

Abgabe 1 Computergestützte Methoden

Gruppe 28, Fatima El-Mokdad (4338026), Helin Sude Polat(4363469)

02.Dezember 2024

Inhaltsverzeichnis

1	Der Zentrale Grenzwertsatz	2
1.1	Aussage	2
1.2	Erklärung der Standardisierung	2
1.3	Anwendungen	2
2	Bearbeitung zur Aufgabe 1	3
2.1	Struktur unseres Datensatzes	3
2.2	Datensatz importieren und Datenaufbreitung	3
2.3	Ermittlung der höchsten mittleren Temperatur in Grad Celsius .	4
2.4	Entwerfen eines Datenbank-Schemas	5
2.4.1	1.Normalform	5
2.4.2	Die 2.Normalform	5
2.5	Umsetzung des Schemas in SQL	7
2.5.1	Visual Code	7
2.6	SQL Abfrage zur höchsten mittleren Temperatur	9

1 Der Zentrale Grenzwertsatz

Der zentrale Grenzwertsatz (ZGS) ist ein fundamentales Resultat der Wahrscheinlichkeitstheorie, das die Verteilung von Summen unabhängiger, identisch verteilter (i.i.d.) Zufallsvariablen (ZV) beschreibt. Er besagt, dass unter bestimmten Voraussetzungen die Summe einer großen Anzahl solcher ZV annähernd normalverteilt ist, unabhängig von der Verteilung der einzelnen ZV. Dies ist besonders nützlich, da die Normalverteilung gut untersucht und mathematisch "handhabbar" ist.

1.1 Aussage

Sei X_1, X_2, \dots, X_n eine Folge von i.i.d. ZV mit dem Erwartungswert $\mu = E(X_i)$ und der Varianz $\sigma^2 = \text{Var}(X_i)$, wobei $0 < \sigma^2 < \infty$ gelte. Dann konvergiert die standardisierte Summe Z_n dieser ZV für $n \rightarrow \infty$ in Verteilung gegen eine Standardnormalverteilung:¹

$$Z_n = \frac{\sum_{i=1}^n x_i - n\mu}{\sigma\sqrt{n}} \rightarrow (0, 1) \quad (1)$$

Das bedeutet, dass für große n die Summe der ZV näherungsweise normalverteilt ist mit Erwartungswert $n\mu$ und Varianz $n\sigma^2$:

$$\sum_{i=1}^n X_i \sim N(n\mu, n\sigma^2) \quad (2)$$

1.2 Erklärung der Standardisierung

Um die Summe der ZV in eine Standardnormalverteilung zu transformieren, subtrahiert man den Erwartungswert $n\mu$ und teilt durch die Standardabweichung $\sigma\sqrt{n}$. Dies führt zu der obigen Formel (1). Die Darstellung (2) ist für $n \rightarrow \infty$ nicht wohldefiniert.

1.3 Anwendungen

Der ZGS wird in vielen Bereichen der Statistik und der Wahrscheinlichkeitstheorie angewendet. Typische Beispiele sind:

- Der ZGS wird unter anderem verwendet, um die Verteilung der Mittelwerte oder Summen von Zufallsvariablen zu modellieren.
- Der ZGS ist somit auch besonders wichtig für die Konstruktion von Konfidenzintervallen, da er sicherstellt, dass die Intervalle auf der Normalverteilung basieren und man somit mithilfe von Stichproben verlässliche Aussagen über eine zugrunde liegende Population treffen kann.

¹Der zentrale Grenzwertsatz hat verschiedene Verallgemeinerungen. Eine davon ist der **Lindeberg-Feller-Zentrale-Grenzwertsatz** [1, Seite 328], der schwächere Bedingungen an die Unabhängigkeit und die identische Verteilung der ZV stellt.

2 Bearbeitung zur Aufgabe 1

2.1 Struktur unseres Datensatzes

Der gegebene Datensatz umfasst, dem Namen nach zu urteilen, Daten zu einer Art Fahrradverleih sowie Wetteraufzeichnungen an verschiedenen Standorten innerhalb der USA im Zeitraum vom 01.01.2023 - 31.12.2023. Für uns sind allerdings nur die Daten der uns zugeteilten Station "Bergen St Ave" relevant. Die obersten Zellen in der ersten Zeile tragen die Spaltennamen: "group", "station", "date", "day of year", "day of week", "month of year", "precipitation", "windspeed", "min temperature", "average temperature", "max temperature" und "count", alle anderen Zellen enthalten die entsprechenden Daten. Uns fällt auch auf, dass die Anzahl der ausgeliehenen Fahrräder von der Wetterlage des jeweiligen Tages abhängt.

Bei der Überprüfung der Datenintegrität haben wir festgestellt, dass ab und zu in so gut wie jeder Spalte mal Werte fehlen, außer in den Spalten "group", "station" und "date".

2.2 Datensatz importieren und Datenaufbereitung

Im ersten Schritt wurde der gegebene Datensatz in Excel importiert, um ihn für die weitere Bearbeitung aufzubereiten. Dabei haben wir eine beliebige Spalte ausgewählt, beispielsweise Spalte A, und die Funktion *Text in Spalten* genutzt, um die Daten korrekt zu formatieren. Im Assistenten wurde die Option *getrennt* ausgewählt und anschliessend auch die Trennzeichen überprüft und überarbeitet, sodass die Daten sauber in den jeweiligen Spalten aufgeteilt werden. Notwendig ist es auch, die Spaltenformate entsprechend anzupassen, um eine korrekte Übertragung sicherzustellen. So haben wir z.B. bei *windspeed* das Datenformat explizit von *Standard* auf *Text* geändert.

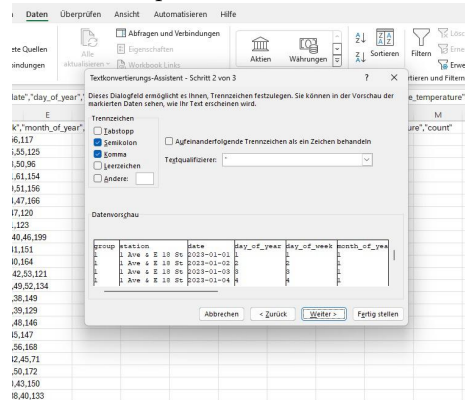


Abb. 1 Textkonvertierung

Im nächsten Schritt haben wir auch alle nicht relevanten Datensätze, die nicht zu unserer Gruppe gehören (1-27, 29-100) aus dem Datensatz entfernt. Dieser Vorgang wurde effizient mit der Filterfunktion durchgeführt, indem wir unsere gewünschten Daten selektiert und die überflüssigen Datensätze gelöscht

The screenshot shows the 'Nach Größe sortieren' (Sort by Size) dialog box in Microsoft Excel. The dialog is titled 'Nach Größe sortieren (aufsteigend)' (Sort by Size (ascending)). It features a list of numbers from 24 to 32. The number 28 is selected, indicated by a checkmark in the checkbox next to it. The 'Zahlenfilter' (Number Filter) option is selected under the 'Filtern' (Filter) section. The 'Suchen' (Search) field is empty. The 'OK' and 'Abbrechen' (Cancel) buttons are located at the bottom of the dialog.

Schritt	G	Ausführung	G	Zahl	G	Formelnformel	Formelartagen	Zellen	Leichen	Beurteilen	Auswerten
$f_x = \max(11; 1365)$											
E			E			H	I	J	K	L	
Bergien St & 4 Ave	29.12.2023	363	6	12	0,02	6,95	43	49	53	32	
Bergien St & 4 Ave	30.12.2023	364	7	12	0	14,54	38	44	47	40	
Bergien St & 4 Ave	31.12.2023	365	1	12	0	12,08	38	41	43	44	
								83			

4

2.4 Entwerfen eines Datenbank-Schemas

2.4.1 1.Normalform

Um die 1.Normalform anwenden zu können, haben wir erstmal sichergestellt, dass alle Werte in der Tabelle atomar sind. Dies bedeutet, dass jede Zeile nur genau einen Wert enthält und wir haben nachgeschaut, ob jede Spalte einen einheitlichen Datentyp hat. In unserer Tabelle haben wir festgestellt, dass dies der Fall ist : Die Spalte `Station` enthält denselben Ortsnamen "Bergen St Ave" in jeder Zeile, was atomar ist, aber redundant (Redundanz werden wir mithilfe der 2NF vermeiden). Die Spalte, `date` enthält immer ein einzelnes Datum, die Spalten: `"day of year"`, `"month of year"` und, `"day of week"`. enthalten immer eindeutig zuordnenbare Werte, sowohl auch die Temperaturen `"wind speed"`, `"precipitation"` immer numerische Einzelwerte sind, was sie ebenfalls atomar macht. Da die Tabelle alle Kriterien der 1NF erfüllt, haben wir sie im nächsten Schritt auf die 2.NF überprüft.

Station	Date	Precipitation	Windspeed	Min.temperature	Average temperature	Max temperature	Count
Bergen St&Ave	01.01.2023	0	10,07	42	50	56	38

Tabelle 1: 1.Normalform

2.4.2 Die 2.Normalform

Mithilfe der 1.NF ist es uns nun möglich die 2.NF herzuleiten: Zuerst ist es wichtig redundante Attribute, wie die Spalte `Station` oder abgeleitete Informationen wie `"day of week"`, `"day of year"` und `"month of year"` als überflüssig anzusehen, da diese Daten im `"Date"` schon enthalten wären. Nun ist es wichtig, dass alle Nicht-Schlüsselattribute vollständig vom gesamten Primärschlüssel abhängen. Das bedeutet, wir erstellen mehrere Tabellen um die Daten nach Redundanz zu ordnen. Beispielsweise sind Spalten wie `SSStation`, `"date"` oder `"wind speed"` unabhängig von der Temperatur. Jetzt entstehen keine partiellen Abhängigkeiten mehr. Die Tabelle Verleih enthält demnach `StationID`, `Date` und `count` (die Menge der erfassten Messungen). `ID` ist hier der Primärschlüssel und `Date`, `StationID` bilden hier die Schlüsselattribute. `Count` allerdings ist ein Nicht-Schlüsselattribut, da die Menge (`count`) nicht zur Identifikation des Datensatzes beiträgt und völlig abhängig von den anderen Attributen ist. Hier vermeiden wir Redundanz indem wir die Station in eine eigene Tabelle auslagern, weil die Station immer gleichbleibt.

Die Tabelle Temperatur hat TemperatureID als Primärschlüssel und Date, StationID den Schlüsselattributen zugewiesen. Die Tabelle dient ausschließlich dazu, die Wetterdaten (wie die minimale, durchschnittliche und maximale Temperatur) ersichtlich zu machen. Die Attribute Date und StationID werden hier ebenfalls ausgewiesen, da diese bei der Zuordnung der Daten wesentlich sind. Jede Messung oder Information sind immer an eine spezifische Station und an ein bestimmtes Datum gebunden. Zum Schluss haben wir bei der Tabelle "Wetter" das, WetterID als Primärschlüssel angesehen und Date, StationID bilden wir die Schlüsselattribute, da diese die Wetter Messungen (windspeed und precipitation) identifizieren.

Tabelle Verleih			
ID	Date	StationID	Count
1	2023-01-01	Bergen St & 4 Ave	38

Tabelle 2: Tabelle Verleih

Tabelle Station	
Station ID	Station
1	Bergen St & 4 Ave

Tabelle 3: Stationstabelle mit ID und Name

Tabelle Temperature					
TemperatureID	Date	StationID	min_temperature	average_temperature	max_temperature
1	2023-01-01	1	42	50	56

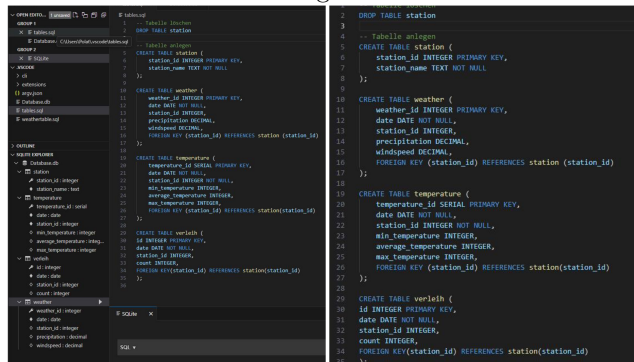
Tabelle 4: Tabelle Temperaturen

Tabelle Wetter				
WetterID	Date	StationID	wind_speed	precipitation
1	2023-01-01	1	10.07	0

Tabelle 5: Wetterdaten am 01.01.2023

2.5 Umsetzung des Schemas in SQL

Um unser Schema in SQL zu darzustellen, haben wir als Erstes das passende SQLite Programm für unseren Windows PC über den gegebenen Link heruntergeladen und über den Visual Studio Code darauf zugegriffen. Eine weitere erforderliche Erweiterung für SQLite haben ebenfalls heruntergeladen. Zunächst haben wir eine Datenbank erstellt, welche wir Database.db genannt haben und eine tables.sql File angelegt. Um die Tabellen dann zu erstellen haben wir folgende Befehle benutzt und ausgeführt:



```
1 CREATE DATABASE Database;
2 DROP TABLE station;
3
4 -- Tabelle anlegen
5 CREATE TABLE station (
6     station_id INTEGER PRIMARY KEY,
7     station_name TEXT NOT NULL
8 );
9
10 CREATE TABLE weather (
11     weather_id INTEGER PRIMARY KEY,
12     date DATE NOT NULL,
13     station_id INTEGER,
14     precipitation DECIMAL,
15     windspeed DECIMAL,
16     FOREIGN KEY (station_id) REFERENCES station(station_id)
17 );
18
19 CREATE TABLE temperature (
20     temperature_id INTEGER PRIMARY KEY,
21     date DATE NOT NULL,
22     station_id INTEGER,
23     min_temperature INTEGER,
24     average_temperature INTEGER,
25     max_temperature INTEGER,
26     FOREIGN KEY (station_id) REFERENCES station(station_id)
27 );
28
29 CREATE TABLE weleath (
30     id INTEGER PRIMARY KEY,
31     date DATE NOT NULL,
32     station_id INTEGER,
33     count INTEGER,
34     FOREIGN KEY (station_id) REFERENCES station(station_id)
35 );
```

Def.Tabelle mit DDL

Da wir bei SQLite im Visual Studio Code unseren Datensatz nicht als Tabelle importieren können, weil man hier mit den Befehlen *INSERT INTO* *tabelle (spalte 1, spalte 2,...)* und *VALUES* alle Werte manuell per Hand eingeben muss, haben wir uns entschieden für die Ermittlung der höchsten mittleren Temperatur in Grad Celcius aus unserem Datensatz das SQLite Online Programm zu verwenden.

2.5.1 Visual Code

Bei beiden Programmen ist der DDL-Teil nicht notwendig, da man bei Visual Studio Code in SQLite, selbst über das Terminal des PC unsere CSV-Datei nicht tabellarisch in unsere Database.db importieren kann (ohne einige verschiedene Erweiterungen) und bei SQLite Online ist der Import ganz ohne Befehle machbar. Trotzdem zeigen wir hier die notwendigen Befehle für die Ausführung eines DDL-Teils einmal auf:

- DDL-Teil:
- CREATE DATABASE - Datenbank erzeugen
- ALTER DATABASE - Datenbank modifizieren
- CREATE TABLE - Tabelle erzeugen
- ALTER TABLE - Tabelle modifizieren
- DROP TABLE - Tabelle löschen
- CREATE INDEX - Index erzeugen
- DROP INDEX - Index löschen

Abbildung 1: Enter Caption

SQLite erstellt automatisch aus unserem Datensatz eine Tabelle, welche die

direkte Weiterarbeit mit der SQL-Abfrage ermöglicht.

2.6 SQL Abfrage zur höchsten mittleren Temperatur

Hierfür haben wir erst einmal unseren Datensatz als gefilterte Tabelle im CSV-Format in SQLite Online importiert, dabei war es wichtig das richtige Trennzeichen (Delimiter) auszuwählen (Bei uns: Semikolon). Mit dem folgenden Befehl haben wir die höchste mittlere Temperatur festgestellt.

```
SQLite  SQLite.1
1 SELECT MAX((CAST(c10 AS FLOAT) - 32) * 5.0 / 9.0) AS max_average_temperature_celsius
2 FROM bike_sharing_data1
1 max_average_temperature_celsius
28.333333333333332
```

Befehlabfrage

Wir haben die Formel $(Fahrenheit - 32) \cdot \frac{5}{9} = Celsius$. für Fahrenheit in Celsius innerhalb des Befehls berücksichtigt und stellen fest, dass die höchste durchschnittliche Temperatur bei $28,33^{\circ}$ lag. c10 Stellt hier die Spalte äverage temperaturein unserem Datensatz da.

Literatur

Achim Klenke. *Wahrscheinlichkeitsrechnung*. Springer, 3. Auflage, 2013.

https://moodle.uni-bielefeld.de/pluginfile.php/527917/mod_resource/content/3/bike_sharing
<https://www.studysmarter.de/studium/mathematik-studium/statistik-studium/der-zentrale-grenzwertsatz>

<https://www.sqlite.org/>

<https://sqliteonline.com/>

Unterlagen der Vorlesung, *Statistik 2*. Universität Bielefeld

Unterlagen der Vorlesung *Computergestützte Methoden*. Universität Bielefeld, 2024.