ermöglichte das einfache Ändern der Reihenfolge von Elementen in einer Liste: insbesondere genutzt im Parallelen-Koordinaten-Plot, um die Achsen zu vertauschen.
- **gampleman/elm-visualization** und **gampleman/elm-rosetree**: Eingesetzt zur Darstellung des Baums.
- **folkertdev/one-true-path-experiment**: Verwendet für die Darstellung von Linien (Pfeilen) in den Parallelen-Koordinaten- und Starplots.

- **myrho/elm-round**: Zum Runden von Float-Zahlen auf zwei Dezimalstellen.
- **avh4/elm-color**: Diente zur Farbgestaltung von SVG-Elementen.

Die Elm-Anwendung wurde in JavaScript (main.js) übersetzt und in eine HTML-Seite (index.html) integriert. Um das Design der Webseite anzupassen, wurden CSS-Klassen verwendet, die einzelne Teile der Elm-Anwendung verändern. Diese CSS-Design-Implementierungen sind in einer separaten CSS-Datei (style.css) gespeichert, die ebenfalls in die HTML-Seite eingebunden wird. Alle diese Dateien befinden sich im Ordner „public“. Die HTML-Seite präsentiert die erstellte Visualisierungsanwendung.

5 Anwendungsfälle

5.1 Anwendung Scatterplot

5.2 Anwendung Parallel-Koordinaten-Plot

5.3 Anwendung Tree

5.4 Anwendung Starplot

6 Verwandte Arbeiten

Führen sie eine kurze Literatursuche in der wissenschaftlichen Literatur zu Informationsvisualisierung und Visual Analytics nach ähnlichen Anwendungen durch. Diskutieren sie mindestens zwei Artikel. Stellen sie Gemeinsamkeiten und Unterschiede dar.

7 Zusammenfassung und Ausblick

Fassen sie die Beiträge ihre Visualisierungsanwendung zusammen. Wo bietet sie für die Personen der Zielgruppe einen echten Mehrwert.

Was wären mögliche sinnvolle Erweiterungen, entweder auf der Ebene der Visualisierungen und/oder auf der Datenebene?

Anhang: Git-Historie

Datum	Commit
2024-05-29	InfoVisualProjekt erstellt
2024-05-29	install more elm packages
2024-06-11	csv Datei
2024-06-11	Csv Datei kopiert
2024-06-11	get csv data test
2024-06-11	get csv data gitlab test
2024-06-11	get csv data index gitlab test
2024-06-11	url kommentar hinzugefügt
2024-06-19	einlesen korrigiert
2024-06-23	Scatterplot Basic hinzugefügt und einlesen geändert
2024-06-23	Parser korrigiert
2024-06-23	Buttons aus anderer Elm Datei: update ausprobiert
2024-06-24	Html.select ausgetestet
2024-06-25	css ausprobiert
2024-06-25	update main.js
2024-06-26	Scatterplot Attribute wechselbar
2024-06-26	padding angepasst
2024-06-26	Kommentare entfernt und Types imported
2024-06-26	Vorbereitung Parallel Koordinaten
2024-07-23	Grundgerüst ParallelKoordinaten erstellt und neu Model.elm für bessere Ordnung
2024-07-24	Grundgerüst Tree erstellt, ExtendedFunctions für map10 und listmap10
2024-08-09	StarPlot begonnen, Change Position Parallel, Buttons für wechseln Plot
2024-08-13	Starplot fertig, input ID, Interaktion: Scatterplot -> Starplot, Tree -> Starplot
2024-08-13	ParallelPlot Achsen Beschriftung unterhalb nicht sichtbar behoben
2024-08-13	Starplot vergleich (2 Starplots mit Änderungsfunktion), Änderung an Main, Model und Starplot
2024-08-14	Interaktion Parallel und ScatterPlot, Fehler behoben bei Scatterplot mit Attribut Sichtbarkeit in select, Filter Optionen Tree umgesetzt
2024-08-15	Filter ParallelCoordinates, Tree Knotengrößen Anpassung und Text Umbruch in Leaf angepasst
2024-08-15	Css bei allen Plots, etwas Gestaltung der Webseite (Caption, Menü für Plots und Beschreibung), Anpassung Submit Button update Fehlerbehandlung
2024-08-16	RAM Scatterplot Fehler behoben, Einheiten hinzugefügt, Plot-Beschreibung hinzugefügt, RAM Daten fehlerhaft in csv für ein paar Smartphones angepasst
2024-08-20	elm.json test-dependencies Eintrag entfernt
2024-08-21	Hints hinzugefügt bei Starplot in Beschreibung
2024-08-xx	Code aufgeräumt (Kommentar-Code entfernt)

Literatur

- [1] OpenAI. *ChatGPT*. URL: <https://openai.com/blog/chatgpt> (besucht am 30.09.2023).
- [2] Nitya verma. *Smartphone Dataset (Cleaned)*. 2024. URL: <https://www.kaggle.com/datasets/nityaverma19/smartphone-dataset-cleaned>.

Anmerkung

Verwendete Hilfsmittel:

ChatGPT[1], für die sprachliche und grammatikalische Verbesserung meiner geschriebenen Texte. Der Inhalt meines Berichtes entstammt von mir und hat sich durch die Verwendung des Hilfsmittels nicht geändert.