Hochschule Bochum
Bochum University
of Applied Sciences

Fachbereich Wirtschaft
Prof. Dr. Christian Bockermann

# Data Science 1

Sommersemester 2023

## Hausarbeit

Die Prüfungsleistung zum Modul *Data Science 1* findet als Hausarbeit statt. Die Aufgabenstellung zur Hausarbeit finden Sie in diesem Dokument.

Für die Bearbeitung der Aufgabenstellung und die Erstellung Ihrer Hausarbeit steht wieder der Jupyter-Notebook Server zu Verfügung. Die Abgabe der Hausarbeit erfolgt dann als PDF-Export Ihres Jupyter-Notebooks. Das PDF Ihres Notebooks schicken Sie bis zum 31.7.2023 per Mail an: `christian.bockermann@hs-bochum.de`
Andere Formen der Abgabe sind nicht vorgehen.

Die Hausarbeit kann in Gruppenarbeit von bis zu drei Personen bearbeitet werden. In diesem Falle genügt *eine* Abgabe.

**In jedem Fall sind in der Hausarbeit am Anfang die Namen und Matrikelnummern aller daran beteiligten Personen zu vermerken (gilt auch für Einzelabgaben).**

Als Materialien können Sie sämtliche Unterlagen aus der Vorlesung und den Übungen mit benutzen, im Internet recherchieren oder weitere Bücher/Kurse mit verwenden. Geben Sie bitte bei Verwendung von umfangreicherem Programm-Code aus dem Netz (mehr als 3-4 Zeilen) die Quelle kurz mit an.

Die Verwendung von ChatGPT oder ähnlichen Hilfsmitteln ist nicht gestattet. Die Prüfungsordnung sieht für Verdachtsfälle die Möglichkeit mündlicher Nachprüfungen vor.

**Aufgabe 1** (Python Basics)

Anfang Juli startet auch dieses Jahr wieder das bedeutenste Radrennen der Welt – die Tour de France. Die Tour wird seit 1903 ausgetragen und ist damit eines der traditionsreichsten Sportereignisse unserer Zeit.
Es handelt sich bei der Tour de France um eine 3-wöchige Rundfahrt, d.h. drei Wochen lang finden Etappenrennen statt an denen Teams mit mehreren Fahrern teilnehmen. Der Fahrer, der über die gesamten Etappen die wenigste Zeit benötigt, gewinnt die Gesamtwertung. Gleichzeitig wird auch für jede der Etappen der Sieger ermittelt und es gibt Sonderwertungen für z.B. Bergfahrer usw.
Es gibt dazu natürlich auch historische Daten. Im Folgenden betrachten wir das Python-Modul `tdf`, das auf dem Notebook Server für Sie verfügbar ist. Diese Modul hat die Funktion `fahrer_liste()`, die eine Liste von Tupeln in folgendem Format bereitstellt:

```
(jahr, etappe, rang, name, team, strecke, zeit)
```

Die Komponente `jahr` bezieht sich auf das Jahr der Austragung der Tour und `etappe` auf die Etappe (idR 21 Etappen) dieser Austragung. `rang` ist die Platzierung des Fahrers auf dieser Etappe, `strecke` die zurückgelegte Entfernung und `zeit` die Zeit in Sekunden, die der Fahrer für die Etappe gebraucht hat.
Die Komponente `name` enthält den Namen des Fahrers und `team` den Namen des Teams, für das der Fahrer an den Start gegangen ist.

Hier ist ein kleines Beispiel, wie die Daten zu benutzen sind:

```
import tdf

fahrer = tdf.fahrer_liste()

jan = fahrer[50]
#  (1997, '1', '44', 'ULLRICH Jan', 'Team Telekom', 41.15,
                                    16799.0)
```

Wie in dem Python Code zu sehen ist, hat beispielsweise der Fahrer am Index 50 der Liste die folgenden Werte:

```
(1997, '1', '44', 'ULLRICH Jan', 'Team Telekom', 41.15, 16799.0)
```

Das heisst, es handelt sich bei dem Eintrag um die Teilnahme von Jan Ullrich an der 1. Etappe der Tour de France von 1997. Er fuhr dabei eine Strecke von 41,15 km in 16799 Sekunden und belegte den 44. Platz.

Für eine derartige Liste sollen Sie die folgenden Aufgaben lösen:

1. Schreiben Sie eine Funktion `jahre(liste)`, die als Parameter die obige Liste bekommt und die Menge der Jahre zurückgibt, für die es Tour-Teilnahmen in der Liste gibt. Das bedeutet insbesondere, das jedes Jahr nur einmal im Ergebnis vorkommt.

2. Schreiben Sie eine Funktion `teilnehmer(liste, jahr)`, die für die gegebene Liste und ein vorgegebenes Jahr die Menge der Namen aller Fahrer, die in diesem Jahr vorkommen zurückliefert.

3. Geben Sie eine Funktion **`anzahlFahrer(liste)`** an, die eine Liste von Tour-Teilnahmen in der obigen Form bekommt, und eine Liste mit Tupeln der Art

   `[ (jahr, anzahlFahrer), ...]`

   d.h. jedes Tupel besteht aus der Jahreszahl und der Anzahl der Fahrer in diesem Jahr. Achten Sie darauf, dass jeder Fahrer pro Jahr nur einmal gezählt wird.

4. Schreiben Sie ein Funktion **`teams(liste, jahr)`**, die die Menge der Team-Namen zurückgibt, die in dem angegebenen Jahr zur Tour angetreten sind.

5. Schreiben Sie eine Funktion **`fahrerVonTeam(liste, jahr, team)`**, die eine Menge mit den Namen aller Fahrer mit dem angegebenen Team-Namen und dem angegebenen Jahr zurückgibt.

6. Geben Sie eine Funktion **`gesamtwertung(liste, jahr, name)`** an, die für die Liste der Tupel, ein vorgebenes Jahr und den übergebenen Fahrernamen ein Tupel mit den folgenden Werten berechnet:

   `(jahr, name, anzahlEtappen, gesamtZeit)`

   Tupel, die keine Zeitangabe haben (z.B. weil ein Fahrer eine Etappe nicht beendet hat), sollen nicht in die Betrachtung einfließen. In der Komponente **`zeit`** steht in diesem Fall der Wert **`None`**.

7. Entwickeln Sie eine Funktion **`teamwertung(liste, jahr)`**, die eine Liste der folgenden Art berechnet:

   `[(jahr, team, teamZeit), ...]`

   Dabei soll der Wert **`teamZeit`** die Gesamtzeit des dritt-besten Fahrers aus dem Team darstellen. Es werden dabei nur Fahrer betrachtet, die mindestens 21 Etappen gefahren sind. Teams, die weniger als 3 Fahrer mit mindestens 21 Etappen haben, sollen in der Liste nicht auftauchen.

   **Hinweis 1:** Es ist sinnvoll auch dieses Problem zunächst weiter zu erlegen und als Zwischenlösung eine Funktion **`teamZeit(liste, jahr, team)`** zu schreiben, die für einen gegebenen Teamnamen, die Anzahl der Etappen und die Gesamtzeit des dritt-besten Fahrers ermittelt.

   **Hinweis 2:** Mit der Funktion **`list.sort(eineListe)`** kann eine Liste sortiert werden. Dazu kann mit einer **`key`**-Funktion der Parameter angegeben werden, nach dem sortiert werden soll. Zum Beispiel:

```
liste = [ ('a', 2), ('B', 1), ('C', 6) ]

# Sortiere die obige Liste und betrachte dabei
# die zweite Komponente der Tupel:
list.sort(liste, key = lambda x: x[1])
```

**Aufgabe 2** (Pandas und Statistiken)

Die Datei `Kurse/DataScience1/data/tdf/etappen.csv` enthält einen größeren Datensatz der Etappen-Ergebnisse der Tour de France seit 1903. Die Daten enthalten zusätzlich für einige der Fahrer auch das Gewicht und die Größe.
Das lässt natürlich Spielraum für einige Analysen zur Entwicklung der Fahrertypen über die vergangene Zeit.

| Jahr | Etappe | Rang | Fahrer | Team | Zeit | Distanz | Gewicht | Größe |
|---|---|---|---|---|---|---|---|---|
| 2022 | 18 | 1 | VINGEGAARD Jonas | Jumbo-Visma | 14390.1 | 143.2 | 60.0 | 1.75 |
| 2022 | 5 | 1 | CLARKE Simon | Israel - Premier Tech | 11615.1 | 157.0 | 63.0 | 1.75 |
| 2022 | 11 | 1 | VINGEGAARD Jonas | Jumbo-Visma | 15482.1 | 151.7 | 60.0 | 1.75 |
| 2022 | 12 | 1 | PIDCOCK Thomas | INEOS Grenadiers | 17724.1 | 165.1 | 58.0 | 1.7 |
| 2022 | 15 | 1 | PHILIPSEN Jasper | Alpecin-Deceuninck | 16047.1 | 202.5 | 75.0 | 1.76 |
| 2022 | 20 | 1 | VAN AERT Wout | Jumbo-Visma | 2879.0 | 40.7 | 78.0 | 1.9 |
| 2022 | 4 | 1 | VAN AERT Wout | Jumbo-Visma | 14496.1 | 171.5 | 78.0 | 1.9 |
| 2022 | 21 | 1 | PHILIPSEN Jasper | Alpecin-Deceuninck | 10712.1 | 115.6 | 75.0 | 1.76 |
| 2022 | 10 | 1 | CORT Magnus | EF Education-EasyPost | 11930.1 | 148.1 | 68.0 | 1.84 |
| 2022 | 8 | 1 | VAN AERT Wout | Jumbo-Visma | 15186.1 | 186.3 | 78.0 | 1.9 |

Die Bedeutung der meisten Spalten ist eigentlich selbsterklärend. Die Daten enthalten die Ergebnisse aller Etappen, d.h. die benötigte Zeit (in Sekunden) der einzelnen Fahrer. Dazu die Platzierung (Rang), die Nr. der Etappe und das Jahr. In einigen Fällen enthält die Spalte *Rang* Werte wie DNF (*did not finish*), DSQ (*disqualified*) oder OTL (*out of time limit*). Dies sind Fahrer, die die entsprechende Etappe nicht beenden konnten oder nachträglich disqualifiziert wurden.
Die Spalte *Team* enthält den Namen des Teams, für den der Fahrer fährt. Sponsoren-Teams waren erst seit ca. 1962 populär, davor gab es häufig Nationalmannschaften.
Aggregiert man z.B. die Zeiten pro Fahrer pro Jahr auf, erhält man die Gesamtzeiten und kann so für ein Jahr die Gesamtwertung ausrechnen.

Wir sind in dieser Aufgabe auf der Suche nach Antworten auf Fragen wie z.B.:

- Wie hat sich die Teilnehmerzahl der Teams/Fahrer über die Jahre entwickelt?
- Wie hat sich die Gesamtlänge der Rundfahrt entwickelt? In den ersten Jahren gab es Etappen, die waren über 400 km lang...

Hintergrund dieser Aufgabe ist es, dass Sie sich mit einem unbekannten Datensatz vertraut machen und mit Hilfe von Pandas untersuchen, welche Informationen aus den Daten herausgesucht werden können.

## Die Aufgaben:

1. Zunächst sollen ein paar generelle Informationen berechnet werden:
   - Wieviele verschiedene Fahrer gibt es in dem Datensatz? Über welchen Zeitraum sind überhaupt Daten verfügbar? Für welche Zeiträume (Jahre) gibt es z.B. *keine* Daten – ggf. warum?
   - Gibt es fehlende Werte? Gibt es Duplikate in den Daten? Welche Spalten enthalten fehlende Werte und wie viele?
   - Wieviele Etappen hatte die Tour de France in den verschiedenen Jahren? Erstellen Sie einen Plot mit der Anzahl der Etappen pro Jahr.
2. Erstellen Sie ein Diagramm, das die Anzahl der teilnehmenden Fahrer pro Jahr darstellt.
3. Erstellen Sie einen DataFrame, der für jedes Jahr einen Eintrag mit dem Gesamtsieger (alle Etappen gefahren, minimale Gesamtzeit) enthält. Jede Zeile soll dabei das Gewicht, die Größe, die gebrauchte Zeit und die insgesamt zurückgelegte Entfernung enthalten.
4. Berechnen Sie für die Gesamtsieger jeweils den Body Mass Index (BMI) auf Basis der Körpergröße und des Gewichts. Wie hat sich der BMI der Sieger über die Jahre entwickelt - erstellen Sie einen Plot dazu.
5. Betrachten wir als nächstes die Entwicklung der Durchschnittsgeschwindigkeiten. Dazu wollen wir für jede Etappe die Durchschnittsgeschwindigkeit des schnellsten und langsamsten Fahrers ermitteln. Plotten Sie die beiden Durchschnitte über alle Jahre und Etappen.

   Ermitteln Sie auch den die höchste und niedrigste Durchschnittsgeschwindigkeit pro Jahr und erstellen Sie einen Plot dazu.
6. Spannend ist häufig auch die Betrachtung kleinerer Ausschnitte der Daten. Stellen Sie eine Liste der Fahrer und deren Anzahl von Teilnahmen an der Tour de France zusammen. Wählen Sie einen der Fahrer mit mehr als 15 Teilnahmen aus und analysieren Sie dessen Geschichte:
   - Bei wie vielen seiner Teilnahmen ist er bis nach Paris gekommen, hat also alle Etappen der jeweiligen Tour erfolgreich beendet?
   - Wie hat sich die Platzierung und die Durchschnittsgeschwindigkeit des Fahrers im Laufe seiner Teilnahmen entwickelt? Überlegen Sie sich, wie Sie diese Entwicklung in Zahlen oder graphisch am geeignetsten darstellen können.

## Hinweis zur Bearbeitung

Es geht bei der Bearbeitung dieser Aufgaben nicht nur um die reine Programmierung in Python. Ziel ist es, die Daten entlang der Teilaufgaben zu analysieren und die Ergebnisse in einem gewissen Rahmen zu interpretieren.

Dazu gehört zu jeder Teilaufgabe, dass Sie kurz skizzieren, wie Sie vorgehen wollen, welche Teil-DataFrames sie ggf. berechnen wollen und was Sie am Ergebnis ggf. kritisch betrachten (z.B. Datenqualität, etc.). Auch dafür haben Sie in Data Science und Kursen wie Wirtschaftsstatistik Methoden und Werkzeuge kennengelernt.
Überlegen Sie sich zudem, wie Sie die Ergebnisse Ihrer Analysen überprüfen können.