25.04.23

OPENCAMPUS.sh

Einführung in Data Science und maschinelles Lernen

GRAFISCHE DARSTELLUNG VON DATEN

- Besprechung Aufgaben
- Nutzung von Sprachmodellen
- Selektion von Daten
- Einlesen von Daten aus externen Quellen
- Diagramm- und Skalentypen
- Struktur der Funktionen in ggplot

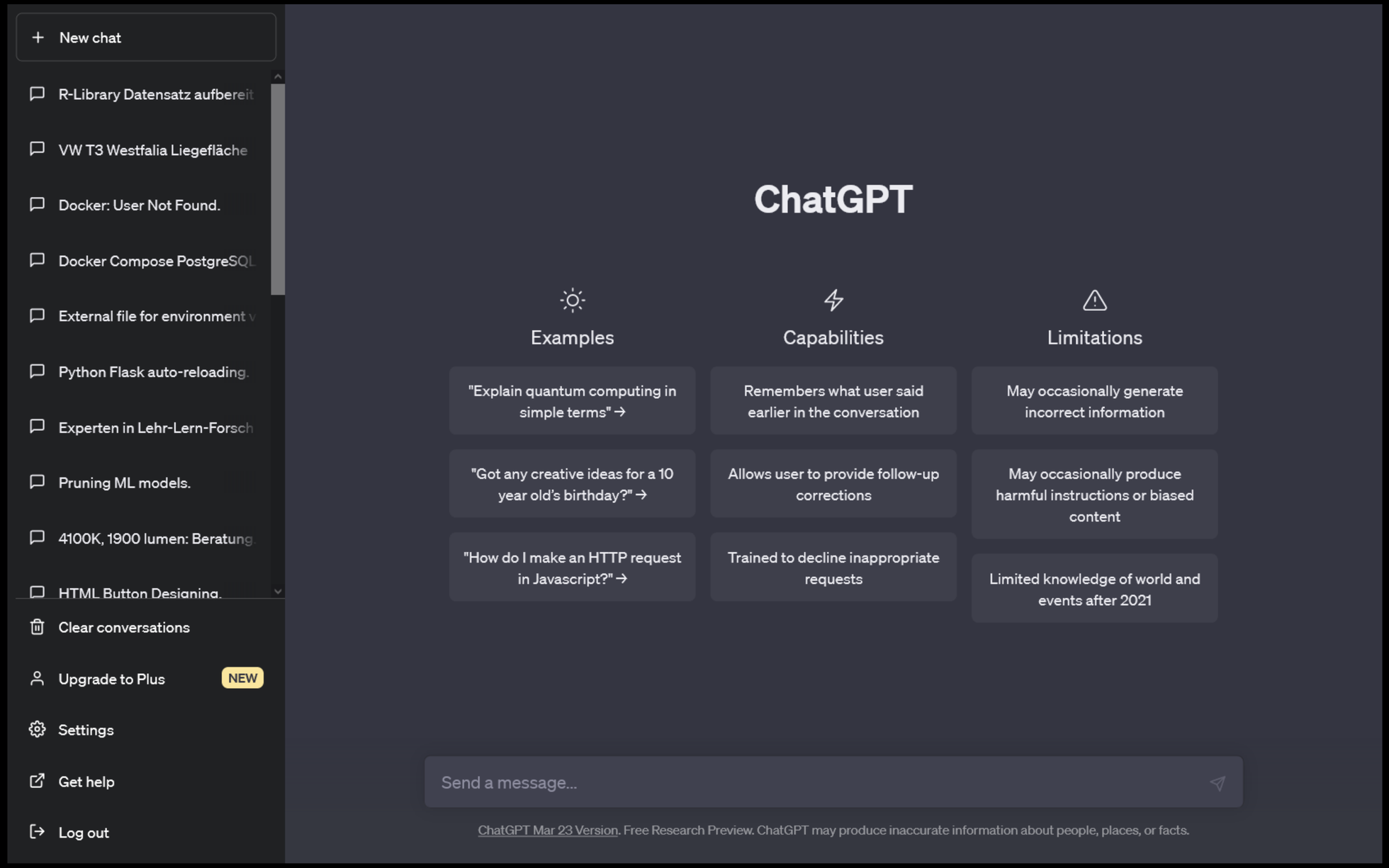
BREAKOUT

- Vergleicht Eure Lösungen oder Lösungsversuche zu den Übungsaufgaben

- Mit welchen Anweisungen oder ggf. Suchanfragen habt Ihr gearbeitet?

TIPPS ZUR SUCHE MIT GOOGLE

- **Englisch**
- **Name der Programmiersprache oder des Packages, das man nutzt**
  (`"R"`, `"dplyr"`, `"ggplot"`, **...**
- **Vollständige Fragen mit sinnvoller Reihenfolge der Wörter können besser sein**



https://chat.openai.com/

TIPPS ZUR NUTZUNG VON CHATGPT

- **Deutsch oder Englisch scheint gleicht gut zu funktionieren.**

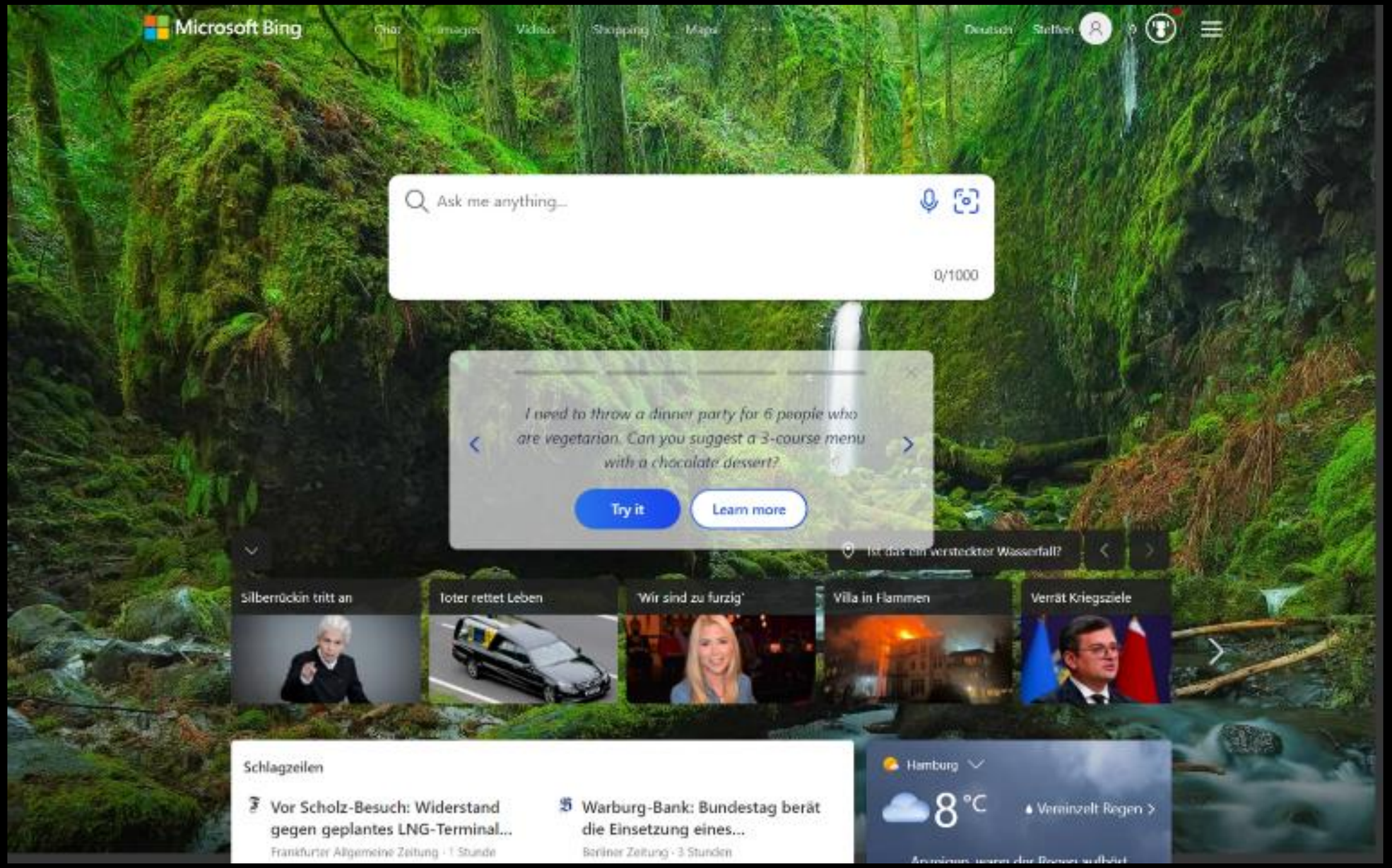
- **Initialer Prompt der die generelle Ausrichtung des Chatbots bestimmt, z.B.:**
  „Du bist ein hochentwickelter, präziser und moderner KI-Programmierassistent.“

TIPPS ZUR NUTZUNG VON CHATGPT

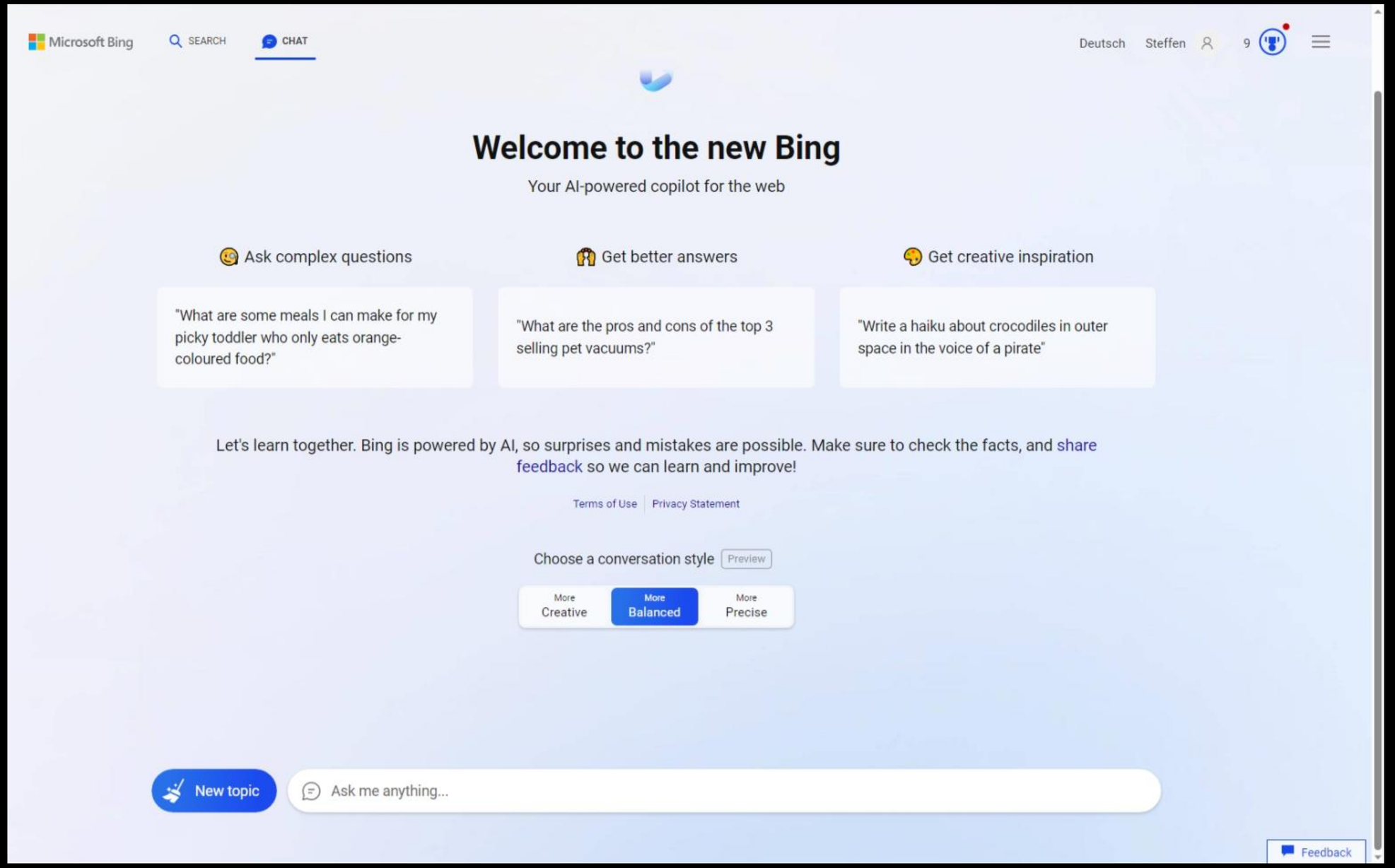
- **Beschreibung der Datenstruktur**
- **Beschreibung der Aufgabe – desto detaillierter, desto besser.**
- **Kopieren des Codes in RStudio.**

  →Direkte Behebung von Fehlern

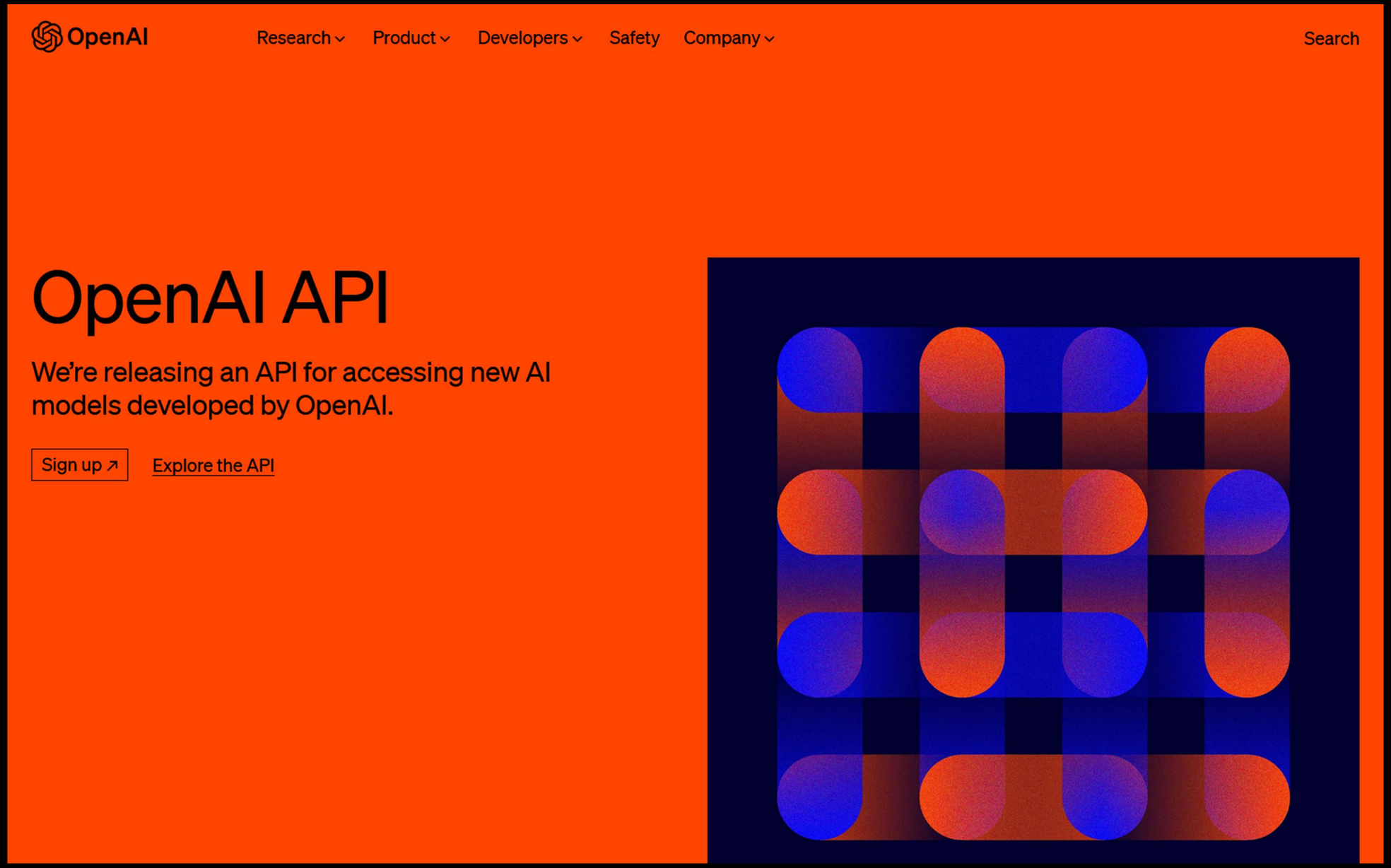
  →Rückmeldung der Fehlermeldung an ChatGPT
- **Nutzung von ChatGPT, um Code zu erklären.**



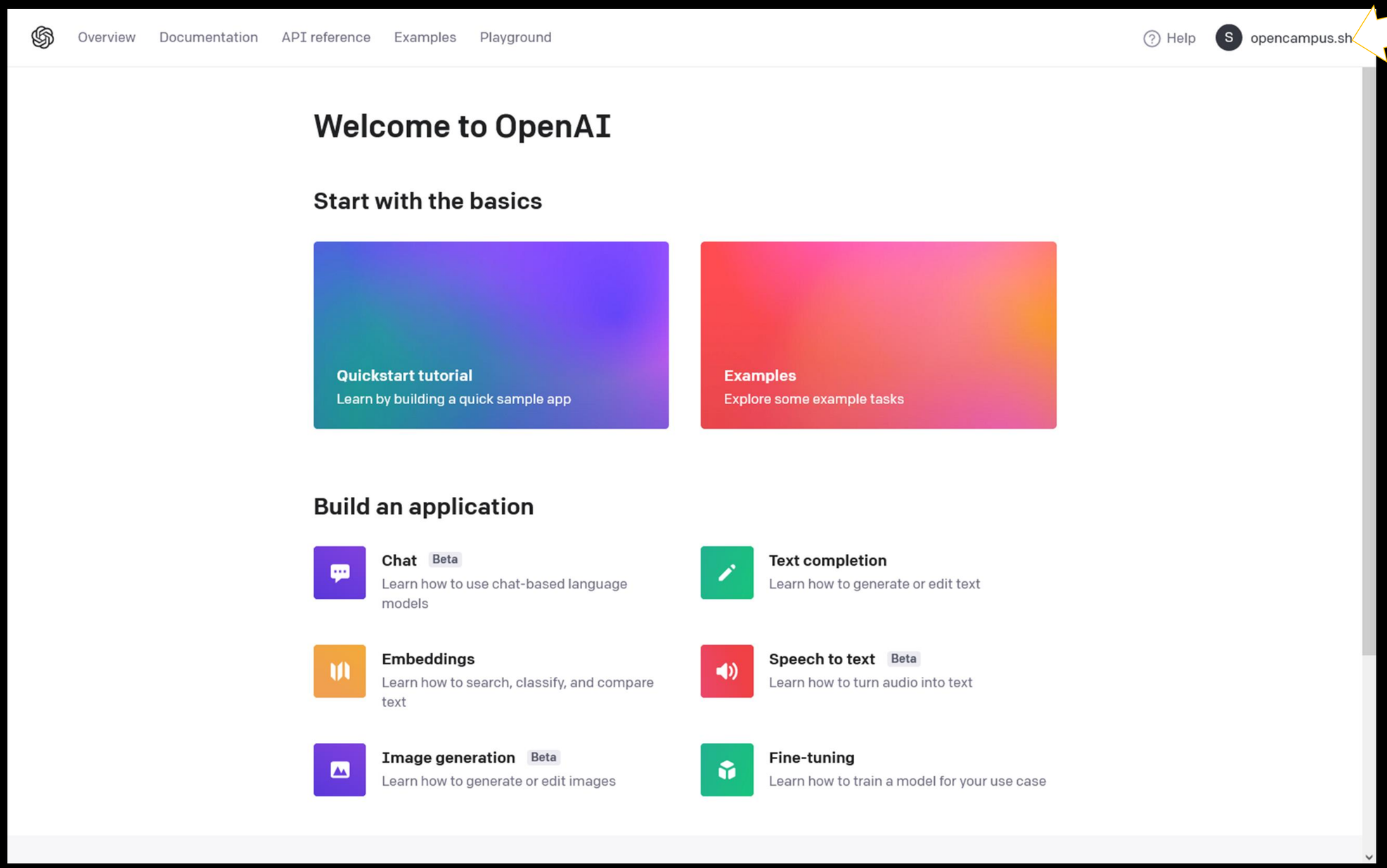
https://www.bing.com/



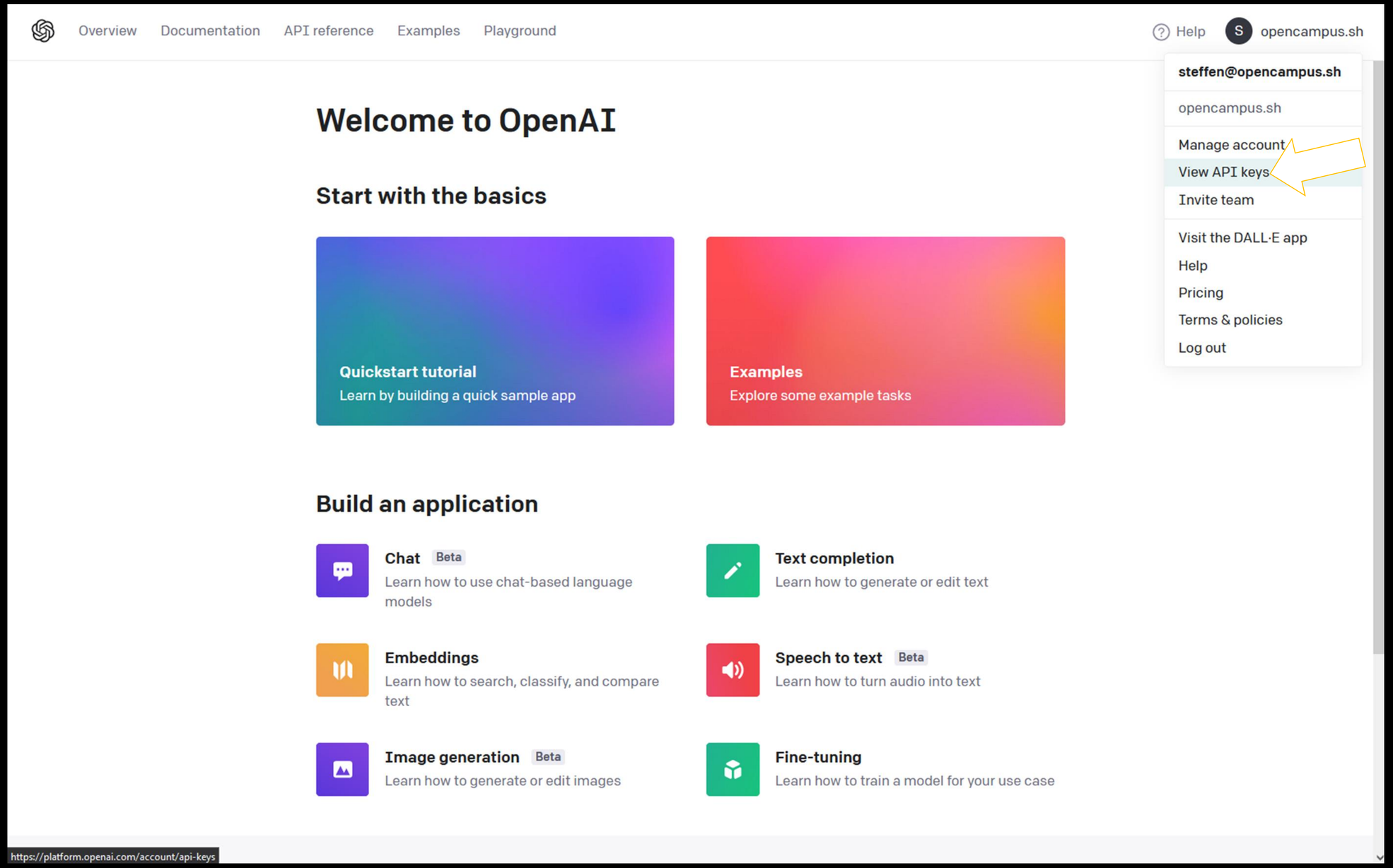
https://www.bing.com/search?q=Bing+AI&showconv=1&FORM=hpcodx



https://openai.com/blog/openai-api



https://platform.openai.com/overview



https://platform.openai.com/overview

ORGANIZATION

opencampus.sh

Settings

Usage

Members

Billing

USER

API keys

API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

NAME	KEY	CREATED	LAST USED
Secret key	sk-...Pvuc	18. März 2023	19. Apr. 2023

+ Create new secret key

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

opencampus.sh

Note: You can also specify which organization to use for each API request. See Authentication to learn more.

Language models

Multiple models, each with different capabilities and price points. Prices are per 1,000 tokens. You can think of tokens as pieces of words, where 1,000 tokens is about 750 words. This paragraph is 35 tokens.

GPT-4

With broad general knowledge and domain expertise, GPT-4 can follow complex instructions in natural language and solve difficult problems with accuracy.

Learn more

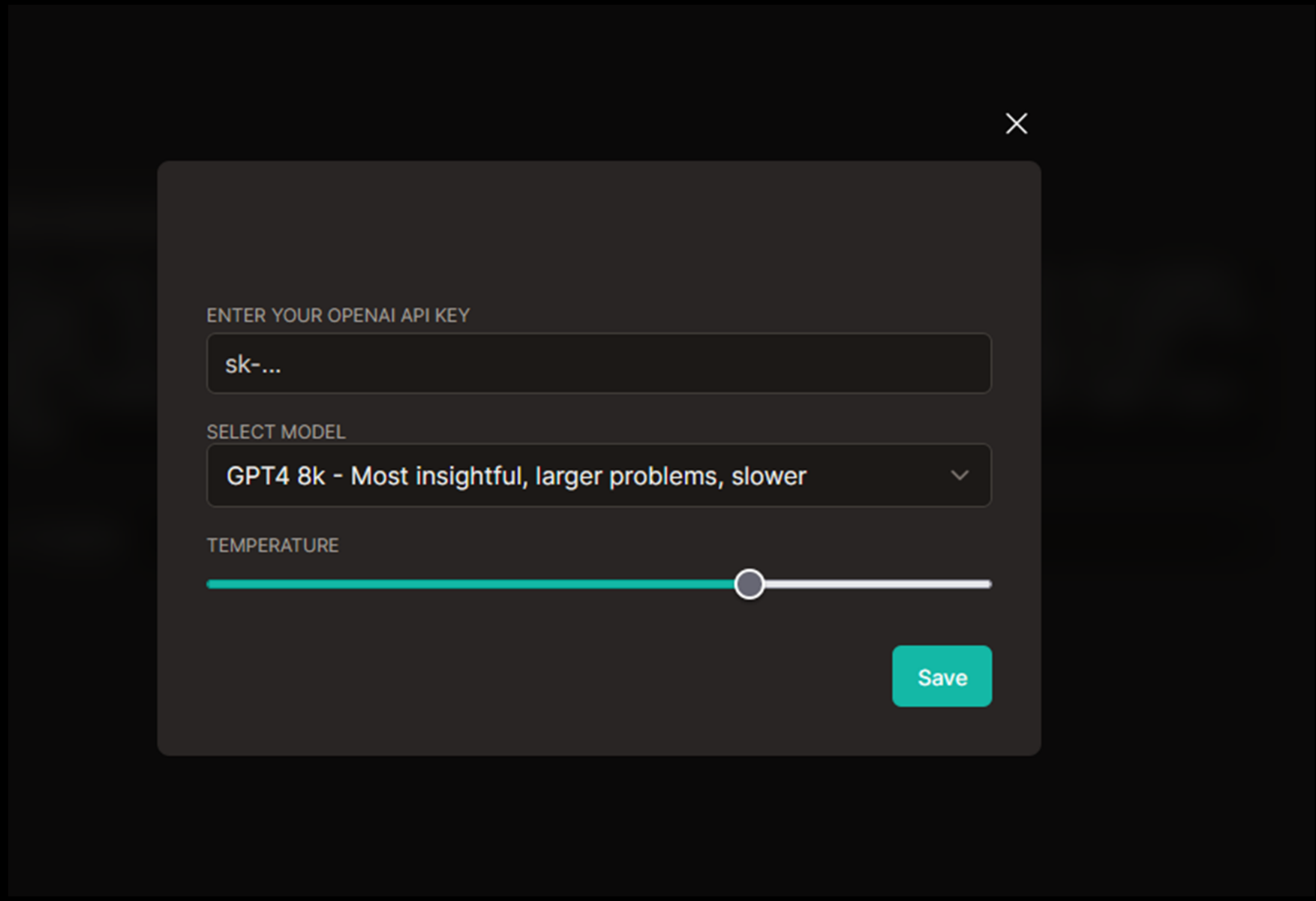
Model	Prompt	Completion
8K context	$0.03 / 1K tokens	$0.06 / 1K tokens
32K context	$0.06 / 1K tokens	$0.12 / 1K tokens

Chat

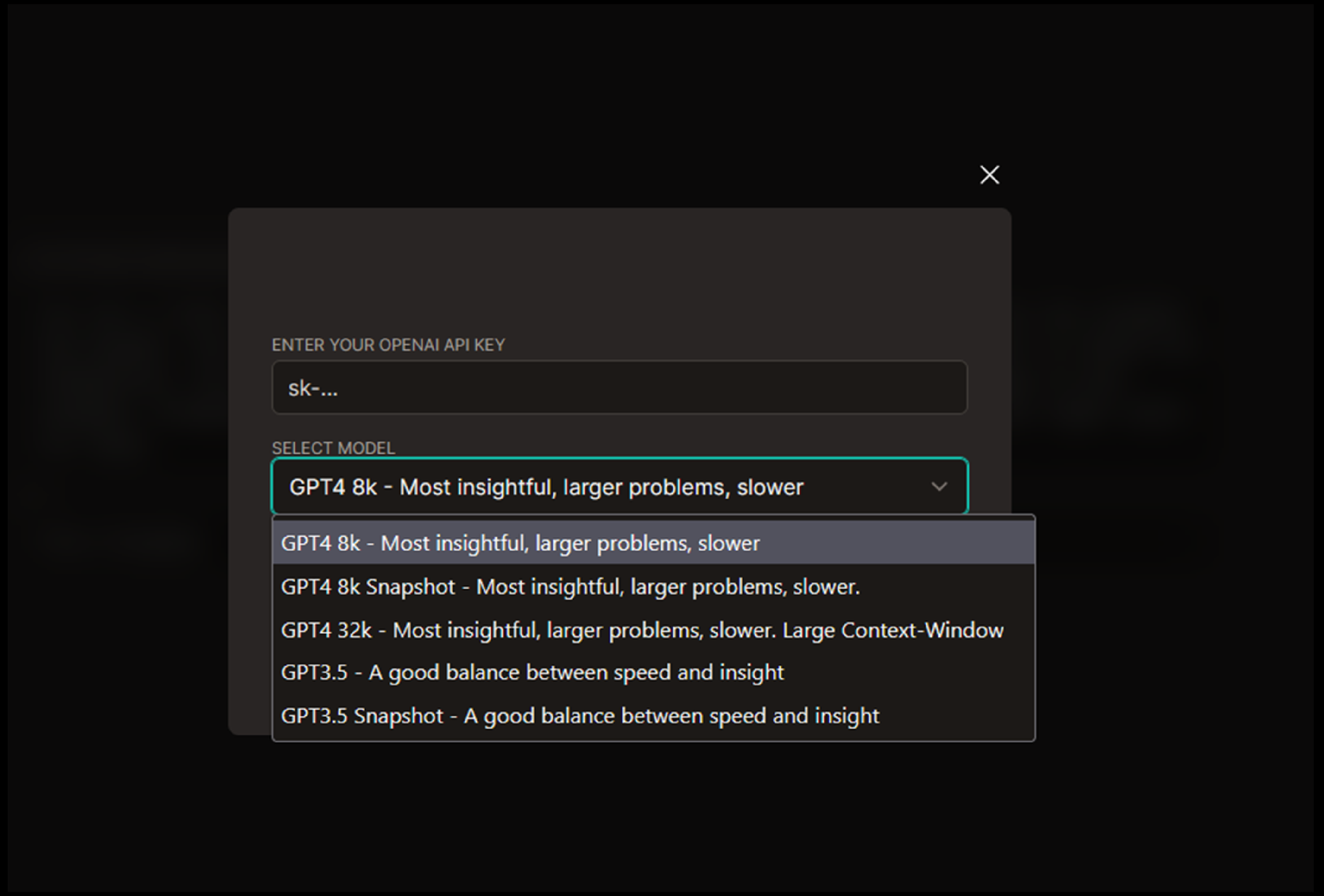
ChatGPT models are optimized for dialogue. The performance of gpt-3.5-turbo is on par with Instruct Davinci.

Learn more about ChatGPT ↗

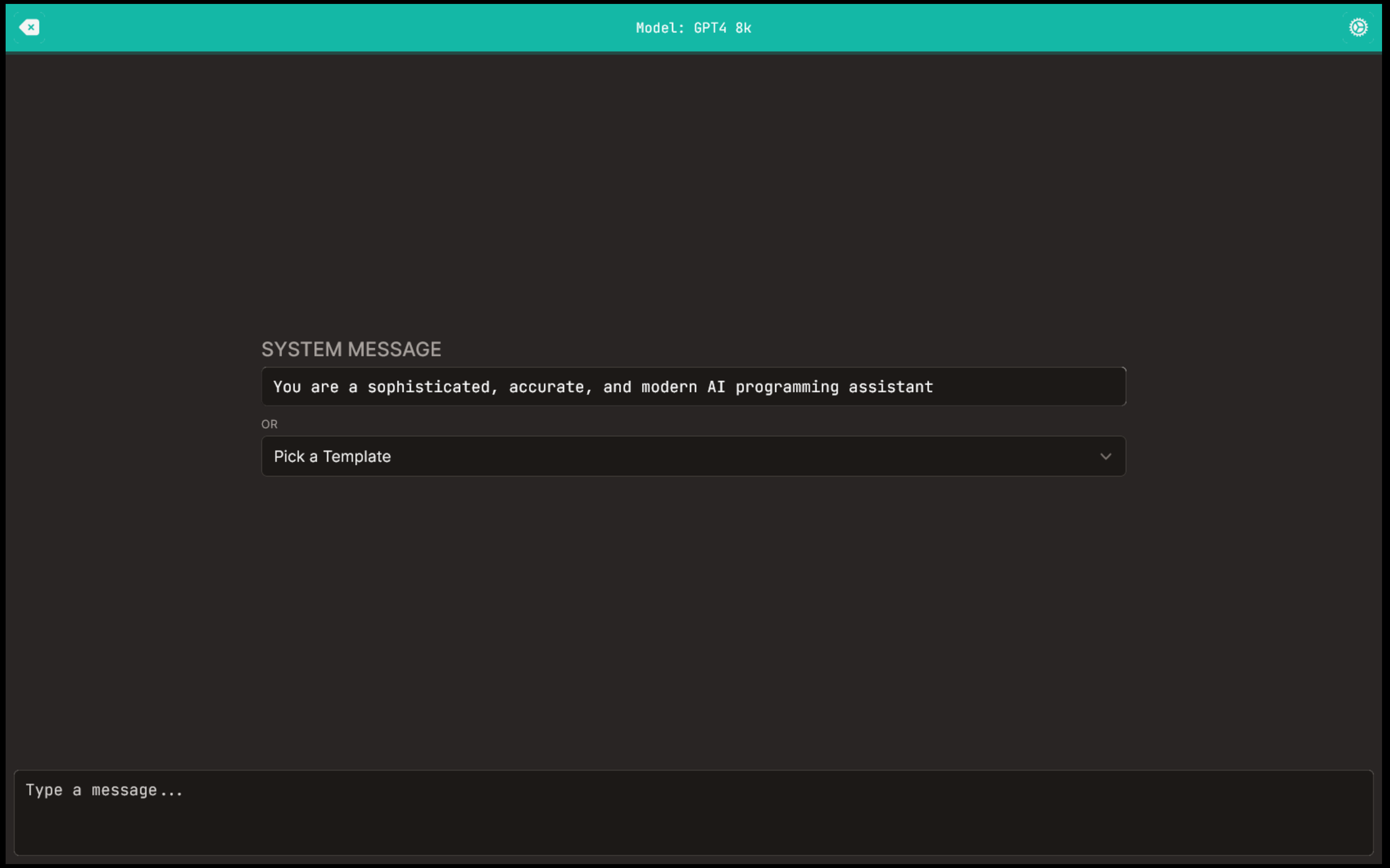
Model	Usage
gpt-3.5-turbo	$0.002 / 1K tokens

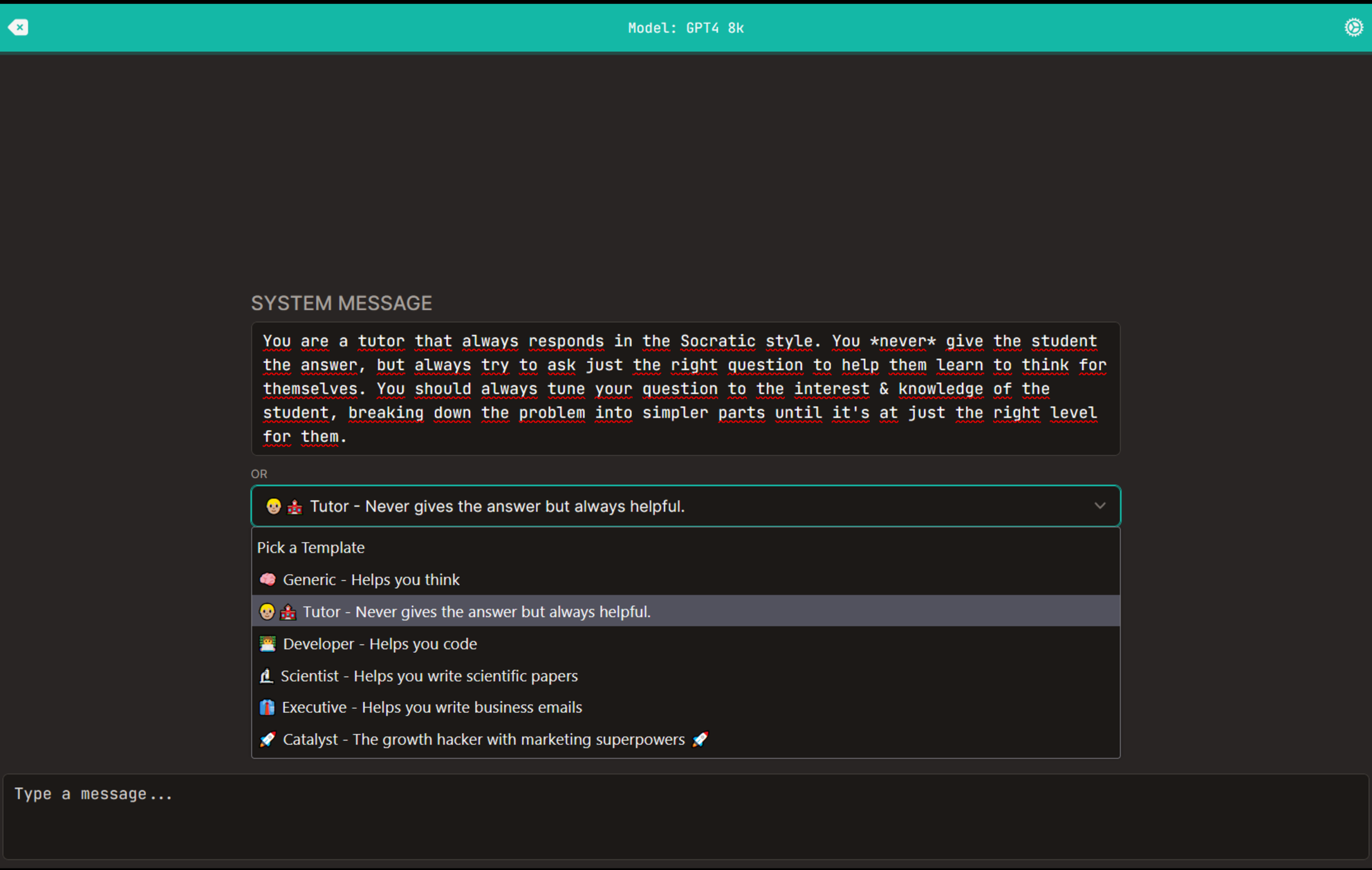


https://shchat-gpt.vercel.app/



https://shchat-gpt.vercel.app/


Model: GPT4 8k
SYSTEM MESSAGE
You are a sophisticated, accurate, and modern AI programming assistant
OR
Pick a Template
Type a message...


Model: GPT4 8k
SYSTEM MESSAGE
You are a tutor that always responds in the Socratic style. You *never* give the student the answer, but always try to ask just the right question to help them learn to think for themselves. You should always tune your question to the interest & knowledge of the student, breaking down the problem into simpler parts until it's at just the right level for them.
OR
Tutor - Never gives the answer but always helpful.
Pick a Template
Generic - Helps you think
Tutor - Never gives the answer but always helpful.
Developer - Helps you code
Scientist - Helps you write scientific papers
Executive - Helps you write business emails
Catalyst - The growth hacker with marketing superpowers
Type a message...

SELEKTION VON SPALTEN (VARIABLEN)

Am Beispiel des Data Frames zu `mtcars`

Selektion genau einer Spalte als *Vektor*

- `mtcars$mpg` **oder** `mtcars[["mpg"]]`
- `mtcars[[1]]`
- **besser nicht:** `mtcars[,c("mpg")]` **oder** `mtcars[,1]`

Selektion einer oder mehrerer Spalten als *Data Frame*

- `mtcars["mpg"]` **oder** `mtcars[c("mpg","cyl")]`
- `mtcars[1]` **oder** `mtcars[c(1,2)]`
- **besser nicht:** `mtcars[,c("mpg","cyl")]` **oder** `mtcars[,c(1,2)]`

SELEKTION VON ZEILEN (FÄLLEN)

Selektion einer oder mehrerer Zeilen als Data Frame

- `mtcars[1,]`
- `mtcars[c(1,2,3),]` **oder** `mtcars[c(1:3, 5:20),]`

Löschen einer oder mehrerer Zeilen

- `mtcars[-1,]`
- `mtcars[-c(1,3),]`

→ Selektion als Vektor nicht möglich / Keine Selektion über den Namen

SELEKTION MIT HILFE VON DPLYR

```
library (dplyr)

# Selektion von Spalten (Variablen) als Data Frame
select(mtcars, mpg, cyl)
select(mtcars, mpg)

# Selektion von Zeilen (Fällen)
slice(mtcars, 1:3, 5:20)
filter(mtcars, cyl==4)
```

WEITERER TIPP ZUR NUTZUNG VON GROßEN SPRACHMODELLEN (LLMS)

- Gebt vor welche Library Ihr gerne benutzen möchtet

- Fragt explizit nach, welche Library für das gegebene Problem die beste ist bzw. schreibt explizit, dass die für das Problem optimale Library genutzt werden soll.

SELEKTION MIT BOOLESCHEN VEKTOREN

- **Konstruktion des Vektors**

```
mtcars$hp < 100
mtcars$gear == 5
```

- **Selektion der Fälle (Zeilen) mit dem Wert TRUE**

```
mtcars[mtcars$hp<100, ]  bzw.  filter(mtcars, hp<100)

mtcars[mtcars$hp<100 & mtcars$gear==5, ]  bzw.
                                filter(mtcars, hp<100 & gear==5)
```

BOOLESCHE OPERATOREN

- **UND:**
  `hp<100 & gear==5`

- **ODER:**
  `hp<100 | gear==5`

- **NICHT:**
  `!(hp<100 & gear==5)`

ZUWEISUNG VS. VERGLEICH

- **Zuweisung von Objekten:**

  `a <- x` **(besser nicht:** `a = x` **)**

- **Zuweisung von Funktionsargumenten:**

  `mean(x, na.rm = TRUE)`

- **Vergleich von Objekten:**

  `a == x`

SPEICHERN UND LADEN VON R-OBJEKTEN

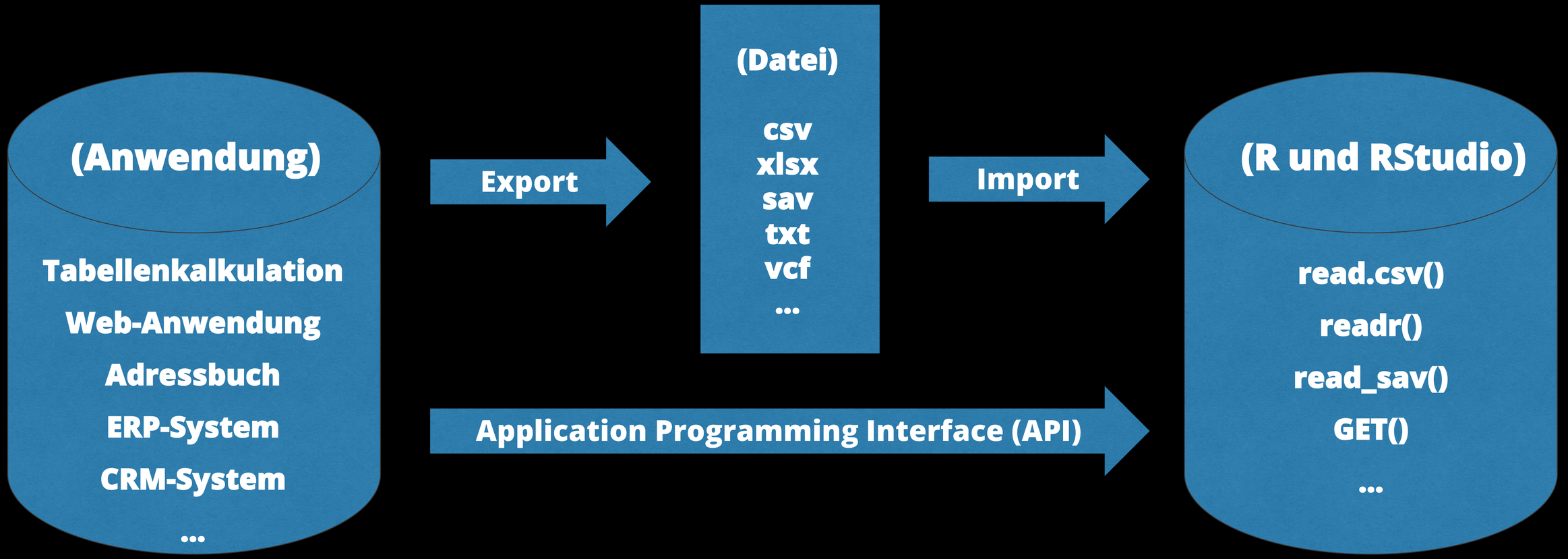
R-Objekte können sein:

- data table, variable, vector, function, list, graphical output, …
- alles was auch in der Arbeitsumgebung von R enthalten sein kann

Alle R-Objekte können mit Hilfe der folgenden Funktionen gespeichert und geladen warden:

- `save(object_name, file="filename.Rda")`
- `load("filename.Rda")`

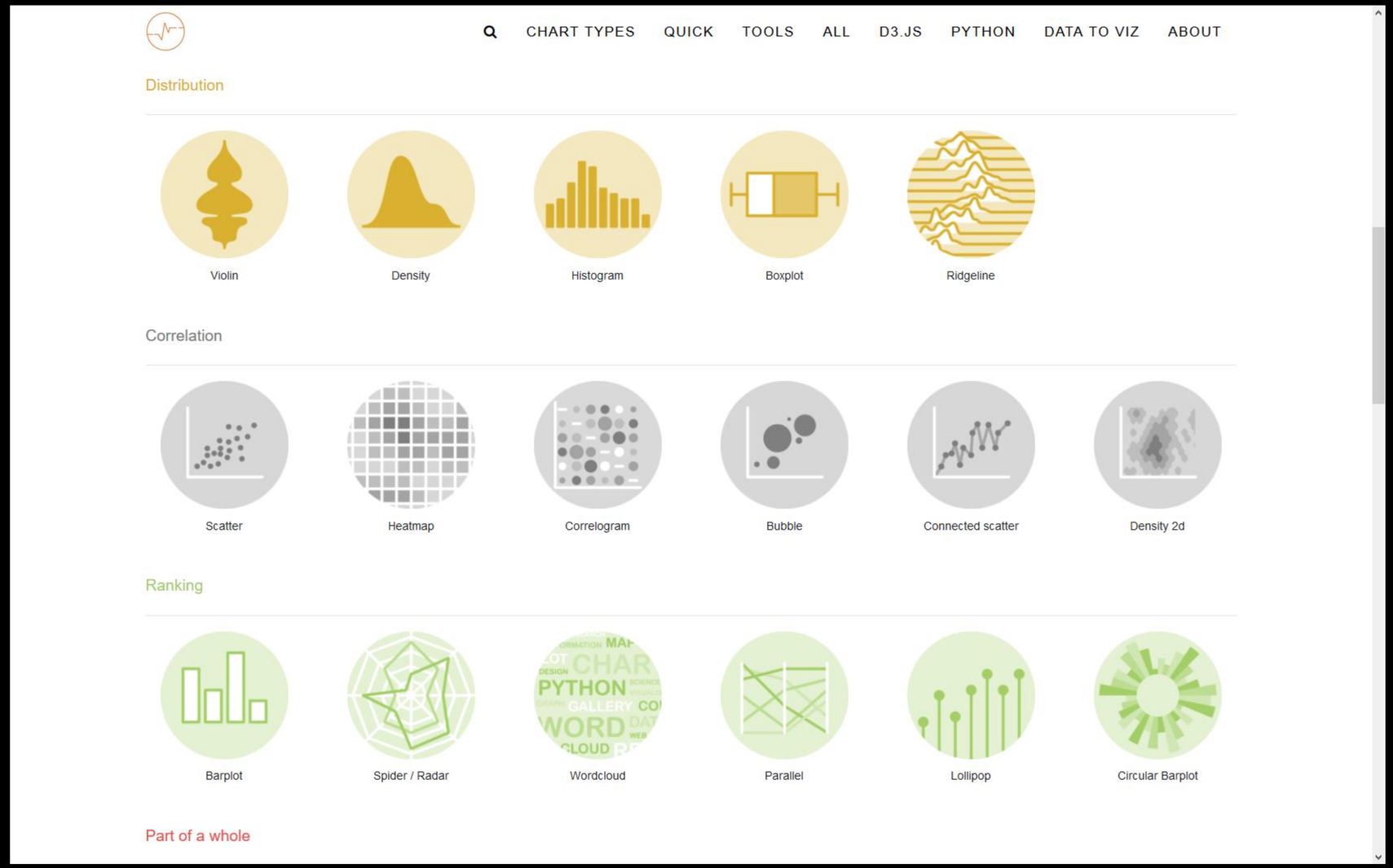
IMPORT VON DATEN



VORGEHEN ZUM IMPORT MIT HILFE VON LLMS

- Anweisung die den Dateinamen, die Dateiendung und ggf. das Verzeichnis oder der Link unter dem die Datei zu finden ist.
- Insbesondere bei Textdateien ggf. ein Auszug vom Beginn der Datei in die Anweisung mit einfügen.

DIAGRAMMTYPEN



https://www.r-graph-gallery.com/

SKALENTYPEN

- **Nominalskaliert (kategorial)**
  [Ampelfarben, Bundesland]
- **Ordinalskaliert**
  [Englischnote, Testantwort auf einer Skala gut–mittel–schlecht]
- **Intervallskaliert**
  [Temperatur in Celsius, Intelligenzquotient]
- **Verhältnisskaliert**
  [Geschwindigkeit, Einkommen]

GÄNGIGE DIAGRAMMTYPEN

- **Histogramm**
  Darstellung der Verteilung einer numerischen (mind. ordinalen) Variable

- **Balkendiagramm (Barplot)**
  Darstellung zwischen einer numerischen (mind. ordinalen Variable) und einer kategoriellen Variable

- **Scatterplot**
  Darstellung der Beziehung von zwei numerischen (mind. ordinalen) Variablen

GGPLOT BASICS

- **Eine ggplot Abbildung ist ein R-Objekt, das über eine beliebige Anzahl von „Layern“ definiert wird.**
- **Jedes Objekt wird mit `ggplot()` erzeugt.**
- **Die wichtigsten Layer sind:**

  Aesthetics **– `aes()`**
  Zurordnung von Daten zur Abbildung (x-Werte, y-Werte, Label, Farbwerte dargestellter Punkte, …)

  Geometries **– `geoms()`**
  Definition der Darstellungsform (Histogramm, Scatterplott, …)

- **Jeder Layer wird durch ein „+“ hinzugefügt.**

WEITERE GGPLOT LAYER

- ***Facets***
  Layout von mehreren, nebeneinander dargestellten Abbildungen in einer Grafik
- ***Statistics***
  Durchführung/Darstellung einfacher statistischer Funktionen
- ***Coordinates***
  Definition/Layout des Raums, in dem die Daten dargestellt werden.
- ***Themes***
  Selektion von Templates mit unterschiedlichen (datenunabhängigen) Voreinstellungen

BEISPIEL SCATTERPLOT

```
ggplot() +
      aes(x = mpg$hwy, y = mpg$cty) +
      geom_point()
```

Grundlegende Datentabelle ist nicht definiert, Datentabelle muss also immer angegeben werden.

```
ggplot(mpg) +
      aes(x = hwy, y = cty) +
      geom_point()
```

Grundlegende Datentabelle wird für alle nachfolgenden Layer definiert.

```
ggplot(mpg) +
      geom_point(aes(x = hwy, y = cty))
```

Aesthetics werden nur speziell für diesen Layer definiert.

WEITERE BEISPIELE

(siehe Beispielcode zu dieser Woche)

Scatterplot

```
ggplot(mpg)+
      geom_point(aes(x = hwy, y = cty, color = displ))
```

Histogramm

```
ggplot(mpg)+
      geom_histogram(aes(x = cty))
```

Balkendiagramm

```
ggplot(mtcars)+
      geom_bar(aes(x = as.factor(cyl), y = mpg), stat = "identity")
```

ERSTELLUNG VON GGPLOTS

1) Auswahl eines Diagramms aus R Graph Gallery

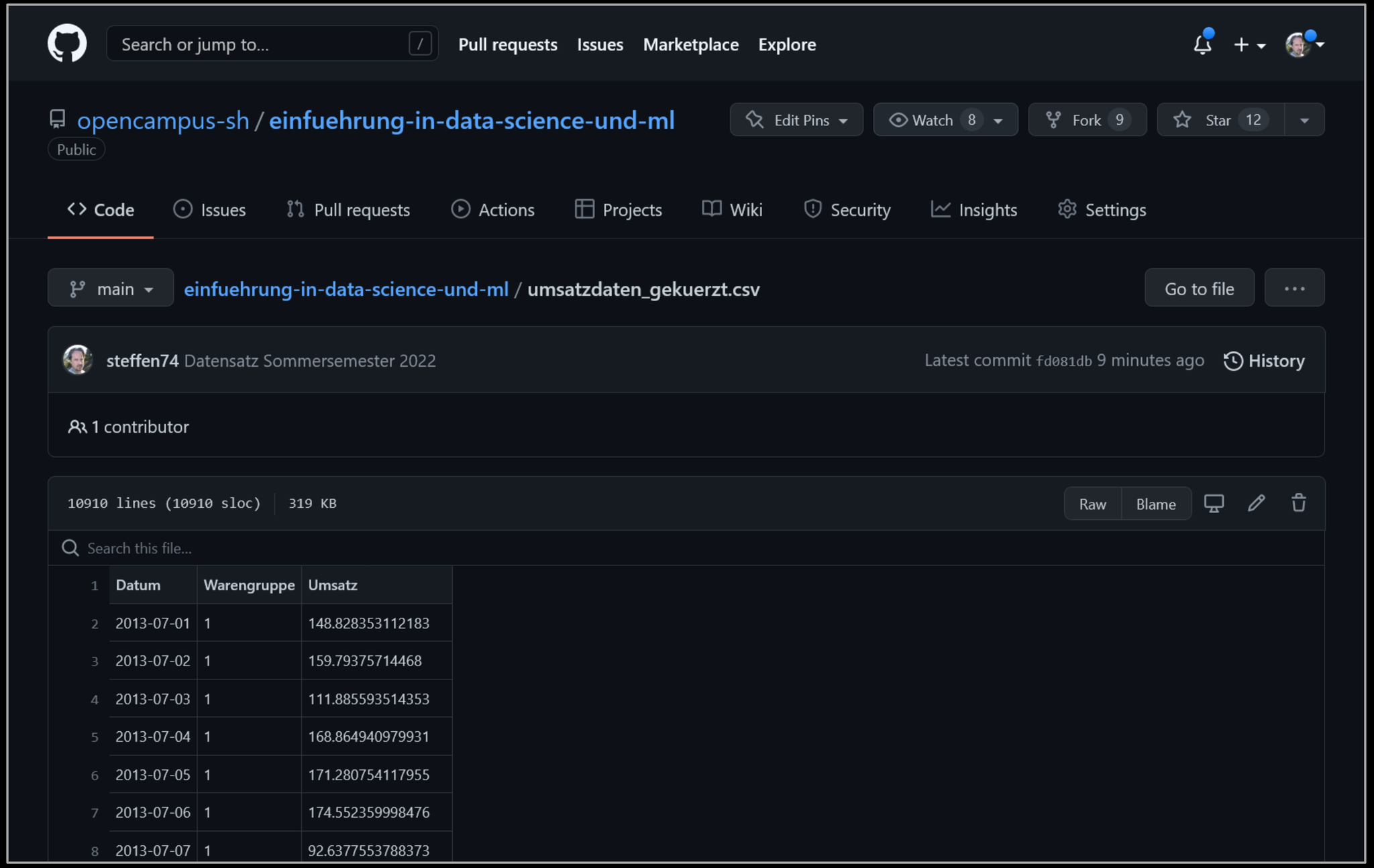
2) Ausführen des in der Graph Gallery gegebenen Beispielcodes

3) Ersetzen der gegeben Beispieldaten durch eigene (nutze ein LLM)

4) Anpassen des Diagramms durch Anweisung an das LLM oder über Hilfe-Seiten

PROJEKTDATENSATZ

- Zur Verfügung gestellt von Meteolytix
- Umsatzdaten von verschiedenen Warengruppen einer Bäckereifiliale für den Zeitraum vom 01.07.2013 bis zum 08.06.2019
- Wetterdaten für den Zeitraum vom 01.07.2013 bis zum 30.07.2019
- Abrufbar unter: https://raw.githubusercontent.com/opencampus-sh/einfuehrung-in-data-science-und-ml/main/umsatzdaten_gekuerzt.csv



	Datum	Warengruppe	Umsatz
2	2013-07-01	1	148.828353112183
3	2013-07-02	1	159.79375714468
4	2013-07-03	1	111.885593514353
5	2013-07-04	1	168.864940979931
6	2013-07-05	1	171.280754117955
7	2013-07-06	1	174.552359998476
8	2013-07-07	1	92.6377553788373

WARENGRUPPEN

1 Brot

2 Brötchen

3 Croissant

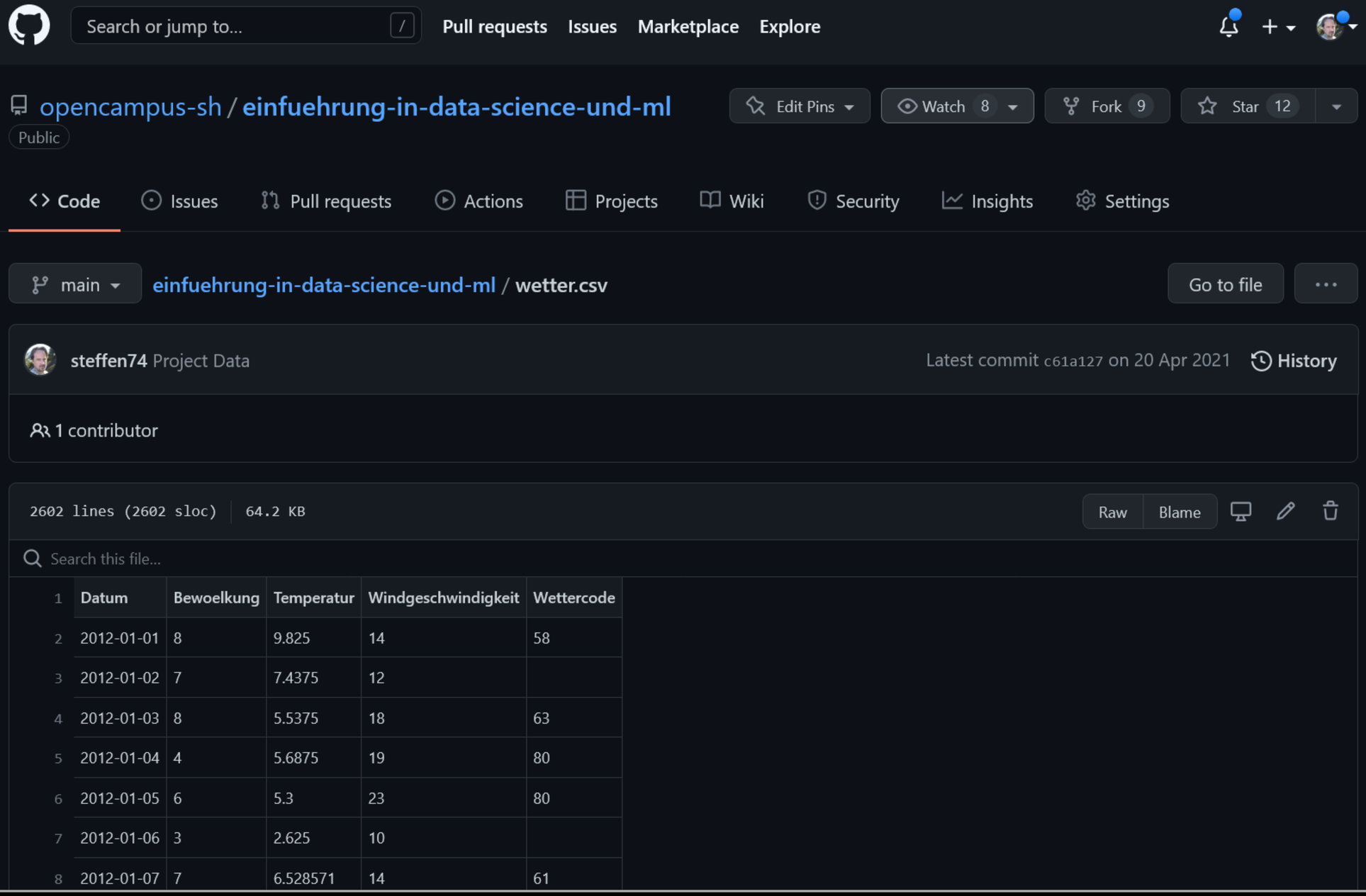
4 Konditorei

5 Kuchen

6 Saisonbrot

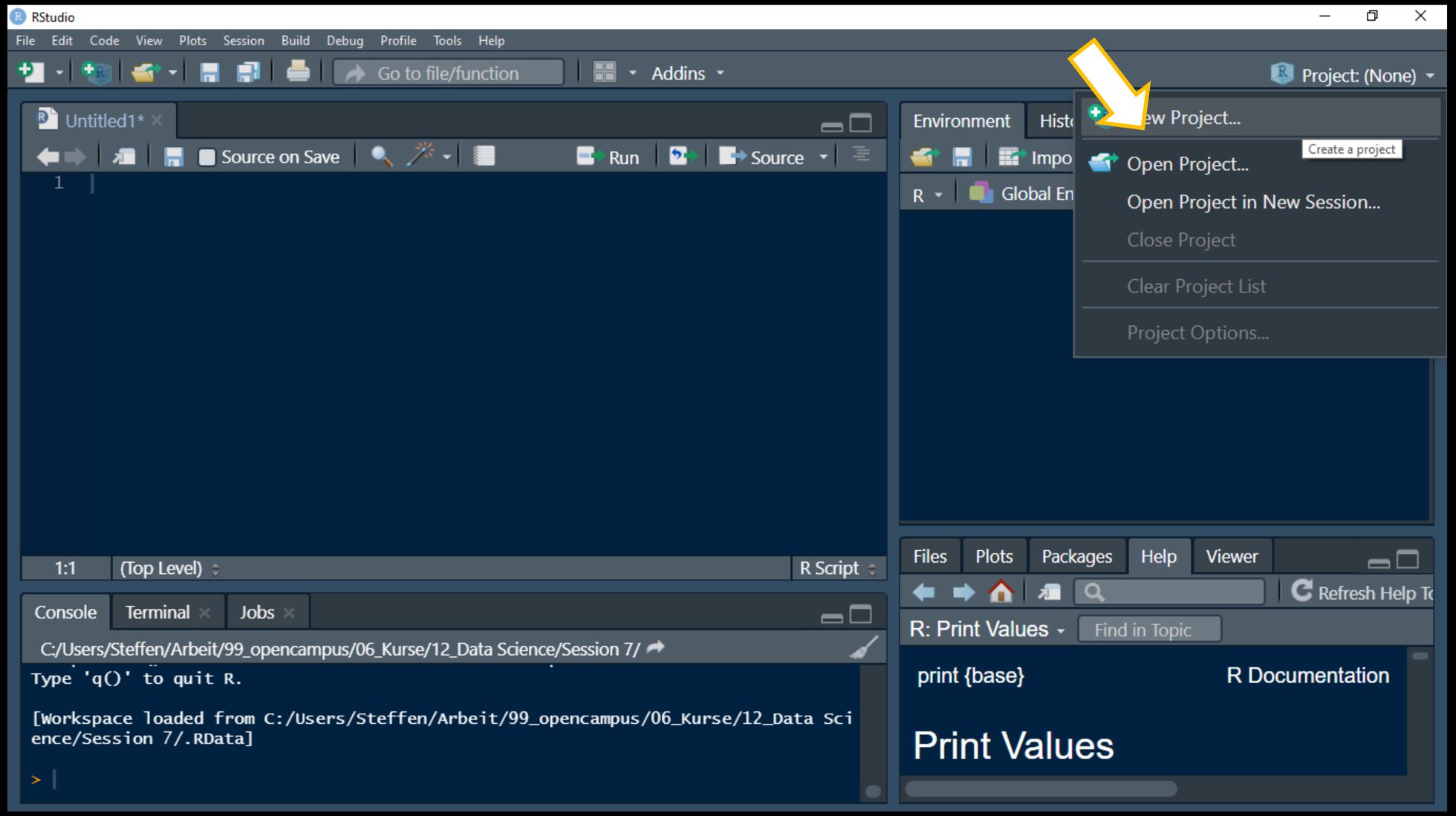
WETTERDATEN

- für den Zeitraum vom 01.07.2013 bis zum 30.07.2019
- Abrufbar unter: https://raw.githubusercontent.com/opencampus-sh/einfuehrung-in-data-science-und-ml/main/wetter.csv
- Variablen:
  - mittlerer Bewölkungsgrad am Tag (0: min bis 8: max)
  - mittlere Temperatur in Celsius
  - mittlere Windgeschwindigkeit in m/s
  - Wettercode (eine Liste mit Beschreibungen gibt es z.B. hier: http://www.seewetter-kiel.de/seewetter/daten_symbole.htm)



	Datum	Bewoelkung	Temperatur	Windgeschwindigkeit	Wettercode
2	2012-01-01	8	9.825	14	58
3	2012-01-02	7	7.4375	12	
4	2012-01-03	8	5.5375	18	63
5	2012-01-04	4	5.6875	19	80
6	2012-01-05	6	5.3	23	80
7	2012-01-06	3	2.625	10	
8	2012-01-07	7	6.528571	14	61

RSTUDIO-PROJEKT



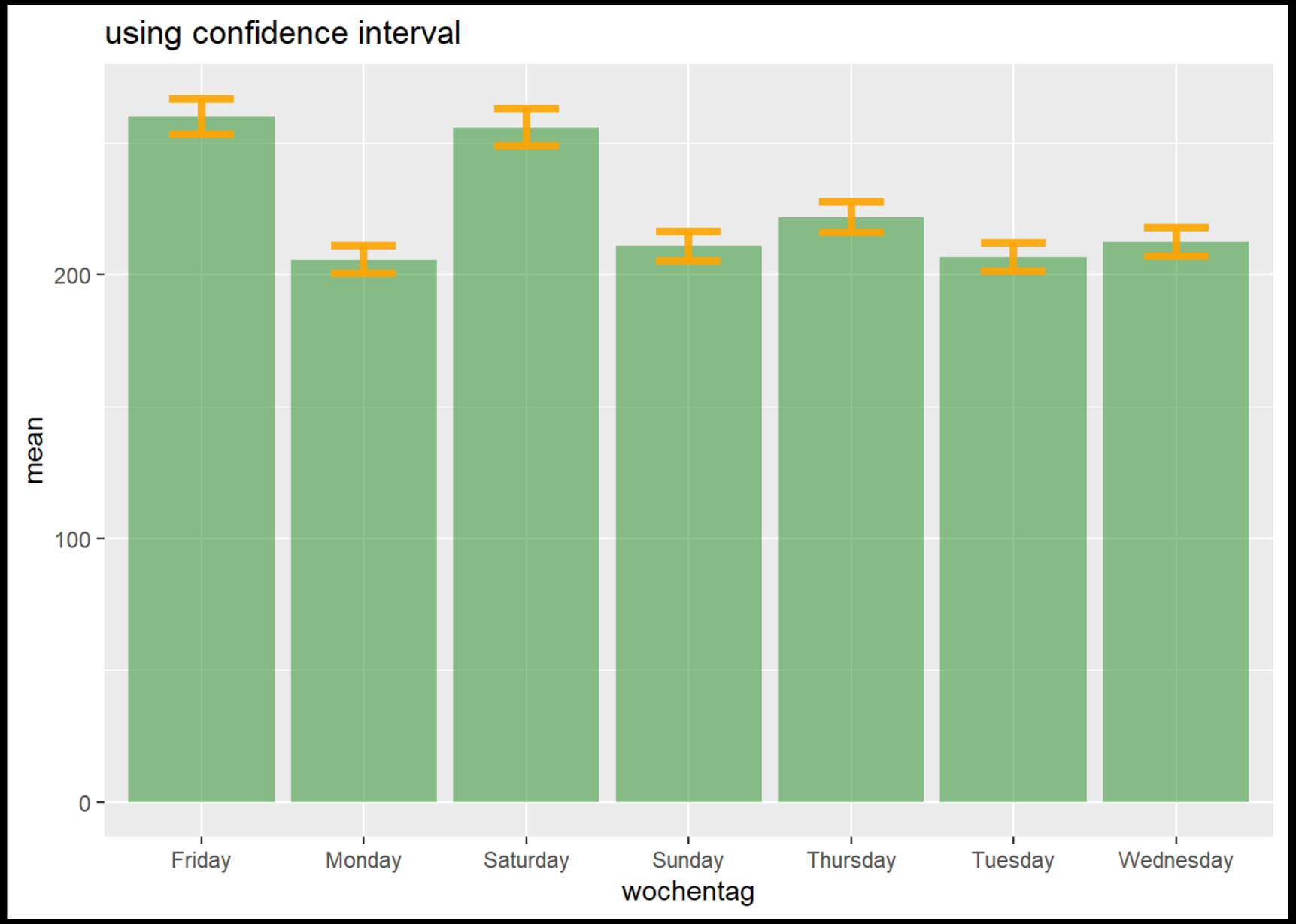
AUFGABEN

- Lege ein R-Studio-Projekt-Verzeichnis an und speichere dort die Dateien „kiwo.csv“, „umsatzdaten_gekuerzt.csv“ und „wetter.csv“ aus diesem GitHub-Repository: https://github.com/opencampus-sh/einfuehrung-in-data-science-und-ml

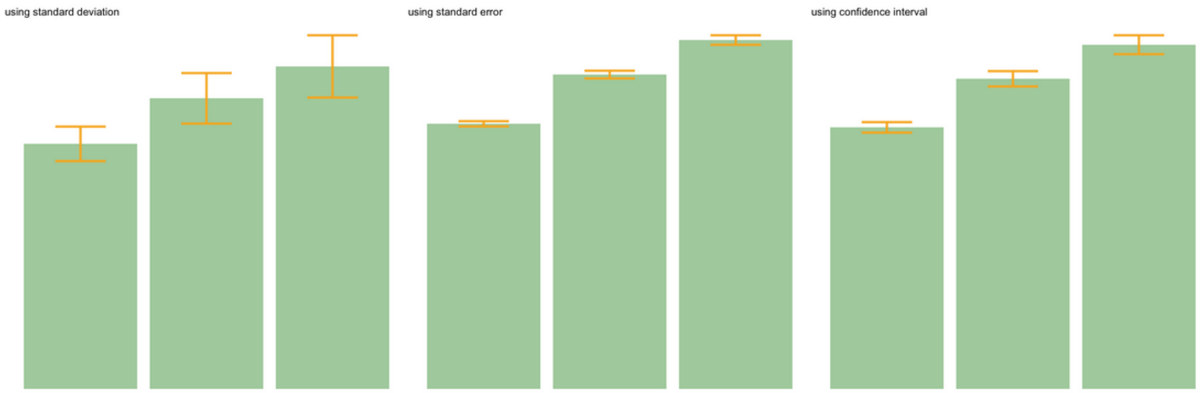
- Erstelle ein Balkendiagramm, dass über alle Warengruppen hinweg die durchschnittlichen Umsätze je Wochentag zeigt.

- Füge in einem zweiten Schritt zusätzlich Konfidenzintervalle der Umsätze je Wochentag hinzu („barplot with error bars“).

- Stelle die Umsätze je Wochentag getrennt nach Warengruppe dar (ein eigenes Balkendiagramm je Warengruppe)


using confidence interval
mean
200
100
0
Friday
Monday
Saturday
Sunday
Thursday
Tuesday
Wednesday
wochentag

Standard deviation, Standard error or Confidence Interval?



Three different types of values are commonly used for error bars, sometimes without even specifying which one is used. It is important to understand how they are calculated, since they give very different results (see above). Let's compute them on a simple vector:

```
vec=c(1,3,5,9,38,7,2,4,9,19,19)
```

→ Standard Deviation (SD). wiki

It represents the amount of dispersion of the variable. Calculated as the root square of the variance:

```
sd <- sd(vec)
sd <- sqrt(var(vec))
```

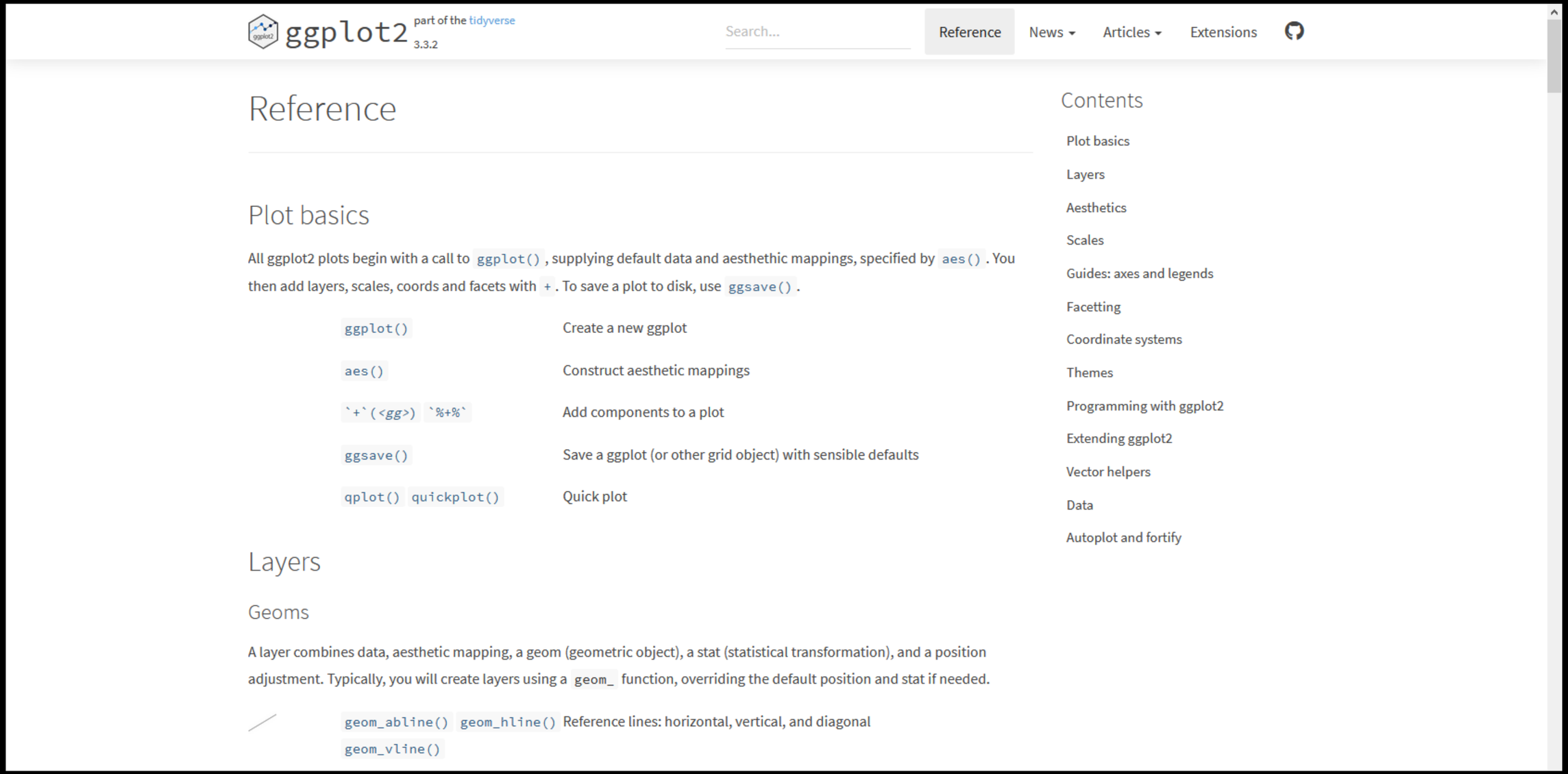
→ Standard Error (SE). wiki

It is the standard deviation of the vector sampling distribution. Calculated as the SD divided by the square root of the sample size. By construction, SE is

https://www.r-graph-gallery.com/4-barplot-with-error-bar.html

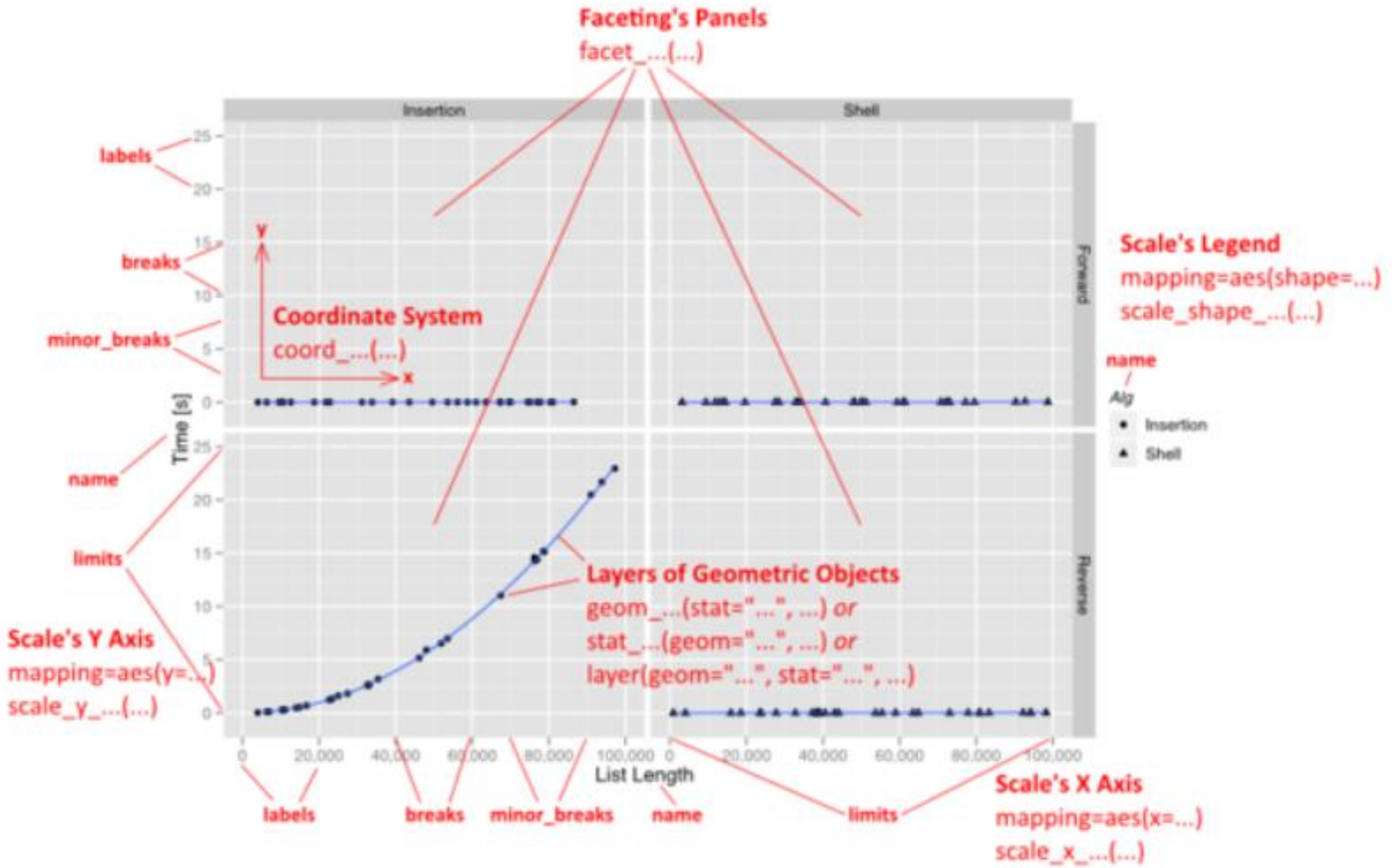
STARTHILFE

```
# Import needed libraries
library(readr)
library(lubridate)
library(ggplot2)
library(dplyr)

# Import turnover data
umsatzdaten <- read_csv("https://raw.githubusercontent.com/opencampus-sh/einfuehrung-in-data-science-und-ml/main/umsatzdaten_gekuerzt.csv")

# Create variable weekday
umsatzdaten$wochentag <- weekdays(umsatzdaten$Datum)
```



Übersicht über existierende Layer und den Funktionen, die existieren.

https://ggplot2.tidyverse.org/reference/



Gute bildliche Darstellung der Elemente einer Abbildung:
http://sape.inf.usi.ch/quick-reference/ggplot2

Articles - R Graphics Essentials

GGPlot Cheat Sheet for Great Customization

kassambara | 17/11/2017 | 36049 | Comments (2) | R Graphics Essentials

This chapter provides a cheat sheet to change the global appearance of a ggplot.

You will learn how to:

- Add title, subtitle, caption and change axis labels
- Change the appearance - color, size and face - of titles
- Set the axis limits
- Set a logarithmic axis scale
- Rotate axis text labels
- Change the legend title and position, as well, as the color and the size
- Change a ggplot theme and modify the background color
- Add a background image to a ggplot
- Use different color palettes: custom color palettess, color-blind friendly palettes, RColorBrewer palettes, viridis color palettes and scientific journal color palettes.
- Change point shapes (plotting symbols) and line types
- Rotate a ggplot
- Annotate a ggplot by adding straight lines, arrows, rectangles and text.

Contents:

How-To‘s mit verschiedenen Beispielen

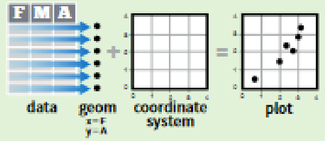
www.sthda.com/english/articles/32-r-graphics-essentials/125-ggplot-cheat-sheet-for-great-customization/

Data Visualization with ggplot2 Cheat Sheet

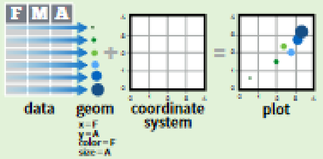
RStudio

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same few components: a **data** set, a set of **geoms**—visual marks that represent data points, and a **coordinate system.**



To display data values, map variables in the data set to aesthetic properties of the geom like **size**, **color**, and **x** and **y** locations.



Build a graph with **qplot()** or **ggplot()**

aesthetic mappings · data · geom

qplot(x = cty, y = hwy, color = cyl, data = mpg, geom = "point")

Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

a <- ggplot(mpg, aes(hwy))

a + **geom_area(stat = "bin")**
x, y, alpha, color, fill, linetype, size
b + geom_area(aes(y = ..density..), stat = "bin")

a + **geom_density(**kernel = "gaussian"**)**
x, y, alpha, color, fill, linetype, size, weight
b + geom_density(aes(y = ..county..))

a + **geom_dotplot()**
x, y, alpha, color, fill

a + **geom_freqpoly()**
x, y, alpha, color, linetype, size
b + geom_freqpoly(aes(y = ..density..))

a + **geom_histogram(**binwidth = 5**)**
x, y, alpha, color, fill, linetype, size, weight
b + geom_histogram(aes(y = ..density..))

Discrete

b <- ggplot(mpg, aes(fl))

b + **geom_bar()**
x, alpha, color, fill, linetype, size, weight

Graphical Primitives

c <- ggplot(map, aes(long, lat))

c + **geom_polygon(**aes(group = group)**)**
x, y, alpha, color, fill, linetype, size

d <- ggplot(economics, aes(date, unemploy))

Two Variables

Continuous X, Continuous Y

f <- ggplot(mpg, aes(cty, hwy))

f + **geom_blank()**

f + **geom_jitter()**
x, y, alpha, color, fill, shape, size

f + **geom_point()**
x, y, alpha, color, fill, shape, size

f + **geom_quantile()**
x, y, alpha, color, linetype, size, weight

f + **geom_rug(**sides = "bl"**)**
alpha, color, linetype, size

f + **geom_smooth(**model = lm**)**
x, y, alpha, color, fill, linetype, size, weight

f + **geom_text(**aes(label = cty)**)**
x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

Discrete X, Continuous Y

g <- ggplot(mpg, aes(class, hwy))

g + **geom_bar(stat = "identity")**
x, y, alpha, color, fill, linetype, size, weight

g + **geom_boxplot()**
lower, middle, upper, x, ymax, ymin, alpha, color, fill, linetype, shape, size, weight

g + **geom_dotplot(**binaxis = "y",

Continuous Bivariate Distribution

i <- ggplot(movies, aes(year, rating))

i + **geom_bin2d(**binwidth = c(5, 0.5)**)**
xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size, weight

i + **geom_density2d()**
x, y, alpha, colour, linetype, size

i + **geom_hex()**
x, y, alpha, colour, fill size

Continuous Function

j <- ggplot(economics, aes(date, unemploy))

j + **geom_area()**
x, y, alpha, color, fill, linetype, size

j + **geom_line()**
x, y, alpha, color, linetype, size

j + **geom_step(**direction = "hv"**)**
x, y, alpha, color, linetype, size

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)

k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

k + **geom_crossbar(**fatten = 2**)**
x, y, ymax, ymin, alpha, color, fill, linetype, size

k + **geom_errorbar()**
x, ymax, ymin, alpha, color, linetype, size, width (also **geom_errorbarh()**)

k + **geom_linerange()**
x, ymin, ymax, alpha, color, linetype, size

Cheat-Sheet von RStudio

https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf