Zürcher Hochschule
für Angewandte Wissenschaften

School of
Engineering

Projektarbeit Web-Scale Datenanalyse auf einer Big Data Appliance

Autor:

Manuel Hunkeler

Betreuende Dozenten:

Prof. Dr. ès sc.
Martin Braschler

Dr.
Thilo Stadelmann

19. Dezember 2012

Zusammenfassung

Das Projekt „Web-Scale Datenanalyse auf einer Big Data Appliance“ umfasst die Analyse von Daten der Firma Codecheck, das Laden der Daten in ein IBM PureData System for Analytics (Netezza System), den Bau eines eigenen Software-Prototypen und die Auswertung und Evaluation der Daten.
Die rohen Daten liegen in Form von Apache-Standart-Logfiles vor. Bei der Betrachtung der Rohdaten wird schnell klar, dass nicht alle Einträge für die weitere Verarbeitung verwendet werden konnten.
Mit einem aus mehreren Java Scripts bestehenden Software-Prototypen werden die Daten selektiert, verarbeitet und erweitert. Für die Erweiterung der Daten wird das externe „Data Science Toolkit“ verwendet. Um das immense Datenvolumen möglichst effizient handhaben zu können, werden die aufbereiteten Daten in das Netezza System portiert.
Für die Auswertung und Visualisierung der Daten werden die Programme IBM SPSS und IBM Cognos Report Studio verwendet. Zusätzlich werden, für weitere Auswertungen, SQL-Statements entwickelt, die direkt auf dem Netezza System ausgeführt werden. Die SQL-Statements generieren eine Resultatetabelle, mit dessen Hilfe man durch Webtools wie „CartoDB“ und „Wordle Cloud“ weitere Visualisierungen realisieren kann.
Durch die vorangehende Datenselektion und die schnellen, durch das Netezza System ermöglichten Abfragen sind der individuellen Datenanalyse kaum mehr Grenzen gesetzt. Die bereits angesprochenen Webtools lassen einige prägnante Interpretationen zu.
Schlussendlich kann man jedoch sagen, dass das Netezza System in erster Linie für Data Warehouse optimiert ist und sich nicht für Data Mining eignet.

Erklärung betreffend das selbständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Ort, Datum:

……

Unterschriften:

……

……..

……

Das Original dieses Formulars ist bei der ZHAW-Version aller abgegebenen Projektarbeiten zu Beginn der Dokumentation nach dem Titelblatt mit Original-Unterschriften und -Datum (keine Kopie) einzufügen.

Inhaltsverzeichnis

1. Einleitung

Hintergrund

Die Einleitung der Kurzbeschreibung dieser Arbeitet lautet wie folgt:

> *"Es findet im Moment, ein Paradigmenumbruch in der Speicherung sehr grosser Mengen strukturierter Daten statt. Die Berechtigung traditioneller relationaler Datenbanken wurden intensiv hinterfragt. Gleichzeitig ist der Umgang mit extrem grossen Datenmengen ("Big Data") eines der neuen spannenden Themen der Informatik“* (Braschler & Stadelmann, 2013).

Da physikalischer Speicher immer günstiger wird und die Hardware Ressourcen der Rechensysteme immer effizienter werden, gibt es für Unternehmen immer mehr Anreiz grosse Mengen von Daten zu sammeln und diese auszuwerten. Das nennt man auch die Analyse von Big Data. Heute erhebt fast jedes grosse Unternehmen der Schweiz, welches Online tätig ist, Daten und verwendet diese für unternehmensspezifische Analysen. Diese Auswertungen helfen dem Management eines Unternehmens sehr oft für die Entwicklung von Marketingstrategien. Man versucht aber auch das Verhalten des Onlineprotalnutzers zu verstehen, um benutzerfreundlicher zu werden oder zum erschliessen von neuen Geschäftsfeldern.
Gemäss der Online Publikation "Analytics: Big Data in der Praxis" vom IBM Institute for Buisness Value wird Big Data grundsätzlich mit der "3V" Formel (Volume, Variety, Velocity) charakterisiert. Dies beschreibt die 3 Grundkirterien für Big Data Masse, Vielfalt und Geschwinkigkeit (vgl. Schroeck, et al., 2012, p. 4).
Auf das Thema Big Data wird im gleichnamigen Absatz auf der Seite 3 näher eingegangen.
Diese Arbeit folgt nicht ganz den 3 Grundkriterien für Big Data, denn die Vielfalt der Daten beschränkt sich auf strukturierte Daten. Die Daten von Codecheck sind Apache Serverlog Files die immer die gleiche Struktur aufweisen. Das Kriterium der grossen Masse an Daten wird durch mehrere Millionen Datenbank Zeileneinträge erfüllt. Das letzte Kriterium der schnellen Verarbeitung der Daten wird mittels einem IBM PureData System for Analytics realisiert. Dieses System wird im Absatz IBM PureData System auf der Seite 5 näher erläutert.

Big Data

Gemäss der IBM Webseite fallen heute jeden Tag etwa 2,5 Trillionen Bytes an Daten an. Diese Daten werden hauptsächlich durch die Social-Media-Netzwerke, digitale Bilder oder Videos, Daten von Kauftransaktionen, GPS-Signalen von Mobiltelefonen, Wettersensoren sowie diverse andere Quellen bei den grosse Mengen von Daten anfallen, generiert. Alle diese Daten werden unter dem Begriff Big Data zusammengefasst (vgl. IBM, 2013).
Big Data wird grundsätzlich durch die drei Grundkriterinen Masse der Daten, Vielfalt und der Geschwindigkeit der Auswertung charakterisiert. Für die Auswertung der Daten in Unternehemen kommt noch eine zusätzliches Kriterium hinzu. Nämlich die Richtigkeit der Daten. IBM spricht hier von den vier relevanten Dimensionen der Big Data.

Masse der Daten

Die Dimension der Masse bezieht sich auf die Menge der Daten, die von Unternehmen oder Forschungsanstalten verwendet werden, um Analysen zu tätigen. Die Fülle dieser Daten wächst kontinuierlich und mit rasanter Geschwindigkeit. Je nach Branche oder Firma spricht man bei Big Data von Datenmengen im Bereich von Terabytes (10^{12}) bis Zetabytes (10^{21}) (vgl. Schroeck, et al., 2012, p. 6).

Vielfalt der Daten

Die Vielfalt der Daten umfasst die Quellen und die Struktur der Daten. Je nach Quelle können diese Daten in strukturierter, unstrukturierter oder semistrukturierter Form vorliegen. Diese Dimension hat die Integration von verschiedenen Datenquellen, Datenformaten und Datentypen in ein komplexes System zum Ziel (vgl. Schroeck, et al., 2012, p. 6).

Geschwindigkeit der Verarbeitung

Die Masse der zu analysierenden Daten werden in Echtzeit generiert. Das heisst, dass das Volumen der Daten stetig zunimmt. Es gibt zeitkritische Prozesse, wie die Betrugserkennung in Echtzeit, die einem Unternehmen nur einen Vorteil bringen, wenn diese System die Daten in nützlicher Frist analysieren und auswerten können. Dazu wurden leistungsstarke Systeme entwickelt, wie zum Beispiel das IBM PureData System for Analytics, das speziell für die Auswertung grosser Daten optimiert wurde (vgl. Schroeck, et al., 2012, p. 6).

Richtigkeit der Daten

Die Qualität, respektive die Zuverlässigkeit der Daten ist im Umgang mit Big Data stets ein Thema. Es gibt Bereinigungsmethoden, die die Qualität der Daten sichert. Es ist aber auch mit den besten Bereinigungsmethoden nicht möglich, zuverlässige Vorhersagen über das Wetter oder die Wirtschaftsentwicklung zu machen. Das Management vieler Unternehmen will riskante Faktoren erkennen und diese gezielt in ihre Entscheidungen einfliessen lassen (vgl. Schroeck, et al., 2012, p. 6).

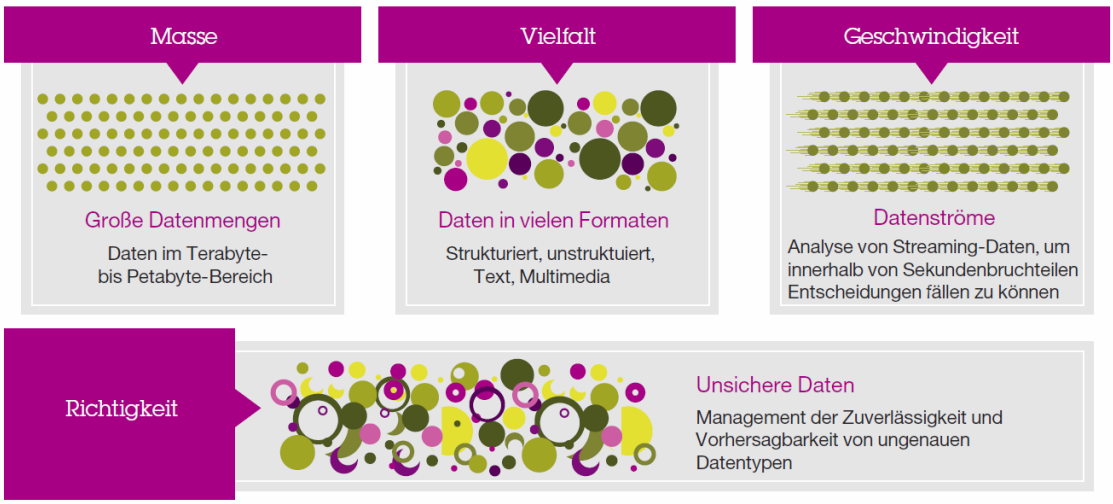


Abbildung 1 Big Data Dimensionen (Schroeck, et al., 2012, p. 6)

IBM PureData System

Das IBM PureData System for Analytics basiert auf der Netezza Technologie. Diese ist eine leicht limitierte SQL-Datenbank für professionelle Analysen. Dieses System bietet die Möglichkeit komplexe Algorithmen in kürzester Zeit auszuführen sowie die Parallelverarbeitung von mehreren Petabytes an Daten zu realisieren.
Bei der Netezza Technologie werden Field Programmable Gate Arrays (FPGA) für eine Hochleistungsverarbeitung von Analysen verwendet. Die Hardware- und Softwarefunktionen sind speziell für Hochleistungsanalysen entworfen, integriert und optimiert worden (vgl. IBM, 2013).

Die wichtigsten Vorteile des IBM PureData Systems sind:

- Es ist 10 bis 100 mal schneller als herkömmliche Systeme
- Ladegeschwindigkeiten von bis zu 5TB pro Stunde
- minimaler Optimierungs- und Verwaltungsbedarf
- hohe Ausfallsicherheit
- Es bietet komplexe Algorithmen und umfassende Bibliotheken von integrierten, mathematischen und statistischen Analysen (vgl. Serwise AG / Big Data, 2013).

Ausgangslage

Dieses Projektarbeit wird in Zusammenarbeit mit dem Institut für angewandte Informationstechnologien (InIT), der Firma Serwise AG in Winterthur und der Firma Codecheck aus Zürich realisiert.
Die Firma Serwise AG stellt Rechenzeit und personelle Unterstützung für ihr Netezza System zur Verfügung.
Die verwendeten Daten liegen in der Form Server-Logfiles vor und werden von der Firma Codecheck geliefert.

Serwise Gruppe

Die Serwise Gruppe setzt sich aus der Service AG Winterthur, der Service Deutschland GmbH, Studentenlab AG und der Infra Support GmbH zusammen. Der Name Serwise setzt sich aus der Verschmelzung der beiden englischen Wörtern "service" und "wise" zusammen, was soviel wie kluger Service bedeutet.
Die Serwise Gruppe beschäftig über zwanzig Mitarbeiter darunter acht bis zwölf Studenten. Serwise hat sich auf die Umsetzung von Projekten, in den betriebswirtschaftliche Expertisen mit fundiertem IT-Umsetzungs-Know-how gefordert ist, spezialisiert. Sie bieten Beratungen für die Erarbeitung von Strategien der Organisations- und Prozessberatung, der Optimierung von Controlling und Rechnungswesen, bis hin zur Evaluation, Implementierung, Anpassung und Eigenentwicklung softwaretechnischer Lösungen.
Im August 2013 baute die Firma ihr „Business Analytics“ mit dem Fokus auf das Thema Big Data aus (vgl. Serwice AG, 2013).
In der Zusammenarbeit mit Mitarbeitern der ZHAW, IBM, AXA Winterthur, Netmodul, Infra Support und der StudentenLab AG baute die Service AG ein IBM PureSystem so um, dass es in einem Anhänger Platz hat. Mit diesem mobilen Big Data Center können potentielle Kunden am eigenen Standort von der Leistungsfähigkeit des IBM PureSystem und der Big Data Thematik überzeugt werden.
Weite Informationen über die Serwise AG und die Big Data Analytics Road Show gibt (Serwice AG, 2013) und (Serwise AG / Big Data, 2013).

Abbildung 2 Mobiles PureSystem (Neue Zürcher Zeitung, 2013).

Codecheck Datenbank

Codecheck ist ein Online-Produkthandbuch. Die Plattform sammelt Fachinformationen zu den Inhaltsstoffen eines Produktes und Expertenmeinungen, die zur Transparenz auf dem Markt beitragen sollen.
Mit dem Mobile-App von Codecheck können Konsumentinnen und Konsumenten Barcode eines Produktes direkt im Laden scannen und in der Codecheck Datenbank suchen. Ist der jeweilige Artikel in der Datenbank erfasst, werden dem Benutzer diverse Informationen wie Tests und Bewertungen der Inhaltsstoffe, technische Angaben, Nährwerte sowie Kommentare von anderen Codecheck Benutzern angezeigt. Für kritische Konsumentinnen und Konsumenten gibt es auch die Funktion „Alternative-Finden“. Diese kann nach alternativen Produkten suchen, bei denen auf Wunsch des Benutzers auf unerwünschte oder kritische Inhaltsstoffe verzichtet werden. Um die Qualität der Produktinformationen zu sichern, arbeitet Codecheck mit diversen und unabhängigen Organisationen zusammen, die sich unteranderem der Produkttansparenz widmen (vgl. Codecheck, 2013).

Zielsetzung

Das Ziel der Projektarbeitet lautet im Wortlaut:

„Die Aufgaben der Studierenden:

- *Die Analyse der Daten der Firma Codecheck (Produktdatenbank, und assoziierte Daten) und Laden in die Netezza Appliance.*
- *Festlegung der zu ermittelnden Information auf Basis Gesprächen mit der Firma Codecheck*
- *Bau von SQL-Abfragen auf der Datenbasis.*
- *Bau eines Prototypen, welcher die aggregierten Resultate aufbereitet und darstellt.*
- *Evaluation der Resultate“* (Braschler & Stadelmann, 2013).

Aufbau der Arbeit

Der Aufbau dieser Arbeit folgt keiner linearen Struktur, kann aber in die folgenden Teile gegliedert werden.

Verwandte Arbeiten
Dieses Kapitel beinhandelt ähnliche Arbeiten oder Projekte, die für die Entscheidung späterer Analysen von Nutzen waren.

Resultat
Hier werden die Resultate in Form von Grafiken und Bildern Illustriert.

Lösungsfindung
Welches Vorgehen soll gewählt werden, um die eigenen Ideen zu realisieren? Es wird ein Grobkonzept erarbeitet, das für die spätere Implementierung als Vorlage dient.
Das Grobkonzept hält sich an die zwei Kriterien "was" und "warum".

Implementierung
Dieser Abschnitt beschreibt die technische Implementierung des Software-Prototypen. Es wird konkret beschrieben "wie" diese Lösung implementiert wurde.

Evaluation der Resultate
In diesem Kapitel wird auf die Grafiken der Resultate näher beschrieben und bewertet.

2. Verwandte Arbeiten

In diesem Kapitel werden ähnliche oder verwandte Arbeiten beschrieben, die für die Ideenfindung und der Entwicklung der angestrebten Lösung einen signifikanten Einfluss hatten.

Die NZZ- Sommer-Serie "Schweizer Karten"

Die Neue Zürcher Zeitung in Zusammenarbeit mit dem Zürcher Design Studio Interactive Things veröffentlichte im Sommer 2013 eine Reihe von zwanzig Schweizer Karten.
Sie verarbeiteten dabei Daten von topographischen, gesellschaftlichen, politischen und wirtschaftlichen Phänomene so, dass diese auf eine direkte Weise oder mittels Chiffren auf einer interaktiven Schweizer Karte visualisiert werden konnten. Zusätzlich wurden die Karten einmal pro Woche mittels einer vertieften Reportage über den entsprechenden Sachverhalt erweitert.
Die Urheber dieser Serie wollten dabei den Blick der Leserschaft für Zusammenhänge und der Entwicklung schärfen, aber dadurch keine neuen Weisheiten verbreiten.

Auf dieser Karte wurden die durchschnittlichen Stromtarife einer 4-Zimmer-Wohunung in jeder Gemeinde der Schweiz visualisiert. Dabei konnten klare Preistrends erkannt werden.

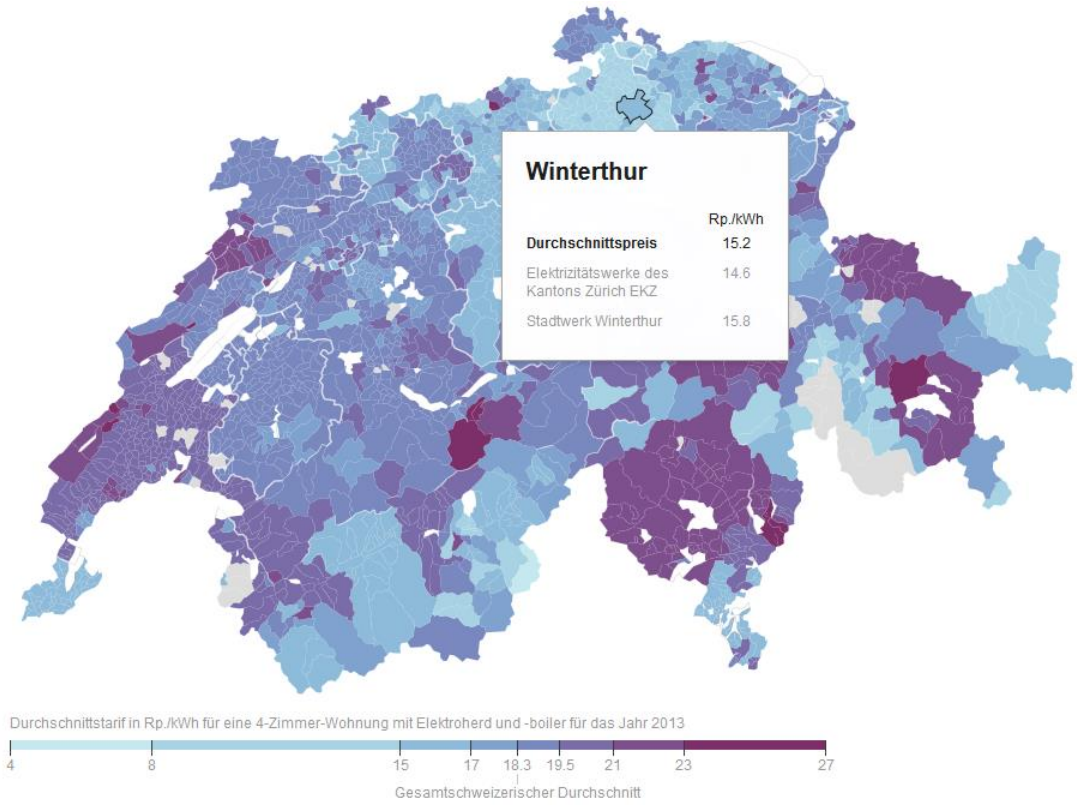


Abbildung 3 Schweizer Strompreise (Neue Zürche Zeitung, 2013).

Weitere Karten und Informationen zur Sommer-Serie "Schweizer Karten" gibt (Neue Zürcher Zeitung, 2013).

Nike Data Viz

Das Nike Data Viz ist ein kleines Projekt, das von Mark Cieliebak und vier Kollegen an der Open Data Conference realisiert wurde. Das Projektteam hatte die Aufgabe innerhalb einer Stunde "etwas Interessantes" mittels einem Datensatz von der Sportartikelfirma Nike zu machen. Dabei entschieden sie sich, die weltweiten Verkaufsaktivitäten und die Produktionsstandorte auf einer Weltkarte zu visualisieren. Sie verwendeten dabei das online "CartoDB" Tool, das für eine kleine Datenmenge bis 5 Megabytes frei genutzt werden kann.

Die Karte unten zeigt das Resultat dieser Arbeit. Die gelben bis roten Punkte stellen die Anzahl NikeShops an einem bestimmten Ort dar. Die Grösse der blauen Punkte visualisiert, wo und wie viele Arbeiter in einer Nike Produktionsstätte arbeiten.
Die Botschaft hinter dieser Kartendarstellung wird auch schnell ersichtlich. Man kann sehr gut erkennen, wo die meisten dieser Sportartikel hergestellt werden und wo sie meistens verkauft werden.

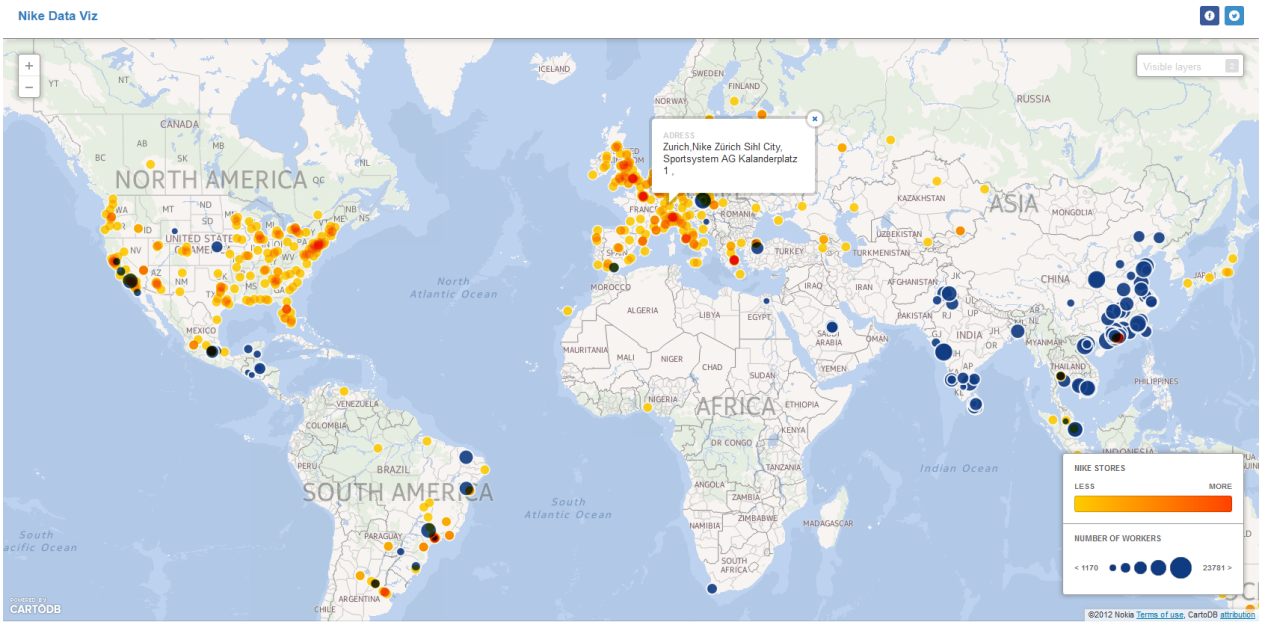


Abbildung 4 Nike Data Viz Karte (Cieliebak, 2013).

Die Nike Data Viz Arbeit hatte auf die Realisierung dieser Arbeit einen wesentlichen Einfluss, denn das CartoDB Tool bietet eine grosse Palette von Visualisierungsmöglichkeiten und ist dabei sehr einfach zu handhaben.
Des Weiteren ist es Open-Source, sodass ein eigner Server der ZHAW für diesen Dienst aufgesetzt werden konnte. Dies ermöglichte es, dieses Tool ohne grossen Kostenaufwand für dieses Projekt verwenden können.
Im Abschnitt CartoDB wird das Tool genauer erklärt und weitere Informationen zur Erstellung eines eignen CartoDB Servers erläutert.

3. Resultat

Analyse der Daten

Die Daten von der Codecheck Datenbank lagen in Form von Apache-Standard-Logfiles (im folgenden "Logfiles" genannt) vor. Codecheck lieferte für jeden Tag ein einzelnes Text File, das etwa 1 bis 1.5 Millionen Zeileneinträge beinhaltete. Um diese Menge von Daten zu handhaben ist es unumgänglich, diese zu verarbeiten und in einer Datenbank zu speichern. So wurde entschieden, dass die Verarbeitung der Logfiles und die Aufbereitung der Datenbank durch den Software-Prototypen realisiert werden sollte. Diese fertige Datenbank sollte dann exportiert und auf das Netezza System portiert werden. Die Auswertung der Daten soll dann mit dem Analysetool IBM SPSS und dem IBM Cognos Report Studio gemacht werden.

Apache-Standard-Logfiles

Die Apache Standard Log Zeilen können in folgende Attribute unterteilt werden:

Bedeutung	Wert der Beispiel Zeile	Erläuterung
IP-Adresse	88.76.206.73	IP-Adresse des anfordernden Hosts
Unbelegt	-	Nicht belegt
Wer	-	Nutzernamen aus einer HTTP-Authentifizierung sonst „-„
Wann	[15/Sep/2013:00:00:06 +0200]	Zeitstempel (Datum, Uhrzeit, Zeitverschiebung)
Was	"GET/static/0eb73f38/moreLess.png HTTP/1.1"	Anforderung eines .png Bildes und das Übertragungsprotokoll (Request)
OK	200	HTTP-Statuscode (200 = Erfolgreiche Anfrage)
Wie viel	322	Menge der gesendeten Daten in Bytes, sonst (z.B. bei Umleitung) „-„
Woher	"http://www.codecheck.info/kosmetik_gesundheit.kat"	Von welcher Internetseite (URL) wird angefordert (Referer)
Womit	Mozilla/5.0 (iPhone; CPU iPhone OS 6_1_3 like Mac OS X)	Mit welchem Browser / Betriebssystem/ Oberfläche

Tabelle 1: Apache Standard Log Zeile (vgl. Wikipedia, 2013).

Prototyp

Es wurde kein Prototyp für die Darstellung der Resultate erstellt, sondern ein Prototyp für die Verarbeitung und Erweiterung des Datensatzes konzipiert. Dieser Prototyp wird im Kapitel „Vorgehen“ auf der Seite 29 näher erklärt.

Darstellung der Daten

IBM SPSS und IBM Cognos Auswertungen

Für die Auswertung und Darstellung der Daten wurde zuerst das Programm IBM SPSS verwendet. Leider verursachte dieses Programm viele Fehler, die nicht in angemessener Zeit gelöst werden konnten. Jedoch konnten einfache Diagramme mit diesem Tool erstellt werden. Für die Erstellung weiterer Diagramme wurde zusätzlich noch das IBM Cognos Programm verwendet.

Analyse der verwendeten Geräte

Diese Analysen visualisieren mit welchen Geräten am meisten auf die Codecheck Datenbank zugegriffen wurde.

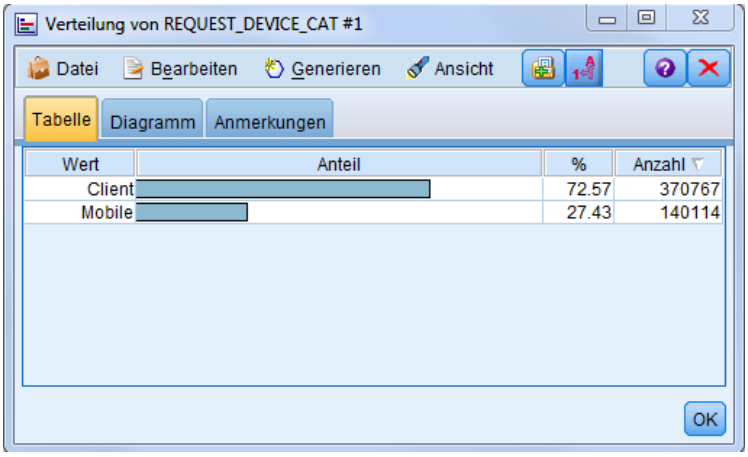


Abbildung 5 SPSS Grafik Vergleich Mobil / Client Geräte.

Anzahl Anfragen	Client	Mobile
Germany	215,812	58,633
Austria	21,126	6,301
Switzerland	13,559	6,091

Abbildung 6 Cognos Grafik Anzahl Anfragen bezogen auf Länder.

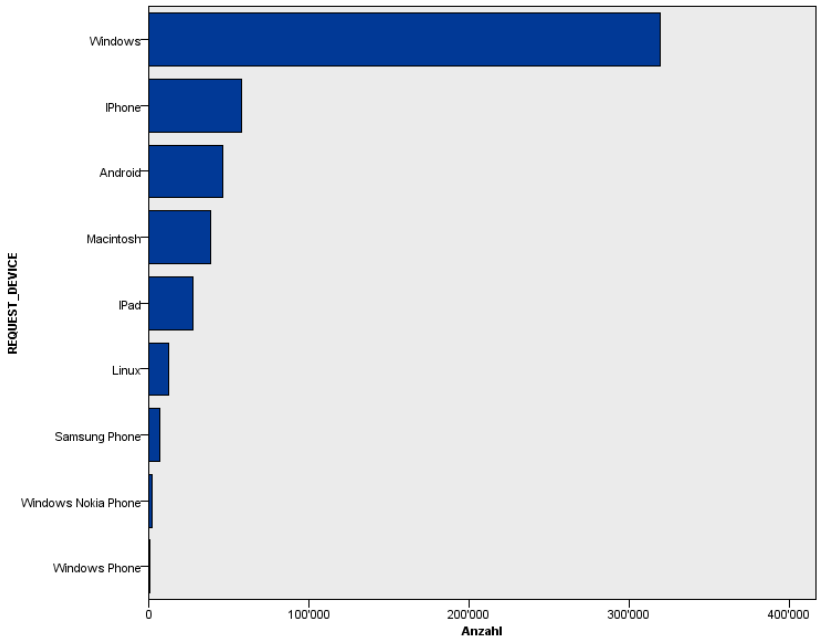


Abbildung 7 SPSS Grafik der verwendeten Geräte.

Analyse der Codecheck Nutzung bezogen auf Länder

In dieser Auswertung soll die Verteilung der Nutzung von Codecheck auf verschiedene Länder dargestellt werden.

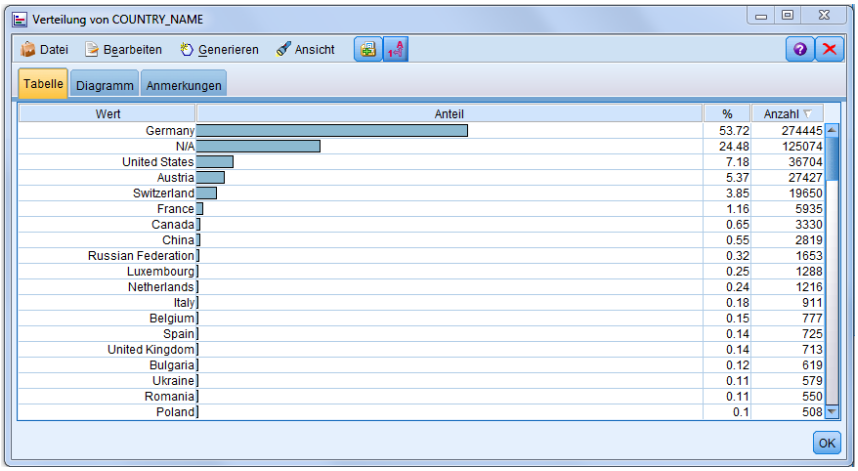


Abbildung 8 SPSS Verteilung der Anfragen auf Länder.

Top 10 Counrtires (by # requests)

Germany	274,445
N/A	125,074
United States	36,704
Austria	27,427
Switzerland	19,650
France	5,935
Canada	3,330
China	2,819
Russian Federation	1,653
Luxembourg	1,288

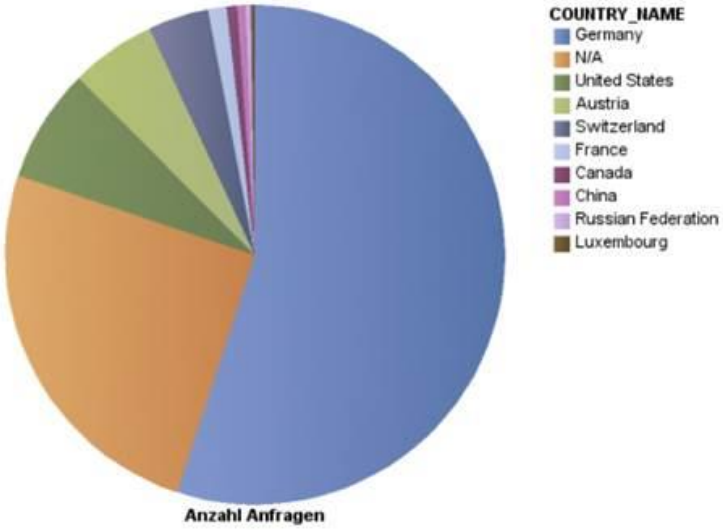
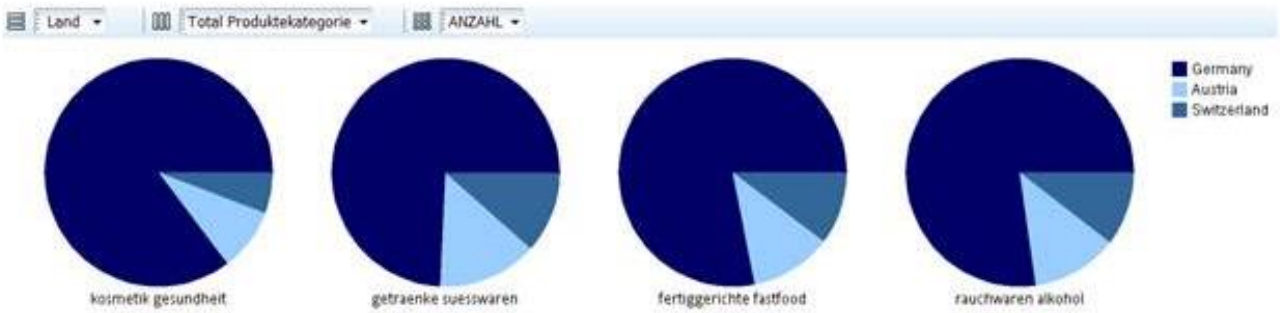


Abbildung 9 Cognos Grafik Kuchendiagramm Verteilung der Anfragen auf verschieden Länder.

Analyse der beliebtesten Produktkategorien

In dieser Analyse wurden die am meisten angefragten Produktkategorien, nämlich Kosmetik und Gesundheit, Getränke und Süsswaren, Fertiggerichte und Fast Food, sowie Rauchwaren und Alkohol ausgewertet. Dabei beschränkte man sich auf die Anfragen der Länder Deutschland, Österreich und die Schweiz. Ziel dieser Auswertung war es, die Häufigkeit der Anfragen der verschiedenen Produktkategorien zu visualisieren.
In einem ersten Schritt wurde mit dem IBM Cognos ein Diagramm erstellt, das die Anzahl an Aufrufen pro Land und Produktkategorie in verschieden Kuchendiagrammen darstellen sollte.



	Total Produktekategorie	kosmetik gesundheit	getraenke suesswaren	fertiggerichte fastfood	rauchwaren alkohol
Total von Land	102'866	85'722	14'188	1'904	1'052
Germany	85'912	73'080	10'533	1'487	812
Austria	10'068	7'711	2'007	220	130
Switzerland	6'886	4'931	1'648	197	110

Abbildung 10 Anzahl Aufrufe pro Land per Produktkategorie.

In einem zweiten Schritt wurde ein SQL-Statement geschrieben, welches die Anzahl der angefragten Produktkategorien pro Stadt oder Gemeinde summiert und die Summe in einem zusätzlichen Datenbank Attribut speichert.
Um nun die relative Anzahl an Aufrufen einer Produktkategorie im Verhältnis zur Bevölkerung zu bekommen, wurde ein zusätzliches SQL-Statement geschrieben, welches das Attribut der summierten Produktkategorien durch die Bevölkerungszahl dividiert und es auf einen darstellbaren Wert skaliert. Die SQL-Statements werden im Kapitel Implementationen auf der Seite 39 genauer erläutert.

Nach dem Ausführen der beiden SQL-Statements, direkt auf dem Netezza System, konnten die gerechneten Werte zusammen mit den Geo-Daten in einer CSV Datei exportiert werden.
Diese konnte dann in das CartoDB Webtool importiert werden und für eine Kartendarstellung verwendet werden. Es wurde dabei darauf geachtet, dass keine Codecheck spezifischen Daten auf das Webtool geladen wurden, sodass die Geheimhaltungsvereinbarung nicht verletzt wurde. Bei der Kartendarstellung wurden die Produktkategorien mit verschieden Layer visualisiert.

Die Karten können unter (Hunkeler, 2013) abgerufen werden.

Abbildung 11 CartoDB Visualisierung der Kategorie Kosmetik und Gesundheit.

Abbildung 12 CartoDB Visualisierung der Kategorie Fertiggerichte und Fast Food.

Abbildung 13 CartoDB Visualisierung der Kategorie Getränke und Genussmittel.

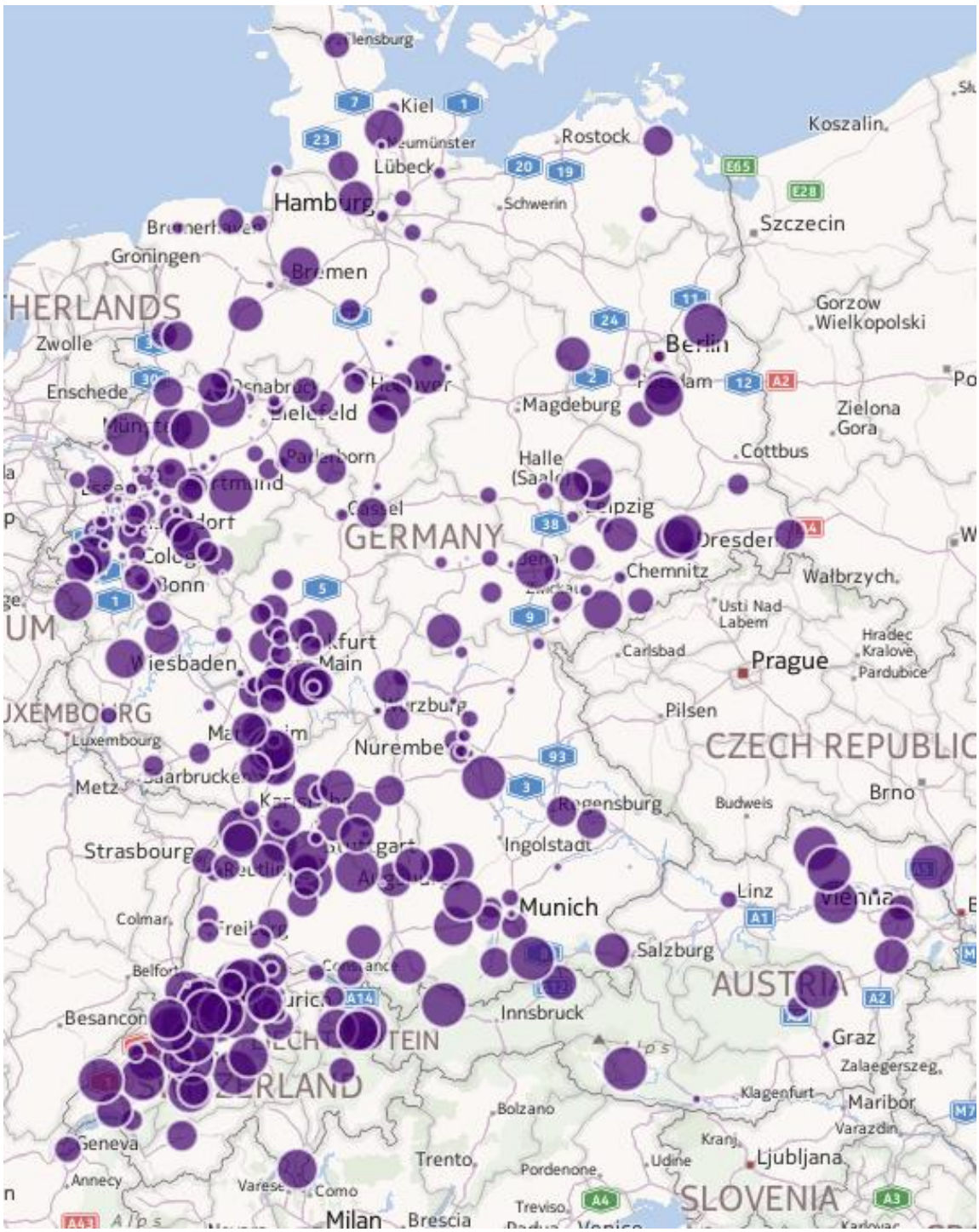


Abbildung 14 CartoDB Visualisierung der Kategorie Rauchwaren und alkoholische Getränke.

Darstellung der Produkte mit Wordle

Zusätzlich zu dem SPSS und Cognos Programm und den SQL-Abfragen wurde noch das Webtool Wordle verwendet, um eine sogenannte „Wordle Cloud“ zu generieren. Dabei wurden die Produkt Anfragen in den Ländern Deutschland, Österreich und der Schweiz visuell dargestellt.

Abbildung 15 ersten 100'000 Produktanfragen in Deutschland.

j

Abbildung 16 ersten 100'000 Produktanfragen in Österreich.

Abbildung 17 ersten 100'000 Produktanfragen in der Schweiz.

4. Lösungsfindung

Grobkonzept

Da die vorgegebene Aufgabenstellung sehr offen formuliert war, wurden auf verschiedene Dokumente wie ein detailliertes Pflichtenheft verzichtet. Das gab mir die Möglichkeit eigene Gedanken über dieses Projekt zu machen und einen eigenen passenden Lösungsweg zu entwickeln.

Da ich keine Erfahrungen im Umgang mit der Auswertung von grossen Mengen von Daten hatte, musste ich mich zuerst in die Welt der Datenanalyse einarbeiten.
Es gab weder anderen, noch identische Vorjahresprojekte oder Vorarbeiten. So wurde mir schnell klar, dass die Funktionen des zu entwickelnden Software-Prototypen von Grund auf neu implementiert werden musste.
Programme wie Splunk und httpAnalyser würden die Verarbeitung dieser grossen Mengen von Daten automatisch übernehmen. Die Aufgabenstellung dieses Projektes beinhaltete aber den Bau eines eigenen Prototypen welcher die Resultate aufbereitet und darstellt.

Zuerst begann ich mich mit dem neuen Begriffen Big Data, dem Netezza System und IBM SPSS auseinandersetzten, um überhaupt abschätzen zu können was der zu entwickelnde Prototyp alles können muss.

Danach entschied ich mich die Logfile-Daten mittels mehreren Scripts zu bearbeiten und in eine Datenbank zu speichern um diese besser handhaben zu können. Ferner sollte der Datensatz mit weiteren Daten ergänzt werden. Es sollten geographische Daten bezogen auf die IP Adresse und Einwohnerzahlen erhoben werden. Die IP Adressen werden mit dem Data Science Toolkit in geographische Daten aufgelöst. Auf das Tool und die verwendete API wird auf der Seite 24 näher eingegangen.

Aus zeitlichen Gründen entschied ich mich dazu die Darstellung der Daten nicht mit einem eigenen Prototypen zu realisieren, sondern mit den Analyseprogrammen IBM SPSS und IBM Cognos Report Studio, sowie dem Webtool Wordle. Für die Darstellung der geographischen Daten soll das CartoDB Tool verwendet werden. Mehr Informationen zu diesem Tool kann auf der Seite 26 gefunden werden.

Data Science Toolkit

Das Data Science Toolkit ist wie das CartoDB ein Open-Source Toolkit. Es bietet eine Reihe von verschieden Tools an, die für die Auswertung von Daten verwendet werden können. Für jedes Einzeltool stellt der Service eine eigene API zur Verfügung.
Die Funktion "IP Address to Coordinates" wurde in diesem Projekt für die Auflösung der IP Adressen in Koordinaten verwendet, darum wurde sie unten ausführlicher beschrieben. Die Beschreibung der anderen Tools, wie zum Beispiel "HTML to Text" oder "Text to Sediment" können unter der Adresse (Warden, 2013) nachgelesen werden.

IP Adresse zu Koordinaten

Diese Funktion ist über die API „/ip2coordinates“ erreichbar. Dabei kann eine IP Adresse mittels einem GET-Request an diese übergeben werden. Wenn die IP Adresse in Koordinaten aufgelöst werden kann, sendet die API ein JSON Objekt mit Längen- und Breitengrad, einen Ländernamen, Ländercode, Ländercode-3, Stadtnamen, eine Postleitzahl und drei weitere Attribute, die aber meistens leer bleiben, zurück. Kann die IP Adresse nicht aufgelöst werden, dann wird ein JSON Objekt, dass den Wert null hat zurückgegeben.
Das empfangene JSON Objekt kann nun von einer Applikation empfangen werden und in die einzelnen Attribute geparst werden.

Beispiel

Es wird folgender GET-Request an die API gesendet:

www.datasciencetoolkit.org/ip2coordinates/123.79.154.11

Dieser Request wird nun in Koordinaten aufgelöst und in einem JSON Objekt zurück an den Sender gesendet. Das JSON Objekt sieht wie folgt aus:

```
{
  "123.79.154.11": {
    "dma_code": 0,
    "latitude": 39.9289016723633,
    "country_code3": "CHN",
    "area_code": 0,
    "longitude": 116.388298034668,
    "country_name": "China",
    "postal_code": "",
    "region": "",
    "locality": "Beijing",
    "country_code": "CN"
}
}
```

Kann der genau gleiche GET-Request nicht in Koordinaten aufgelöst werden würde die API folgendes JSON Objekt zurück senden:

```
{
  "123.79.154.11": null
}
```

Installation eines eigenen Data Science Toolkit

Da am Anfang des Projektes eine grosse Menge an Daten zu vermuten war, wurde wie beim CartoDB entschieden einen eignen Server dieses Services aufzusetzen. Der Grund für diese Entscheidung war, dass das Senden und Empfangen aller GET-Requests über das Internet eine zu lange Zeit in Anspruch genommen hätte. Mit einem internen Server kann durch kürzere Wege eine schnellere Abfrage gewährleistet werden.

Für die Installation des Services gibt es verschieden Möglichkeiten. Es kann ein .ova Image heruntergeladen werden das mittels einer VM VirtualBox von Oracle oder dem VMware Player direkt als virtuellen Server ausgeführt werden kann. In diesem Projekt war dies aber nicht möglich, weil schon ein virtueller Server für die Installation verwendet wurde.

Für die Installation des Services steht ein Readme zur Verfügung, das unter (Warden, 2013) abgerufen werden kann.

IBM SPSS

SPSS ist eine Programmfamilie von IBM, dass zur statistischen Analyse von Daten verwendet wird. Das Basismodul ermöglicht ein grundlegendes Datenmanagement und umfangreiche statistische sowie grafische Datenanalysen der gängigsten statistischen Verfahren.

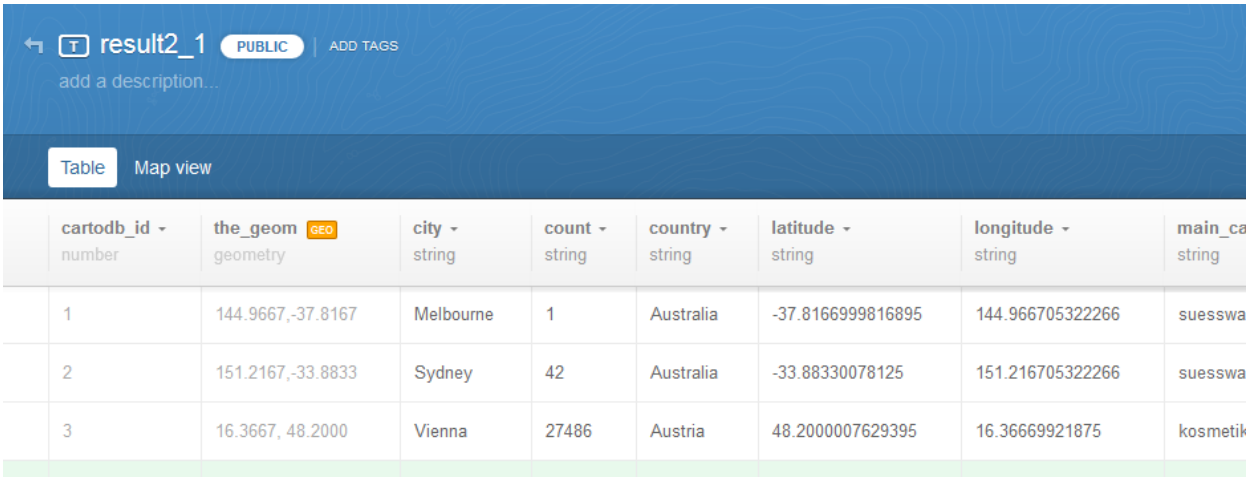
Das SPSS kommt oft in wissenschaftlichen Bereichen zum Einsatz, es findet aber auch in Unternehmen für Marktforschung Anwendungen. (vgl. Wikipedia, 2013).

CartoDB

CartoDB Tool ist eine online Datenbank in einer Cloud, die auf PostgreSQL und PostGIS basiert. Das Tool wird über ein Webinterface gesteuert und bietet in erster Linie die Speicherung der Daten in einer Datenbanktabelle und Funktionen zur Abbildung geographischer Daten auf einer Karte. Es können auch Daten analysiert werden.
Um das Webinterface nutzen zu können muss man einen Benutzerkonto erstellen. Dieses kann gratis verwendet werden um Daten von einer Grösse von 5 Megabytes zu visualisieren. Dabei ist die freie Anzahl der Tabellen auf 5 und die Anzahl der Visualisierungen auf 10'000 beschränkt. Bei mehr als 5 Tabellen und mehr als 5 Megabytes Daten ist das online CartoDB kostenpflichtig.

Verwendung von CartoDB
Nach der Anmeldung des Benutzerkontos kann CartoDB direkt verwendet werden. CartoDB ist in fünf Register "tables", "visualizations", "common data", "documentation", "<benutzername>" unterteilt.

Im Register **"tables"** können Daten über den "New Table" Button in das Tool importiert werden. Es werden Daten in den Formaten .CSV, .GEOJSON oder .SQL akzeptiert, welche direkt eine Datenbanktabelle erzeugen. Zusätzlich gibt es die Möglichkeit Daten von einer Dropbox oder von Google-Drive zu importieren.
Die erzeugten Datenbanktabellen werden nach dem importieren direkt in einem Tabellenmenü (siehe Abbildung 18)angezeigt. Um die Tabellen auf einer Landkarte visualisieren zu können, müssen diese geographischen Daten Längen- und Breitengrade beinhalten. Diese zwei Attribute werden automatisch erkannt sofern sie Longitude oder Latitude genannt werden. Sie können aber auch manuell als GEO-Daten definiert werden. Diese werden dann zu einem Attribut des Datentyp "Geometry" zusammengefasst und können auf einer Karte visualisiert werden.



cartodb_id number	the_geom geometry	city string	count string	country string	latitude string	longitude string	main_ca string
1	144.9667,-37.8167	Melbourne	1	Australia	-37.8166999816895	144.966705322266	suesswa
2	151.2167,-33.8833	Sydney	42	Australia	-33.88330078125	151.216705322266	suesswa
3	16.3667, 48.2000	Vienna	27486	Austria	48.2000007629395	16.36669921875	kosmetik

Abbildung 18 CartoDB Tabellenmenü

Sind die GEO-Daten definiert, so kann die Tabelle mittels klicken auf den "Map view" Button (Siehe Abbildung 18) auf der Karte dargestellt werden. Die Geo-Punkte werden dann wie in der Abbildung 19 ersichtlich auf einem Karten Layer dargestellt.

Rechts in der Kartenansicht können nun weitere Layer hinzugefügt und bearbeitet werden. Dabei kann man die Farbe und Grösse der GEO-Punkte im "Visualization Wizard" nach eigenem Ermessen anpassen. Zusätzlich besteht die Möglichkeit die Punkte mittels SQL-Statements auf verschiedene Layer zu verteilen. Die Layer können in der späteren Kartenansicht ein- und ausgeblendet werden.

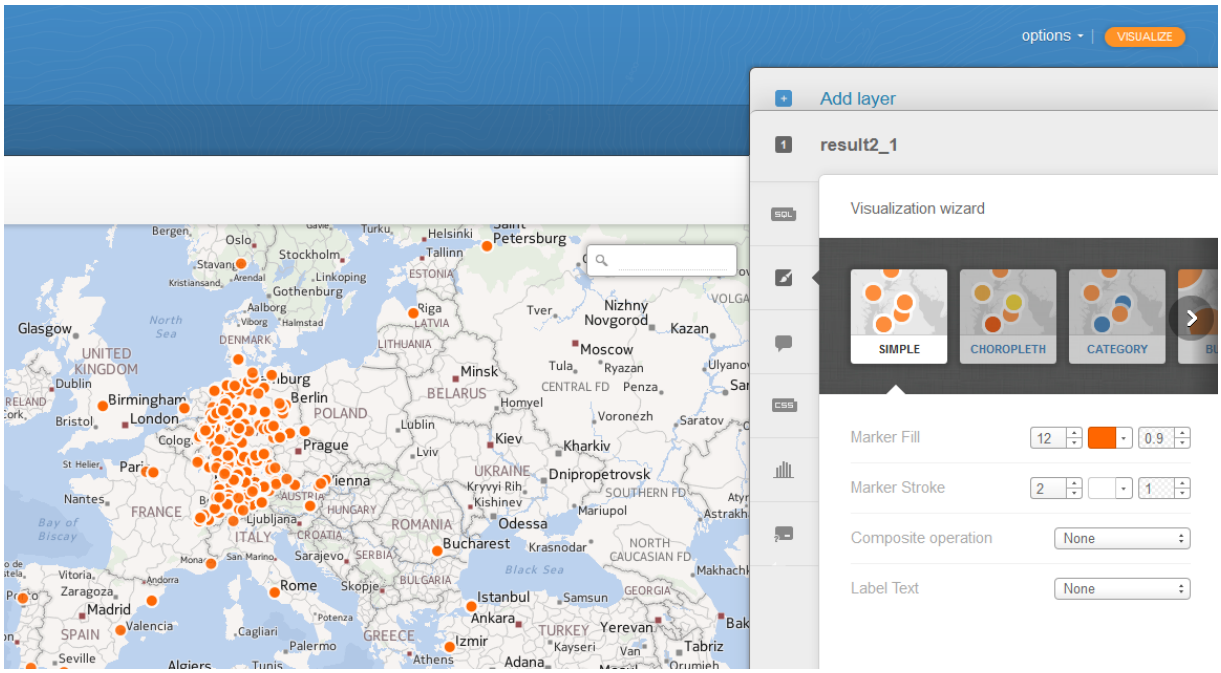


Abbildung 19 CartoDB Karte.

Sind nun alle Layer definiert so kann die Visualisierung der Karte gespeichert werden. Das geschieht über den orangen "Visualize" Button.
Ist die visualisierte Karte gespeichert, so kann diese mittels Klicken auf den "Publish" Button veröffentlicht werden. Die Karte ist dann über eine URL aufrufbar.

Im Register "**visualizations**" können die gespeicherten Visualisierungen angeschaut und bearbeitet werden.

Das Register "**common data**" bietet verschieden Beispieldatensätze, die frei verwendet werden können um die verschiedenen Visualisierungsmöglichkeiten zu demonstrieren.

Im Register "**documentation**" werden verschiedene Tutorials angeboten, die die Visualisierungsmöglichkeiten Schritt für Schritt erklären. In diesem Register ist auch die Beschreibung der API zu finden.

Im"<**benutzername**>" Register können die Benutzereinbestellungen getätigt werden. Zusätzlich wird in diesem Register der benutzerspezifische Schlüssel für der API angezeigt.

Weiter Informationen über das CartoDB kann unter (CartoDB, 2013) abgerufen werden.

Installation eines eigenen CartoDB Service

Das CartoDB ist ein Open-Source Tool und kann frei heruntergeladen werden. Um die Kosten für die grösseren Speicherkapazitäten beim Webinterface zu sparen, wurde entschieden das Tool auf einem ZHAW-Server einzurichten. Das ganze Tool liegt in Form eines GitHub-Repository vor und kann auf einem Linux Server installiert werden. Für die Installation gibt es eine Installationsanleitung in Form eines Readme, welches (CartoDB, 2013)abrufbar ist.

Vorgehen

Der implementierte Software-Prototyp wurde nicht als einzelne Applikation realisiert sondern besteht aus mehreren Scripts, die die Daten in mehreren Schritten verarbeiten sollen. Das erste Script soll die Daten der Logfiles parsen, für meine Zwecke brauchbare von unbrauchbaren Informationen trennen und sie für eine Datenbank aufbereiten. Die weiteren Scripts sollen die Daten aus der Datenbank lesen, verarbeiten und sie wieder in die Datenbank schreiben.
In den nachfolgenden Abschnitten werden die Grundgedanken der Datenbank, wie auch von den einzelnen Scripts erklärt. Zum Schluss wird das Laden der Daten in das Netezza System beschrieben.

Interpretation der Daten

Beim analysieren der Logfiles wurde schnell klar, das es in diesen viele verwandte Zeileneinträge gibt. Das kommt davon, dass der Client beim Aufruf einer Webseite immer mehrere GET-Request an den Server sendet, um den HTML Code, die Bilder und die verschiedenen Scrips zu empfangen. Der Host Server protokolliert diese alle im Log.
Da diese zusätzlichen Anfragen keine weiteren Informationen hergeben, mussten sie für die Auswertung der Daten nach dem Parsen aus den Logfiles herausgefiltert werden.

Datenbank

Die Datenbank besteht aus einer Tabelle und verfügt über einen Primärschlüssel und mehrere Attribute. Diese ergeben sich primär aus den geparsten Logfiles. Zusätzlich wird die Datenbank mit weiteren Attributen für die Erfassung von geographischen Daten, dem Plain-Text der jeweiligen Webseite, wie auch Attribute für die Verarbeitung des Requests und des Browsers angereichert. Diese zusätzlichen Attribute werden für die spätere Auswertung der Daten verwendet.

Logfile Parser Script

Der Logfile Parser besitzt keine Datenbankanbindung stellt aber die Grundfunktionen zum Parsen der Logfiles, sowie die Evaluation und Aussortierung der unbrauchbaren Zeileneinträge zur Verfügung. Nach dem Parsen der Logfiles erzeugt das Script eine Textdatei, in der alle Logeinträge zu SQL-Insert Statements verarbeitet werden.

Diese SQL-Insert Statements werden in einem nächsten Schritt mittels der SQL Command Shell manuell in die vorbereitete Datenbank eingefügt.

Geo-Daten Script

Das zweite Script verfügt über eine Datenbankanbindung und realisiert die Auflösung der IP-Adressen zu geographischen Daten. Das Script stellt eine Verbindung zur Datenbank her und liest IP-Adressen aus. Diese werden danach, wenn möglich, mit der Data Science Toolkit API IP2Coordinates in geographische Daten aufgelöst und mittels einem Update-Statement in die Datenbank geschrieben.

Wenn die IP-Adresse nicht aufgelöst werden kann wird der Standartwert der Datenbank Attribute beibehalten.
Diese Daten können bei der Auswertung für die Darstellung der Resultate auf einer Karte verwendet werden.

Einwohnerzahl Script

Das dritte Script erweitert die Datenbank mit den Bevölkerungszahlen der einzelnen Städte oder Gemeinden.
Dabei liesst das Script die Bevölkerungszahlen aus einer Textdatei heraus und vergleicht diese mit dem entsprechendem Datenbankwert. Da es keine einheitlichen Datensätze für diese Bevölkerungszahlen gibt, wäre es ein zu grosser Aufwand gewesen alle Städte und Gemeinden, die in der Datenbank vorkommen, eine Einwohnerzahl zuzuweisen. Um den Aufwand in Grenzen zu halten, beschränkte man sich dabei auf die Länder Deutschland, Schweiz und Österreich. Die Bevölkerungszahlen können zusammen mit den Geo-Daten für eine realistischere Darstellung auf einer Karte dienen.

HTML Plain-Text Script

Das vierte Script verfügt auch über eine Datenbankverbindung und liesst das „Request“ Attribut der Logeinträge aus. Mit dieser Information kann nun eine URL gebildet werden, mit dessen Hilfe der gesamte HTML-Code eingelesen werden kann. Dieser wird geparst und in ein vorbereitetes Attribut in der Datenbank geschrieben. Dieser gesamte Text könnte bei der Analyse der Daten für eine Textanalyse verwendet werden.

Browser Script

Im fünften Script wird wie bei den anderen Scripts das Attribut „Browser“ eines Logeintrages ausgelesen. Das gelesene Attribut wird im Script geparst und nach Merkmalen, wie "mobile", "IPhone", "GoogleBot" und "Windows NT" geprüft. Mit dieser Überprüfung kann bestimmt werden mit welcher Art von Gerät eine Anfrage an den Server gestellt wurde.
Diese Attribute geben bei der Auswertung der Daten die Möglichkeit, nach einem Gerätetyp zu selektieren.

Request Script

Beim sechsten Script wird das Attribut „Request“ aus der Datenbank gelesen und in seine Einzelteile geparst. Das Script überprüft zusätzlich um welche Art von Anfrage es sich handelt. Stellt das Script ein Produkt- oder Kategorieanfrage fest, so wird das eingelesene Attribut geparst und in Kategorie Level aufgeteilt. Diese weitern, neu gewonnen Attribute werden in der Datenbank als neue Attribute gespeichert.
Stellt das Script eine Anfrage für ein alternatives Produkt fest, wird aus dem Request Attribut eine URL zusammengestellt und der gesamten HTML-Code der angeforderten Seite eingelesen. Aus

diesem HTML-Code wird dann der Titel heraus extrahiert und zusammen mit dem Vermerk "Alternative Products" in die Datenbank geschrieben.
Stellt das Script eine allgemeine Suchanfrage fest, so wird der Request auch geparst und der gesuchte Begriff zusammen mit dem Vermerk „Product seach“ in die Datenbank geschrieben.
Das Script fängt auch die Anfragen für ein die Einsicht in Kommentare, einer Produkt Version, so wie auch das nachschauen eines Herstellers ab. Diese Attribute werden nicht verarbeitet sondern es wird lediglich ein Vermerk „Comment Popup“, „Product Versions“, „Manufacturer Lookup“ oder „Product Edit“ in die Datenbank geschrieben. Die Anfragen, die keiner dieser Anforderungen entsprechen, werden als unbrauchbare Daten betrachtet. Es werden die Standartwerte („-„) der Datenbank Attribute beibehalten.
Die Erzeugung dieser zusätzlichen Attribute bieten bei der Analyse der Daten die Möglichkeit, nach spezifischen Kategorien, Produkten oder Suchbegriffen zu suchen oder zu selektieren.

ETL-Prozess

Nach dem Ausführen der einzelnen Scripts ist die Datenbank mit genügend Informationen angereichert um sie zu exportieren.
Das Laden der Daten in das Netezza System wurde vor Ort bei der Firma Serwise vorgenommen. Dabei wurde ich von Herr Altmann tatkräftig unterstützt.

Für das effektive Laden der Daten in das Netezza System empfahl er die Verwendung des ETL Programmes Pentaho Kettle**.** In diesem Programm wird die exportierte MySQL Datenbank in Form einer „.CSV“ Datei eingelesen und verarbeitet.

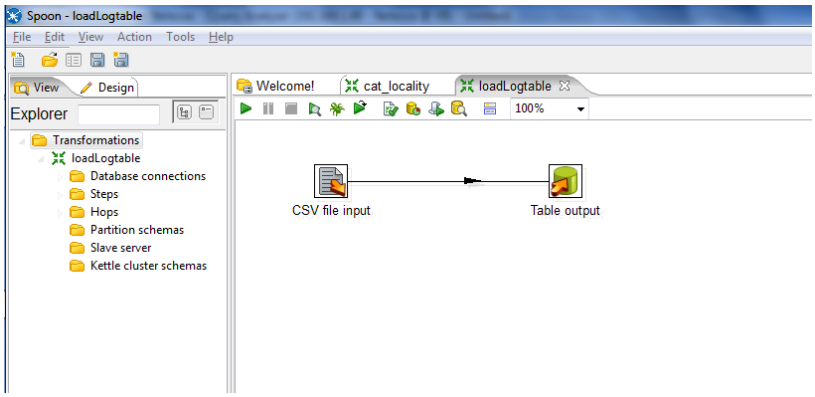


Abbildung 20 ETL-Lade-Prozess

Wie man in der Fehler! Verweisquelle konnte nicht gefunden werden. erkennen kann wird diese erarbeitung mittels Einfügen und Verbinden von einzelnen Bausteinen konfiguriert, respektive modelliert. Sind alle Bausteine korrekt konfiguriert, kann der Ladeprozess gestartet werden.

Der der erste Ladeprozess funktionierte auf anhieb, leider entstand ein Encoding Problem, dass nicht direkt Bemerkt wurde. Nach kurzer Zeit wurde die lokale MySQL Datenbank nochmal um das Attribut „Einwohnerzahl“ erweitert. Dass hatte zur Folge, dass die ganze Datenbank nochmals exportiert werden musste und nochmals neu in das Netezza System geladen werden musste.

Kurz nach dem Start des zweiten Ladeprozesses trat wieder ein Encoding Problem auf. Danach war nicht mehr möglich die Datenbank in das Netezza System zu laden.
Nach stundenlanger Fehleranalyse entschied ich mich dazu alle Sonderzeichen, wie „ä“, „ü“ und „ö“ zu ersetzen. Dabei wurde ein Parser Script so angepasst, dass es die exportierte CSV Datei, Zeile für Zeile durchgeht und dabei alle Sonderzeichen durch ASCII-Zeichen ersetzt.
Nach dem ersetzen aller Sonderzeichen konnte die angepasste CSV Datei mit dem Latin-1 Zeichensatz in das Netezza-System geladen werden.

In der folgenden Abbildung 21 soll die Architektur des Prototypen visualisieren

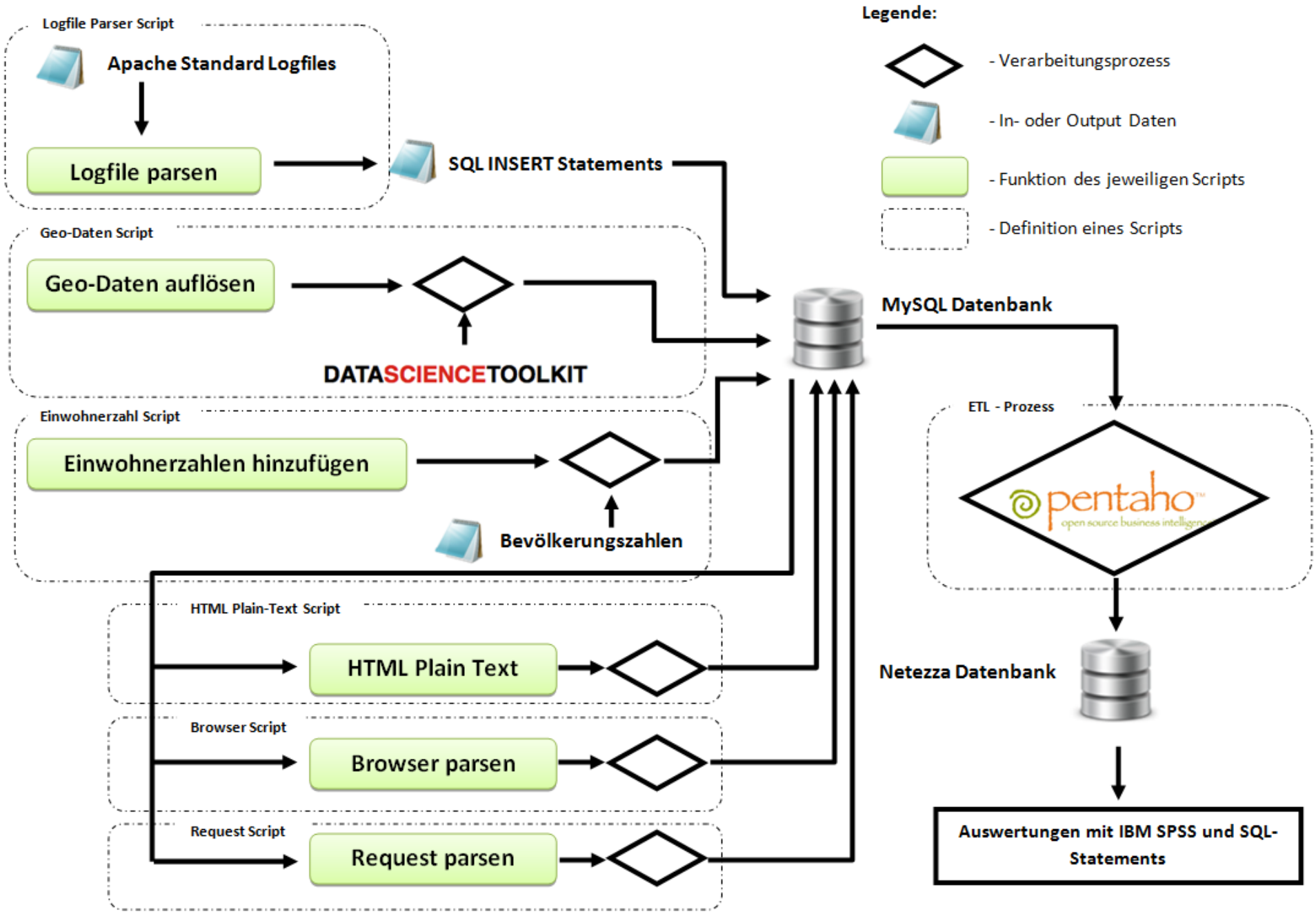


Abbildung 21 Architektur des Prototypen.

Auswahl für die Weiterentwicklungen

Verbesserung der Scripts

Die einzelnen Scripts sind so implementiert, dass sie speziell für diese Logfiles verwendet werden können.
Beim Erstellen der Scripts wurde darauf geachtet, dass beim parsen der jeweiligen Attribute möglichst viele Spezialfälle abgefangen werden, um allfällige Fehlerquellen zu minimieren. Das Ziel war es einen Block von Daten ohne Unterbrechungen zu verarbeitet. Denn diese können bei der Verarbeitung von grossen Mengen von Daten zu langen und mühsamen Verzögerungen führen.

Die Scripts wurden im Laufe des Projektes stets optimiert und arbeiten nunmehr fast fehlerfrei. Es treten aber immer wieder neue Spezialfälle auf, die einen Fehler verursachen können.
Da es sich aber schon bald abzeichnete, dass nicht mehr als sieben Logfiles an Daten für das Projekt zur Verfügung stehen werden, wurden die Parser nur auf diese sieben Dateien optimiert.

Wenn aber ein späteres Projekt diese Parser wieder verwenden möchte, müssten diese für eine grössere Menge an Daten noch durch weiteres Abfangen von Spezialfällen erweitert werden.
Um auch weitere Encoding Probleme zu umgehen wäre es sicher von Vorteil, wenn alle Sonderzeichen schon beim Parsen der Logfiles durch ASCII Zeichen ersetzt würden.

Tests

Wie in der Beschreibung der Projektarbeit geschrieben steht, wurde hier nur ein Software-Prototyp, der aus mehreren Scripts besteht, entwickelt. Dieser erfüllt im Normalfall die funktionalen Anforderungen einer Aufgabe.
Diese wurden durch das ausführen der Scripts getestet. Um aber eine 100% Sicherheit über die Funktionalität des Prototypen zu erhalten, müssten jedoch noch explizite Tests geschrieben werden.
Aus zeitlichen Gründen konnten keine expliziten Tests mehr realisiert werden.

5. Implementation

In diesem Kapitel wird die technische Implementation der einzelnen Scripts des Software Prototypen, wie auch der Aufbau der Datenbank beschrieben. Der Prototyp besteht wie im Grobkonzept beschrieben aus mehreren Scripts, welche ihre Aufgaben alle sequentiell abarbeiten.
Die Prototypenteile sind in der Programmiersprache Java realisiert und verwendet die Java Umgebung 1.7.
Für die Zwischenspeicherung der Daten wird eine MySQL Datenbank verwendet, welche über die MySQL-Workbrench gemanagt wird.

Initialisierung der MySQL-Datenbank

In der Datenbank wird eine Tabelle erzeugt. Sie verfügt über verschiedene Attribute und einen Primärschlüssel. Die Attribute IP-Address, Date, Referer und Browser werden für die Speicherung der geparsten Logeinträge verwendet, wie auch die Attribute Latitude, Longitude, Country_code3, Country_name, Country_code, Locality und Population die für die Erfassung der geographischen Daten verwendet werden. Weiter besitzt die Datenbank noch ein Attribut Sidecontent, in dem der gesamte Plaintext der jeweiligen HTML-Seite der angefragten Seite gespeichert wird. Um die bearbeiteten Werte des Request und Browser Attributes zu speichern werden zusätzlich noch die Attribute Product_ID, Request_cat_level 1 bis 3, EAN_number, Request_product, Request_device und Requet_device_cat zur Datenbank hinzugefügt. Die Varchar Attribute werden mit dem Standartwert „-„ initialisiert und die Double Attribute mit 0.0.

Attribut Datenbank	Datentyp	Erkärung
PimaryKey	Integer (11)	Automatisch generierter Primärschlüssel
IP-Address	Varchar (30)	IP-Adresse
Date	Datetime	Zeitstempel
Referer	Varchar (500)	Referer
Browser	Varchar(500)	Browser / Betriebssystem / Oberfläche
Latitude	Double	Breitengrad
Longitude	Double	Längengrad
Country_code3	Varchar (10)	Länderabkürzung z.B. DEU (3 Ziffern)
Country_name	Varchar (40)	Ländername
Country_code	Varchar (10)	Länderabkürzung z.B. DE (2 Ziffern)
Locality	Varchar (40)	Stadt
Sidecontent	Longtext	HTML Plain-Text
Product_ID	Varchar (15)	Produkt ID
EAN_number	Varchar (20)	EAN Nummer
Request_cat_level_1	Varchar (50)	Toplevel der Kategorie, oder die Beschreibung der Anfrage. Zum Beispiel „Product search“ oder „Alternative search“
Request_cat_level_2	Varchar (200)	Erste Spezialisierung der Kategorie oder gesuchter Begriff
Request_cat_level_3	Varchar (50)	Zweite Spezialisierung der Kategorie oder Seitenanzahl der Suche.
Request-product	Varchar (70)	Angabe des spezifischen Produktes, wenn ein solches vorliegt sonst „-„.
Request_device	Varchar (20)	Mit was für ein Gerät wurde die Anfrage getätigt (IPhone, IPad, Googlebot, Windows Phone, usw.)
Requet_device_cat	Varchar (20)	Was für eine Art von Gerät wurde verwendet (Mobil, Client und Webcrawler)
Population	Integer (11)	Einwohner Zahl der jeweiligen Stadt

Tabelle 2 Aufbau Datenbank

Datenbankanbindung

Für die Datenbankanbindung wurde ein eigenes Package definiert. In diesem Package ist die Grundlegende Funktion der Datenbankverbindung, sowie die Ausführung eines SQL-Statements implementiert. Die Datenbankverbindung wurde über JDBC realisiert und verwendet die von der MySQL Datenbank mitgelieferten Bibliothek JConnector.

Implementierung Logfile Parser Script

Der Logfile Parser hat wie im Grobkonzept beschrieben die Aufgabe Logfiles zu parsen und unbrauchbare Informationen zu verwerfen.
Das parsen der Logfiles wird mittels Regular Expressions (Regex) realisiert. Regex hat in diesem Fall den Vorteil, dass nur ein Muster definiert werden muss, nach dem alle Logzeilen geparst werden können. Bei allfälligen Fehlern muss nur das Muster angepasst werden und nicht die ganze Codestruktur.

Da das Apache-Standart-Logfile ein Standardformat ist und die Erstellung dieser Muster nicht ganz trivial sind gibt es schon fertige Regex Muster, die für das parsen dieser Logfiles verwendet werden können. Für diesen Logfile Parser wurde folgendes Muster verwendet.

```
String logPattern= „^([\\d.]+) (\\S+) (\\S+) \\[([\\w:/]+\\s[+\\-]\\d{4})\\]”+
                   “ \"(.+?)\" (\\d{3}) (\\d+|\\S+) \"([^\"]+)\" \"([^\"]+)\"“;
```

Für die Erstellung, Bearbeitung und das Testen dieser Muster gibt es Webtools, die die Funktion dieser Muster veranschaulichen. Eines dieser Tools ist unter (Skinner, 2008) frei verwendbar.

Um nun die verwandten Zeileneinträge zu erkennen und diese auszusortieren werden die geparsten Einträge auf Signalwörter oder Signalabkürzungen geprüft. Kommt in einer Anfrage zum Beispiel die Abkürzung „.jpg“, „.gif“ oder „.png“ vor, so kann man diese Einträge zweifellos als Bildanfragen identifizieren und diese als verwandte Zeile ignorieren. Nach diesem Verfahren werden auch CSS, JavaScript und weitere verwandte Anfragen herausgefiltert und ignoriert.

Die verwendeten Zeileneinträge werden dann zu SQL-Insert Statements zusammengesetzt und in eine Ausgabedatei geschrieben.

Implementierung Geo-Daten Script

Im zweiten Script wird die IP Adresse zuerst mittels einem HTTP-Request an die Data Science Toolkit API IP2Coordinates gesendet. Die geographischen Daten werden dann in Form eines JSON Objekt zurück gesendet. Diese Daten werden mit einer JSON Parser Bibliothek geparst und in die Datenbank geschrieben.
Da die gleichen IP Adressen mehrmals vorkommen, werden die geographischen Daten jeder IP Adresse in einer generischen Hashmap gespeichert.
Es wurde eine generische Hashmap als Speicher gewählt, weil sie auch bei vielen Dateneinträgen eine konstante Zugriffszeit gewährleistet (Optimaler weise O(1)). Die IP Adressen können als Schlüssel verwendet werden und die geographischen Daten als Datenwert der Hashmap.
Um die langsamen HTTP-Requests zu minimieren schaut das Script bei jeder IP Adresse zuerst in der Hashmap nach, ob diese IP Adresse schon einmal angefragt wurde. Ist diese in der Hashmap gespeichert, so werden die Daten aus der Map bezogen. Ist das nicht der Fall so wird ein HTTP-Request an die API gesendet.

Im Laufe des Projektes wurde aber klar, dass auch mit dem Speichern der Daten die Auflösung der IP Adressen zu langsam ist. Es wurde entschieden einen internen Server der ZHAW mit dem Data Science Toolkit aufzurüsten um die HTTP-Requestzeit weiter zu minimieren.

Implementierung Einwohnerzahl Script

Die Einwohnerzahlen werden aus Textdatei gelesen und in eine Hashmap gespeichert. Diese Hashmap dient in diesem Fall als Lookup-Table. Danach werden aus der Datenbank alle Orte ausgelesen und mit der Hashmap verglichen. Wird die Ortschaft in der Hashmap gefunden, so wird

die entsprechende Einwohnerzahl in die Datenbank geschrieben. Wird der Ort nicht gefunden bleibt der Standartwert 0 in der Datenbank stehen. Die Bevölkerungszahlen von Deutschland und Österreich wurden von (Brinkhoff, 2013) und die Bevölkerungszahlen der Schweiz von (Bundesamt für Statistik, 2013) bezogen.

Implementierung HTML Plain-Text Script

Das HTML Plain-Text Script liest nach dem zusammensetzen der URL den gesamten HTML-Code ein. Dieser wird mit der Java Bibliothek JSOUP geparst und mittels einem Update-Statement in die Datenbank geschrieben.
Da das Einlesen der HTML Codes, wie beim Auflösen der Geo-Daten, einen HTTP-Request auslöst, wurde auch hier eine generische Hashmap verwendet um das Script zu beschleunigen. Die URL's wurden dabei als Schlüssel und der verarbeitete HTML Plain-Text als Datenwert gespeichert.

Implementierung Browser Script

Die Verarbeitung des Browser Attributes wird mittels String Split-Funktionen realisiert. Nach dem überprüfen der Merkmale, werden die verarbeiteten Daten mittels Update-Statement in zwei vorbereitete Attribute in die Datenbank geschrieben.

Implementierung Request Script

Das eingelesene Attribut wird mit String Split-Funktionen geparst und auf die im Grobkonzept erläuterten Request Typen geprüft. Das Script identifiziert die Requests anhand von Signalwörtern. Zum Beispiel kann man bei einem spezifischen Produkt am Schluss der Zeichenkette immer ein „.pro“ und am Anfang einer allgemeinen Suchanfrage immer eine „/product.search?“ finden.

Da die Suchbegriffe von alternativen Suchen nicht im Request Feld ersichtlich sind, muss der ganze HTML Code eingelesen werden. Um die Geschwindigkeit des Einlesens zu verbessern, wurde auch hier eine Hashmap zur Speicherung der angefragten Seiten verwendet.

Aufbau der SQL-Statements

Die folgenden SQL-Statements wurden für die Auswertung der Produktkategorien verwendet. Sie werden direkt auf dem Netezza System ausgeführt und generieren eine Resultattabelle, die danach mittels einer CSV Datei exportiert werden kann.

Das erste SQL-Statement zählt alle Kategorie Level 1 Einträge der zu analysierenden Produktkategorien bezogen auf einen Ort (Locality). Dabei werden die Kategorien mit dem Vermerk „Rauchwaren“ und „Getränke Alkohol“ zu „Rauchwaren Alkohol“ und die Kategorien „Getränke Genussmittel“ und „Süsswaren zu „Getränke Süsswaren“ zusammengefasst. Die gezählten Werte werden dann in eine neue Tabelle namens „Population_agg“ gespeichert. Die neue Tabelle besitzt die Attribute „Latitude“, „Longitude“, „Country_name“, „Request_cat_level_aggregated“, “Population” und „Anzahl“.

```
Select
      LATITUDE,
      LONGITUDE,
      COUNTRY_NAME,
      LOCALITY,
      request_cat_level_aggregated,
      POPULATION,
      count(*) as Anzahl

into POPULATION_AGG
from (
      SELECT
            LATITUDE,
            LONGITUDE,
            COUNTRY_NAME,
            LOCALITY,
      case when (request_cat_level_1 = 'raucherwaren' or
                  request_cat_level_1 =  'getraenke alkohol')
            then 'rauchwaren alkohol'

            when (request_cat_level_1 = 'getraenke genussmittel' or
                  request_cat_level_1 = 'suesswaren')
            then 'getraenke suesswaren'

            else request_cat_level_1
            end as request_cat_level_aggregated,
            POPULATION
      FROM CODECHECK.UCODECHECK.LOGTABLE_V2
      where request_product != '-' and request_cat_level_1
            in ('kosmetik gesundheit', 'suesswaren', 'raucherwaren',
            'getraenke genussmittel', 'fertiggerichte fastfood',
            'getraenke alkohol')
) as a
group by    LATITUDE,
            LONGITUDE,
            COUNTRY_NAME,
            LOCALITY,
            request_cat_level_aggregated,
            POPULATION
```

Das zweite SQL-Statement verwendet nun die erzeugte Tabelle „Population_agg“ um die gezählten Werte durch die Einwohnerzahl zu dividieren. Vor der Division müssen die Werte noch zu Float gecasted werden. Nach der Division werden sie mit 1‘000‘000 skaliert, sodass alle Werte als Ganzzahl im CartoDB Tool visualisiert werden können.

```
SELECT
      LATITUDE,
      LONGITUDE,
      COUNTRY_NAME,
      LOCALITY,
      REQUEST_CAT_LEVEL_AGGREGATED,
      POPULATION,
      ANZAHL,
      (cast(ANZAHL as float) / POPULATION * 1000000)
            as population_rel

  FROM CODECHECK.UCODECHECK.POPULATION_AGG
```

6. Evaluation der Resultate

Diese Kapitel hinterfragt die erhalten Ergebnisse und Darstellungen der einzelnen Auswertungen.

Am Anfang des Projektes versprach Codecheck die Lieferung von Logfile Daten die über einen Zeitraum von bis zu 10 Jahren gehen sollten.
Nach ein paar Wochen wurden die ersten Testdaten geliefert, die eine Zeitspanne von sieben Tagen umfassten. Mit diesen Daten wurde der entwickelte Prototyp getestet und die ersten Auswertungsversuche durchgeführt.
Im Laufe des Projektes stellte sich aber heraus, dass die Serwice AG und die Firma Codecheck sich bezüglich der Geheimhaltungsvereinbarung nicht einigen konnten. Darum wurde die Analysen der Daten auf der Basis der Testdaten durchgeführt.
Das hatte zur Folge, dass die Genauigkeit der erstellten Diagramme und Auswertungen stark eingeschränkt wurde.

Evaluation Analyse der verwendeten Geräte

Diese Diagramme wurden wie im Kapitel „Resultate“ erwähnt mit den Programmen IBM SPSS und IBM Cognos Repord Studio erstellt. Ziel dieser Auswertung war es, herauszufinden, welche Art von Geräten die meisten Anfragen an die Codecheck Datenbank sendet.
Aus der Abbildung 5 und Abbildung 6 kann herausgelesen werden, dass es zwei Gruppen von Benutzer der Codecheck Datenbank gibt. Die eine Gruppe (ca. 27.5% der Benutzer) verwendet eine Mobil-App meistens auf einem IPhone oder einem Android Gerät. Daraus kann geschlossen werden, dass diese Benutzer die Strichcodes der angefragten Produkte direkt im Laden mit ihren Smartphones scannen und in der Codecheck Datenbank nachschauen.
Die zweite Gruppe umfasst rund 72.5% der Codecheck Datenbank Nutzer. Sie sendeten ihre Anfragen mit einem stationären Client. Dabei spiegelt die Abbildung 7 die Verwendung der meist verwendeten Betriebssysteme im allgemeinen wieder. Trotzdem kann man sagen, dass die zweite Gruppe von Benutzern Codecheck von zu Hause aus verwendet.
Es kann nun angenommen werden, dass die Nutzer der zweiten Gruppe sich nicht nur darüber entscheiden, ob sie ein Produkt kaufen, sondern auch wo sie diese Produkte erstehen. Um diese Annahme jedoch beweisen, müsste man noch weiterführende Studien über das verhalten der Codecheck Nutzer durchführen.
Diese Diagramme zeigen ebenso, dass es sich lohnt die Weiterentwicklungen für Windows Geräte, IPhone und Android Geräte voranzutreiben.
Da sich die Daten auf einen Zeitraum von sieben Tage beschränkt, sind diese Auswertungen auf globale Sicht als nicht repräsentativ zu erachten.

Evaluation der Analyse der Codecheck Nutzung bezogen auf Länder

Die Auswertung der Abbildung 8 und Abbildung 9 veranschaulicht die Verteilung der Anfragen auf die verschieden Länder. Um eine Verfälschung der Daten zu verhindern wurden zuerst alle Suchmaschinen Bots herausgefiltert.
Trotz dieser Massnahme sind die Grafiken mit Vorsicht zu betrachten, denn etwa 25% der IP Adressen konnten nicht in geographischen Daten aufgelöst werden. Zusätzlich wird das Bild dadurch Verfälscht, dass grosse Internet Provider wie AOL ihre europäischen Kunden über Server in den USA authentifizieren und somit diesen amerikanische IP Adressen zuweisen (vgl. GEOTEK Systemhaus Berlin, 2012).
Die Abbildung 9 veranschaulicht gut, dass die meisten Anfragen im deutschsprachigen Raum getätigt wurden. Auffallend ist auch, dass die Verwendung von Codecheck in Deutschland und Österreich grösser ist als in der Schweiz obwohl Codecheck eine Schweizer Datenbank ist.
Aufgrund dieser Erkenntnisse wurden die Kriterien der Kartendarstellung und der Wordle-Clouds definiert.

Evaluation der Analyse der beliebtesten Produktkategorien

Die ausgewählten Produktkategorien wurden einerseits frei gewählt, auf der andere Seite wurde darauf geachtet, dass möglichst die Kategorien gewählt wurden, die am meisten angefragt wurden. Man beschränkte sich somit auf die Produktkategorien Kosmetik und Gesundheit, Getränke und Süsswaren, Fertiggerichten und Fast Food und Rauchwaren und Alkohol. Weil die meisten Anfragen aus dem deutschsprachigen Raum kamen und weil für die später Kartendarstellung die Bevölkerungszahlen verwendet wurden, wurde entschieden diese Auswertung nur auf die Länder Deutschland, Österreich und der Schweiz anzuwenden.
Die Auswertung der Abbildung 10 zeigt, dass die Verteilung der gewählten Produktkategorien in den drei Ländern ungefähr identisch ist. Es ist gut zu erkennen, dass die Produkte der Kosmetik und Gesundheit Kategorie am häufigsten angefragt werden. Im Gegensatz dazu werden Rauchwaren und Alkohol von den vier Kategorien am wenigsten angeschaut. Es ist auch hier ersichtlich, dass die Codecheck Datenbank am meisten in Deutschland genutzt wird.

Bei den Abbildungen 11 bis 14 wurde die relative Anzahl an Aufrufen einer Produktkategorie im Verhältnis zur Bevölkerung dargestellt. Die farbigen Punkte auf der Karte sind um so grösser, je mehr Anfragen an einem spezifischen Ort getätigt wurden. Der Wert der die Grösse der Punkte definiert wird wie folgt berechnet.

$$Relative\ Kreisgrösse = \left(\frac{\sum der\ jeweiligen\ Produktkategorie}{Bevölkerungszahl}\right) * 1'000'000$$

Wenn man nur den summierten Wert für die Darstellung der Punkte verwendet hätte, dann würden die Punkte dort am grössten sein, wo die meisten Leute leben. Man wollte aber eine von der Bevölkerung unabhängige Visualisierung der Anfragen machen.
Um diese Abhängigkeit zwischen der Bevölkerung und den Kategorieanfragen zu eliminieren, wurden die gezählten Werte durch die Einwohnerzahl dividiert. Da dies zum Teil sehr kleine Werte zur Folge

hatte, mussten diese Zahlen mit einem sinnvollen Wert skaliert werden, um sie auf der Karte darzustellen. Es wurde dabei eine Skalierung mit dem Faktor 1‘000‘000 gewählt, so dass auch der kleineste Wert als Ganzzahl dargestellt werden kannte. Bei der Visualisierung der Daten wurden zu grossen und zu kleine Werte gefiltert.

Die Karte in Abbildung 11 veranschaulicht die Produktkategorie Kosmetik und Gesundheit. Man kann gut erkennen, dass in allen drei Länder diese Kategorie sehr häufig angefragt wird. Dieser Trend hat sich zwar schon durch die vorhergehenden Auswertungen abgezeichnet. Hinzukommt, dass diese Grafik auch veranschaulicht, dass die Verteilung zwischen Stadt und Land in etwa gleich ist.

Die Abbildung 12 zeigt die Produktkategorie Fertiggerichte und Fast Food. Auf dieser Karte kann man gut erkennen, dass die meisten Anfragen Deutschlands und Österreichs in den grossen Städten getätigt werden. In der Schweiz ist der Stadt Land Trend nicht so stark ausgeprägt. Ein Grund dafür könnte sein, dass es in Deutschland nur in grösseren Orten Fast Food Restaurants wie McDonalds oder Burger King gibt. Dies könnte zur Folge haben, dass die Landbevölkerung sich wegen den grösseren Distanzen nur wenig mit Fast Food und Fertiggerichten befasst. In der Schweiz sind die Distanzen zu grösseren Orten für die Landbevölkerung viel kleiner. Somit kommen die Menschen vom Land auch öfters mit Fast Food Restaurants in kontakt und interessieren sich somit auch mehr dafür. Darum ist in der Schweiz die Dichte der Anfragen viel höher als in den anderen zwei Ländern.

Die Abbildung 13 visualisiert die Kategorie Getränke und Genussmittel. In dieser Kategorie befinden sich zum Beispiel Coca Cola und Haribo. Es ist auch hier zu erkennen, das es in Deutschland einen unterschiedlichen Trend zwischen Stadt und Land gibt. In der Schweiz und Österreich ist die Verteilung ungefähr gleich. Es fällt auf, dass es in der Westschweiz wie auch im Wallis viele Anfragen in dieser Kategorie gestellt werden. Annahmen warum dies so ist kann man nur schwer treffen.

Die Karte bei Abbildung 14 stellt die Anfragen der Kategorien Rauchwaren und alkoholische Getränke dar. Wie bei der IBM Cognos Grafik in er Abbildung 9 bereits ersichtlich war, wird diese Produktkategorie am wenigsten angefragt. Es ist aus diesem Grund auch nicht verwunderlich, dass verhältnismässig nur wenige Punkte auf der Karte dargestellt werden. Es ist in allen drei Ländern wieder charakteristisch, dass die meisten Anfragen aus Städten kommen. In der Schweiz kann man zusätzlich noch erkennen, dass es in der Ostschweiz nur wenige Anfragen dieser Kategorie gibt.

Evaluation der Wordle-Clouds

Die Wordle-Clouds sind einfach aufgebaut. Das Wordle Tool nimmt eine Liste von Wörtern entgegen und stellt diese zufällig dar. Dabei werden Wörter die mehrmals als andere Wörter vorkommen grösser und Fetter dargestellt.
In den Abbildungen 15 bis 17 abgebildeten Wordle-Clouds wurden nur die ersten 100‘000 Produkte dargestellt die in der Datenbank angefragt wurden. Auch hier beschränkte man sich auf die zuvor erwähnten drei Länder. Damit die Clouds nicht zu überladen wirken, beschränkte man sich auf die ersten 100'000 Einträge.

Die Abbildung 15 zeigt die in Deutschland angefragten Produkte. Dabei fällt schnell auf, dass die meisten Produkte Pflege Produkte wie Haarwaschmittel oder Cremen sind.

Die Abbildung 16 zeigt die angefragten Produkte in Österreich. Beim betrachten dieser Cloud kann man erkennen, dass es wie in Deutschland viel Anfragen für Pflegeprodukte gibt. Im Gegensatz zu den Deutschen suchen die Österreicher aber öfters nach Rauchwahren wie Marlboro und Energy Drinks wie Red Bull.

Die Abbildung 17 zeigt die Produkt Anfragen aus der Schweiz. Auch hier gibt es einen Trend zu Pflege Produkte. Es werden aber mehr Lebensmittel Produkte angefragt.

Diese drei Wolken zeigen, dass in jedem Land ein anderer Fokus hinsichtlich der Verwendung der Codecheck Datenbank existiert. In Deutschland verwendet man Codecheck vorwiegend für das überprüfen von Inhaltstoffen bei Pflege Produkten. In Österreich sucht man mehr nach Rauchwaren und Süssgetränken und in der Schweiz überprüft eher Lebensmittelinhalte. In jeder Wolke sind aber die Pflege Produkte signifikant vertreten. Daraus kann man schliessen, dass Codecheck gute Informationen für die Inhaltsstoffe von Pflegeprodukten liefert.

7. Rückblick

Reflexion

Trotzt grosser fachmännischer Unterstützung durch Herr Altmann der Serwice AG, wie auch der Hilfe meiner Projektbetreuer Herr Braschler und Herr Stadelmann, stellte sich das Projekt als grosse Herausforderung dar.
Die Aufgabenstellung war sehr offen formuliert, sodass es viele Freiheiten gab eine eigene Lösung zu entwickeln. Durch diese Freiheiten könnte man auch Gefahr laufen, sich durch vorschnelle Entscheidungen in grössere Probleme zu versticken. Dies offeriert jedoch auch die Möglichkeit durch individuelle Lösungsansätze zu meistern.

Ich versuchte mir nächst einen Überblick über die Thematik der Datenanalyse und des Datamining zu verschaffen. Danach analysierte ich die Daten und überlegte mir, wie diese verarbeitet und erweitert werden konnten. Ich entschied mich dazu, dass mehrere Scripts, implementiert in Java, dies übernehmen sollten.
Für die verschiedenen Zwischenschritte der Verarbeitung wurden alle Daten in einer MySQL Datenbank gespeichert. Waren alle Daten verarbeitet, wurde die Datenbank exportiert und mittels dem ETL-Tool Pentaho Kettle in das Netezza System importiert. Leider verursachte der ETL-Prozess ein Encoding Problem, dass erst durch ersetzten der Sonderzeichen gelöst werden konnte.

Die Analyse der Daten war das kritischste im ganzen Projekt. Zuerst versuchte ich mit der Unterstützung von Herr Altmann eine Auswertung mittels IBM SPSS zu realisieren. Leider funktionierte das nicht wie gewünscht. Das IBM SPSS konnte die eingelesenen Attribute nicht in richtige Kategorien einordnen, so wurde es unmöglich eine komplexere Auswertung zu realisieren. Es war dennoch möglich einfache Diagramme mit dem SPSS zu generieren.

Um noch weitere Analysen zu realisieren, entschied ich mich dazu die weiteren Auswertungen mittels SQL-Abfragen zu realisieren. Dabei wurden SQL-Statements formuliert, die eine Resultattabelle generiert. Diese wurde dann mittels Microsoft Excel, dem Webtool Wordle oder dem CartoDB Tool visualisiert.

Obwohl durch die Nutzung von IBM SPSS zunächst nur hinreichende Analysen möglich waren, war es mir möglich die Analyse der Daten durch andere Programme wie CartoDB, IBM Cognos Report Studio und dem Webtool Wordle-Could aufschlussreichere Auswertungen zu generieren.

Fazit

Zu beginn des Projektes verlief alles nach Plan. Die Scripts verarbeiteten die Daten nach kurzer Zeit so wie sie sollten. Die Erweiterung der Datenbank funktionierte auch einwandfrei. Die ersten Verarbeitungsschritte wurden lokal auf meinem Laptop durchgeführt. Mit der wachsender Datenmenge reichten diese Hardware Ressourcen jedoch nicht mehr aus. So entschied ich mich,

zusammen mit meinen Projektbetreuern die weiteren Daten auf einer Windows Server VM von der ZHAW zu verarbeiten. Zusätzlich wurde entschieden, dass die Service CartoDB und Data Science Toolkit auch auf einem internen ZHAW-Server installiert werden sollten. Diese Installationen sollten aus zeitlichen Gründen durch einen Mitarbeiter der ZHAW gemacht werden. Trotz der Zusage dieses Mitarbeiters, die Services in nützlicher Frist zum Laufen zu bringen, wurden sie leider nicht innerhalb der Projektzeit fertiggestellt. Das führte zu grossen Verzögerungen. Weiter stellte sich heraus, dass die Geheimhaltungsvereinbarung von Codecheck bei der Firma Serwise AG nicht akzeptiert wurde. Das hatte zur Folge, dass keine weiteren Daten von Codecheck geliefert werden konnten.
Dieser Entscheid schränkte die Analysemöglichkeiten soweit ein, dass keine Auswertungen über einen grösseren Zeitraum mehr realisiert werden konnten.

Ich entschied mich, mit den bestehenden Daten weiterzuarbeiten und bestmögliche Auswertungen zu realisieren.
Nach dem ersten, erfolgreichen Laden der vorbereiteten MySQL-Datenbank in das Netezza System, wollte ich eine neue Datenbank mit dem zusätzlichen Attribut Einwohnerzahl in das System laden. Diese Daten waren speziell für die Auswertung der Kartendarstellung gedacht. Das dabei aufgetretene Encoding Problem kostete mich wieder viel Zeit. Es war auch nicht möglich das Encoding Problem mit einem passenden Join zu umgehen. Ein Join auf dem gemeinsamen Attribut „Locality“ hätte zu einem grossen Datenverlust geführt, weil genau in diese Spalte am meisten von dem Encoding Problem betroffen troffen war.
Um das Encoding Problem zu lösen blieb mir nichts anders übrig als alle Sonderzeichen aus der exportierten Datenbank zu ersetzen und diese mit einem anderen Zeichensatz in das Netezza System zu laden.
Da auch der CartoDB Service auf dem ZHAW Server nicht rechtzeitig fertig gestellt werden konnte, entschied ich mich dazu ein Benutzer Login bei CartoDB zu kaufen, um eine Karten Ansicht zu generieren.

Weitere Erkenntnisse sind zum einen, dass das Netezza System für Data Warehouse optimiert ist und sich nicht unbedingt für Data Mining eignet. Zum anderen wäre für eine weiterführende Analyse dieser Daten einen Spezialist in Statistik und Psychologie hinzuziehen um genauere und weiterreichende Schlüsse zu ziehen.
Nichts desto trotz kann man sagen, dass dieses Projekt einen grossen Lernfaktor im Bereich Datenanalyse, Datenauswertung und Projektmanagement hatte. Zusätzlich bot mir die Zusammenarbeit mit der Firma Serwice AG einen exklusiven Einblick in die Welt von Big Data.

Danksagung

Ich möchte mich bei allen Leuten bedanken, die mich in diesem Projekt unterstützt haben.
Ein spezieller Dank geht an Herr Philipp Altmann und Dominik Beusch von der Serwise AG Winterthur. Herr Altmann und Herr Beusch boten mir Unterstützung im Umgang mit dem Netezza System und halfen mir bei der Auswertung der Daten.

8. Anhang

Quellenverzeichnis

Braschler, M. & Stadelmann, T., 2013. *Web-Scale Datenanalyse auf einer Big Data Appliance.* Winterthur: Institut für angewandte Informationstechnologie (InIT).

Brinkhoff, T., 2013. *City Population.* [Online]
Available at: http://www.citypopulation.de/index_d.html
[Zugriff am 10 12 2013].

Bundesamt für Statistik, 2013. *Statistischer Atlas der Schweiz.* [Online]
Available at: http://www.atlas.bfs.admin.ch/core/projects/13/de-de/viewer.htm?13.8721.de
[Zugriff am 10 12 2013].

CartoDB, 2013. *Carto DB Installation.* [Online]
Available at: https://github.com/CartoDB/cartodb
[Zugriff am 30 10 2013].

CartoDB, 2013. *CartoDB.* [Online]
Available at: http://www.cartodb.com
[Zugriff am 20 10 2013].

Cieliebak, M., 2013. *Nike Data Viz.* [Online]
Available at: http://poog.cartodb.com/viz/f89be49e-26af-11e3-ae5f-1109119d348b/embed_map?title=true&description=true&search=false&shareable=true&cartodb_logo=true&layer_selector=true&legends=true&scrollwheel=true&sublayer_options=1|1&sql=SELECT%20*%20FROM%20nike_factor
[Zugriff am 29 10 2013].

Codecheck, 2013. *Codecheck.info.* [Online]
Available at: http://www.codecheck.info/About/page.pag
[Zugriff am 09 11 2013].

GEOTEK Systemhaus Berlin, 2012. *Geolocation.* [Online]
Available at: http://meineipadresse.de/html/geolocation.php
[Zugriff am 11 12 2013].

Hunkeler, M., 2013. *CartoDB.* [Online]
Available at: http://hunkeman-1.cartodb.com/viz/f7183eae-64aa-11e3-8b89-e1a1fdc45666/embed_map?title=true&description=true&search=false&shareable=true&cartodb_logo=true&layer_selector=true&legends=true&scrollwheel=true&sublayer_options=1|1|1|1&sql=SELECT%20*%20FROM%20b
[Zugriff am 12 12 2013].

IBM, 2013. *IBM - Big Data.* [Online]
Available at: http://www-03.ibm.com/software/products/de/category/SWP10?cmp=109hf&ct=109hf77w&cr=google&cm=k&csr=41429big_data_-_ch&ccy=de&ck=bigdata&cs=phrase&S_PKG=-&S_TACT=109HF77W&mkwid=sOt7rkcHP-dc_29850462722_432i044571
[Zugriff am 09 11 2013].

IBM, 2013. *IBM PureSystems: Produktfamilie – PureData System for Analytics.* [Online]
Available at: http://www.ibm.com/ibm/puresystems/de/de/pd_analytics.html
[Zugriff am 23 11 2013].

Java2s.com, 2012. *Parse an Apache log file with Regular Expressions.* [Online]
Available at: Parse an Apache log file with Regular Expressions
[Zugriff am 30 10 2013].

Neue Zürche Zeitung, 2013. *Flickenteppich für die Steckdosen.* [Online]
Available at: http://www.nzz.ch/aktuell/inland-sommerserie-schweizer-karten-interaktiv/vergleich-strom-versorger-1.18120907
[Zugriff am 12 11 2013].

Neue Zürcher Zeitung, 2013. *Sommer-Serie «Schweizer Karten».* [Online]
Available at: http://www.nzz.ch/aktuell/inland-sommerserie-schweizer-karten-interaktiv/#
[Zugriff am 12 11 2013].

Schroeck, M. et al., 2012. *IBM.com.* [Online]
Available at: http://www-935.ibm.com/services/de/gbs/thoughtleadership/GBE03519-DEDE-00.pdf
[Zugriff am 23 11 2013].

Serwice AG, 2013. *Serwice AG.* [Online]
Available at: http://www.serwise.com/
[Zugriff am 24 11 2013].

Serwise AG / Big Data, 2013. *Big Data IBM PureSystems.* [Online]
Available at: http://www.bigdata.ch/de/partner/
[Zugriff am 24 11 2013].

Skinner, G., 2008. *RegExr.* [Online]
Available at: gskinner.com/RegExr/
[Zugriff am 30 10 2013].

Warden, P., 2013. *Data Science Toolkit.* [Online]
Available at: http://www.datasciencetoolkit.org/
[Zugriff am 10 10 2013].

Warden, P., 2013. *DSTK Installation / ec2setup.txt.* [Online]
Available at:
https://github.com/petewarden/dstk/blob/ee8f44df6ff395818c742f5523f583cf82884d3d/docs/ec2setup.txt
[Zugriff am 30 10 2013].

Wikipedia, 2013. *Wikipedia / Big Data.* [Online]
Available at: http://de.wikipedia.org/wiki/Big_Data
[Zugriff am 09 11 2013].

Wikipedia, 2013. *Wikipedia / IBM SPSS.* [Online]
Available at: http://de.wikipedia.org/wiki/SPSS
[Zugriff am 28 10 2013].
Wikipedia, 2013. *Wikipedia / Logdatei.* [Online]
Available at: http://de.wikipedia.org/wiki/Logdatei
[Zugriff am 03 10 2013].

Tabellen

Abbildungs-Verzeichnis