

33 Minuten für...



„Coden mit KI“

Dr. Denis Arnold

Universität zu Köln

Universitäts- und Stadtbibliothek Köln

C³RDM

Der Markt ...

- GitHub Copilot
- GitLab Duo
- AWS CodeWhisperer
- Microsoft IntelliCode
- CodePal.ai
- ...
- ChatGPT
- Google Bard
- ...

Zwei Angebote mit dem leichtesten Zugang

- Codepal.ai
 - Erstes ausprobieren auf der Webseite ohne Anmelden möglich
 - Viele Sprachen
 - Viele Modi
 - Verschiedene Lizenzmodelle von \$ 0 bis auf Anfrage
- Google Bard
 - Anmeldung mit Google-Konto
 - Freie Eingabe auf Deutsch
 - Experimenteller Status: Keine Gebühren

{ Code Generator }

Generate Code Online with AI ?

Write a function in **JavaScript** that performs a spell check

Create
Standard

Your generated code will be shown here.

+ Neuer Chat



Hallo, ich bin Bard

Sag mir, womit ich dir helfen kann, oder wähle unten einen der Vorschläge aus.

Verstehen

Entwerfen

Entdecken

Produktvergleich

Hotelreservierung änd...

Berufsfelder



Prompt hier eingeben



● Köln, Deutschland
[Anhand deiner IP-Adresse](#) • [Standort aktualisieren](#)

Bard zeigt möglicherweise fehlerhafte Informationen an, z. B. zu Personen. Prüfe daher die Antworten noch einmal. [Datenschutz und Bard](#)

Szenario

Eine Student:in hat Daten in einer csv Datei und will in R einen t-Test machen und die Verteilungen mittels Boxplot visualisieren.

Szenario

Eine Student:in hat Daten in einer csv Datei und will in R einen t-Test machen und die Verteilungen mittels Boxplot visualisieren.

Herausforderungen:

- Wie sind die Daten in der Datei organisiert?
- Sind die Voraussetzungen für einen t-Test gegeben?
- Welche Art von t-Test soll gerechnet werden?



FREE Try our new tools: [Extension Creator](#), [Live Webpage Generator](#) and [Color Palette Generator](#).

{ Code Generator }

Generate Code Online with AI

Write a function in that

```
1 # This function reads data from a CSV file, performs a t-test, and creates a boxplot.
2 #
3 # PARAMETERS:
4 #   - file_path: The path to the CSV file.
5 #     TYPE: character
```

Skript codepal 1

Prompt: Write a function in “R” that “reads data from a csv file, performs a t-test and makes a boxplot”

```
# This function reads data from a CSV file, performs a t-test, and creates a boxplot.
#
# PARAMETERS:
# - file_path: The path to the CSV file.
#   TYPE: character
#   EXAMPLE: "data.csv"
#
# RETURNS:
# - None. The function generates a boxplot and prints the results of the t-test.
#
# EXCEPTIONS:
# - This function assumes that the CSV file has a header row and contains numeric data.
# - If the file does not exist or cannot be read, an error message will be displayed.
```

Skript codepal 2

Prompt: Write a function in “R” that “reads data from a csv file, performs a t-test and makes a boxplot”

```
perform_t_test_and_boxplot <- function(file_path) {  
  # Read the data from the CSV file  
  data <- read.csv(file_path)  
  
  # Perform a t-test  
  t_test_result <- t.test(data$Group1, data$Group2)  
  
  # Create a boxplot  
  boxplot(data$Group1, data$Group2, main = "Boxplot", xlab = "Group", ylab = "Value")  
  
  # Print the results of the t-test  
  cat("T-Test Results:\n")  
  cat("=====\n")  
  cat("p-value:", t_test_result$p.value, "\n")  
  cat("t-value:", t_test_result$statistic, "\n")  
  cat("degrees of freedom:", t_test_result$parameter, "\n")  
}
```

Skript codepal 3

Prompt: Write a function in “R” that “reads data from a csv file, performs a t-test and makes a boxplot”

```
# MAIN SCRIPT EXECUTION

# Example usage
file_path <- "data.csv" # Example file path
perform_t_test_and_boxplot(file_path)

# Usage example

# USAGE EXAMPLE:
# Using a different file
file_path <- "data2.csv"
perform_t_test_and_boxplot(file_path)
```

+ Neuer Chat

Letzte Themen

- R-Skript für CSV-Datei, t...
- Dublin Core Metadaten in SVG ...

● Köln, Deutschland
[Anhand deiner IP-Adresse](#) • [Standort aktualisieren](#)

D Ich hätte gerne ein R Skript, das mir eine cvs Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

Weitere Vorschläge ansehen

Hier ist ein R-Skript, das diese Anforderungen erfüllt:

```
R

library(ggplot2)
library(stats)

# CSV-Datei einlesen
data <- read.csv("data.csv")
```

Prompt hier eingeben

Bard zeigt möglicherweise fehlerhafte Informationen an, z. B. zu Personen. Prüfe daher die Antworten noch einmal. [Datenschutz und Bard](#)

Skript Bard 1

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

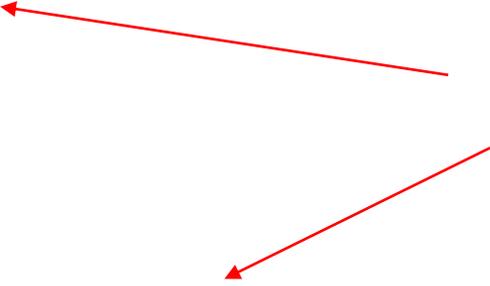
```
• library(ggplot2)
•
• # CSV-Datei einlesen
• data <- read.csv("data.csv")
•
• # Variablennamen festlegen
• x <- data$x
• y <- data$y
•
• # T-Test durchführen
• t.test(x, y)
•
• # Boxplot erstellen
• ggplot(data, aes(x=Gruppe, y=Wert)) +
•   geom_boxplot() +
•   labs(x="Gruppe", y="Wert")
```

Skript Bard 1

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

```
• library(ggplot2)
•
• # CSV-Datei einlesen
• data <- read.csv("data.csv")
•
• # Variablennamen festlegen
• x <- data$x
• y <- data$y
•
• # T-Test durchführen
• t.test(x, y)
•
• # Boxplot erstellen
• ggplot(data, aes(x=Gruppe, y=Wert)) +
•   geom_boxplot() +
•   labs(x="Gruppe", y="Wert")
```

Unterschiedliche Anforderungen an data !!!



Skript Bard 2

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

Dieses Skript funktioniert wie folgt:

- Die erste Zeile lädt die `ggplot2`-Bibliothek. Diese Bibliothek wird benötigt, um Boxplots zu erstellen.
- Die zweite Zeile liest die CSV-Datei mit dem Namen `data.csv` ein.
- Die dritte Zeile legt die Variablennamen `x` und `y` fest. `x` ist die Variable, für die der t-Test durchgeführt werden soll, und `y` ist die Vergleichsvariable.
- Die vierte Zeile führt den t-Test durch. Die Ergebnisse werden in der Konsole ausgegeben.
- Die fünfte Zeile erstellt den Boxplot. Die Variable `Gruppe` wird verwendet, um die Gruppen zu unterscheiden.
- Die sechste Zeile legt die Beschriftungen für die x- und y-Achse fest.

Skript Bard 3

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

Wenn Sie das Skript ausführen, werden die folgenden Ergebnisse angezeigt:

```
## Welch Two Sample t-test
##
## data: x and y
## t = -2.13, df = 19.65, p-value = 0.043
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -1.617842 -0.242158
## sample estimates:
## mean of x mean of y
## 0.5 1.0
```

Skript Bard 4

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

Der Boxplot zeigt, dass die beiden Gruppen in Bezug auf die Variable x unterschiedlich sind. Die Gruppe $x=1$ hat im Durchschnitt einen niedrigeren Wert als die Gruppe $x=2$.

Sie können das Skript an Ihre Bedürfnisse anpassen, indem Sie die folgenden Änderungen vornehmen:

- Sie können die Namen der Variablen ändern.
- Sie können die Gruppennamen ändern.
- Sie können die Beschriftungen für die x - und y -Achse ändern.

Skript Bard 5

Prompt : Ich hätte gerne ein R Skript, das mir eine csv Datei einliest, einen t-Test macht und die Daten in einem Boxplot visualisiert

- Sie können die Art des t-Tests ändern. Zum Beispiel können Sie einen einseitigen t-Test durchführen, indem Sie die Option `alternative="greater"` oder `alternative="less"` verwenden.

Hier ist ein Beispiel für einen einseitigen t-Test:

R

```
t.test(x, y, alternative="greater")
```

Verwende Code mit Vorsicht. [Weitere Informationen](#)

Dieser Test prüft die Hypothese, dass der Mittelwert von x größer oder gleich dem Mittelwert von y ist.

Erkenntnisse Szenario 1:

- Die Assistenten schlagen z.T. Code vor, der auf ein denkbares Szenario passen könnte
 - Sie machen dabei starke Annahmen über die vagen Angaben zu den Daten
 - Sie geben hier (noch?) keine Expliziten Hinweise, dass und wie sich die Struktur der Daten vom Beispiel unterscheiden können
 - Unterschiedliche Anforderungen an Daten im selben Skript !!!
 - Sie gehen nicht auf die Voraussetzungen des verwendeten statistischen Tests ein und testen die Daten auch nicht hierauf
- Assistenten sind erstmal keine Methodenberatung, wenn man nicht danach fragt

Methodenfrage Bard

Wie kann ich prüfen, ob die Daten in der csv Datei einen t-test zulassen und was muss ich sonst noch beachten?

Um zu prüfen, ob die Daten in einer CSV-Datei einen t-Test zulassen, müssen Sie die folgenden Voraussetzungen überprüfen:

Die Daten müssen normalverteilt sein. Dies können Sie mit einem Shapiro-Wilk-Test prüfen.

Die Datenvarianzen müssen homogen sein. Dies können Sie mit einem Levene-Test prüfen.

Wenn die Daten nicht normalverteilt sind, können Sie einen nichtparametrischen t-Test durchführen, z. B. den Wilcoxon-Signed-Rank-Test. Wenn die Datenvarianzen nicht homogen sind, können Sie einen Welch-t-Test durchführen.

Hier sind einige weitere Dinge, die Sie beim Vergleich von Daten mit einem t-Test beachten sollten:

Die Daten müssen unabhängig voneinander sein. Dies bedeutet, dass die Beobachtungen in jeder Gruppe nicht voneinander beeinflusst werden dürfen.

Methodenfrage Bard 2

Wie kann ich prüfen, ob die Daten in der csv Datei einen t-test zulassen und was muss ich sonst noch beachten?

Die Daten müssen eine ausreichende Anzahl von Beobachtungen enthalten. Die Mindestanzahl von Beobachtungen hängt von der Art des t-Tests und dem gewünschten Signifikanzniveau ab.

Wenn Sie diese Voraussetzungen nicht überprüfen, können Sie falsche Ergebnisse erhalten.

Hier sind einige Tipps, wie Sie die Voraussetzungen für einen t-Test prüfen können:

Verwenden Sie eine Statistiksoftware, um die Tests durchzuführen.

Überprüfen Sie die Ergebnisse der Tests visuell. Wenn die Daten nicht normalverteilt sind, können Sie z. B. ein Histogramm oder ein Quantil-Quantil-Diagramm erstellen. Wenn die Datenvarianzen nicht homogen sind, können Sie z. B. Boxplots für jede Gruppe erstellen.

Wenn Sie die Voraussetzungen für einen t-Test nicht erfüllen, können Sie alternative Tests durchführen.

Szenario

Wir wollen gerne Dublin Core Metadaten in eine svg Datei schreiben

Szenario

Wir wollen gerne Dublin Core Metadaten in eine svg Datei schreiben

Herausforderungen:

Was ist Dublin Core?

Wie funktioniert eine svg Datei?

Szenario

Wir wollen gerne Dublin Core Metadaten in eine svg Datei schreiben

Herausforderungen:

Was ist Dublin Core?

Wie funktioniert eine svg Datei?



```
<svg viewBox="0 0 100 100">  
  <rect width="100" height="100" fill="#005176" />  
</svg>
```

Skript codepal 1

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file”

```
def add_dublin_core_metadata(svg_file_path: str, title: str, creator: str, description: str):  
    """  
    Function to add Dublin Core metadata into an SVG file.  
  
    Parameters:  
    - svg_file_path: str  
        The file path of the SVG file to which the metadata will be added.  
    - title: str  
        The title of the SVG file.  
    - creator: str  
        The creator/author of the SVG file.  
    - description: str  
        The description of the SVG file.  
  
    Raises:  
    - FileNotFoundError:  
        If the SVG file does not exist at the specified file path.  
    """
```

Skript codepal 2

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file”

```
# Opening the SVG file in read mode
with open(svg_file_path, 'r') as file:
    svg_content = file.read()

# Creating the Dublin Core metadata string
metadata = f'<metadata>\n<title>{title}</title>\n<creator>{creator}</creator>\n<description>{description}</description>\n</metadata>'

# Adding the metadata to the SVG content
updated_svg_content = svg_content.replace('</svg>', f'{metadata}\n</svg>')

# Writing the updated SVG content back to the file
with open(svg_file_path, 'w') as file:
    file.write(updated_svg_content)
```

Skript codepal 1

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file using a xml library”

```
import xml.etree.ElementTree as ET

def add_dublin_core_metadata(svg_file_path: str, title: str, creator: str, description: str):
    """
    Function to add Dublin Core metadata into an SVG file using an XML library.
    Parameters:
    - svg_file_path: str
        The file path of the SVG file where the metadata will be added.
    - title: str
        The title of the SVG file.
    - creator: str
        The creator/author of the SVG file.
    - description: str
        The description of the SVG file.
    Raises:
    - FileNotFoundError:
        If the SVG file specified by 'svg_file_path' does not exist.
    """
```

Skript codepal 2

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file using a xml library”

```
try:
    # Parse the SVG file
    tree = ET.parse(svg_file_path)
    root = tree.getroot()

    # Create the Dublin Core metadata elements
    dc_metadata = ET.Element("{http://purl.org/dc/elements/1.1/}metadata")
    dc_title = ET.SubElement(dc_metadata, "{http://purl.org/dc/elements/1.1/}title")
    dc_creator = ET.SubElement(dc_metadata, "{http://purl.org/dc/elements/1.1/}creator")
    dc_description = ET.SubElement(dc_metadata, "{http://purl.org/dc/elements/1.1/}description")

    # Set the text values for the Dublin Core metadata elements
    dc_title.text = title
    dc_creator.text = creator
    dc_description.text = description
```

Skript codepal 3

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file using a xml library”

```
# Append the Dublin Core metadata to the root element
    root.append(dc_metadata)

# Save the modified SVG file
    tree.write(svg_file_path)

    print("Dublin Core metadata added successfully.")
except FileNotFoundError:
    raise FileNotFoundError(f"SVG file '{svg_file_path}' not found.")
```

Skript codepal 4

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file using a xml library”

```
# Example usage of the add_dublin_core_metadata function:  
  
# Add Dublin Core metadata to an SVG file  
svg_file_path = "path/to/svg/file.svg"  
title = "My SVG File"  
creator = "John Doe"  
description = "An example SVG file"  
add_dublin_core_metadata(svg_file_path, title, creator, description)
```

Skript codepal 4

Prompt: Write a function in “Python” that “adds dublin core meta data to a svg file using a xml library”

```
# Example usage of the add_dublin_core_metadata function:
```

```
# Add Dublin Core metadata to an SVG file
```

```
svg_file_path = "path/to/svg/file.svg"
```

```
title = "My SVG File"
```

```
creator = "John Doe"
```

```
description = "An example SVG file"
```

```
add_dublin_core_metadata(svg_file_path, title, creator, description)
```



```
<svg xmlns:dc="http://purl.org/dc/elements/1.1/" viewBox="0 0 100 100">  
  <rect width="100" height="100" fill="#005176" />  
  <dc:metadata><dc:title>My SVG File</dc:title><dc:creator>John  
Doe</dc:creator><dc:description>An example SVG  
file</dc:description></dc:metadata></svg>
```

Was dürfen wir mit dem Code machen?

- Google Bard FAQ:
 - Kann Bard beim Programmieren helfen?
 - Ja, Bard kann beim Programmieren und bei Themen rund um das Programmieren helfen. Da Bard sich aber noch in der Testphase befindet, bist du für die Verwendung von Code oder Coding-Erklärungen verantwortlich. Teste den Code sorgfältig und prüfe ihn auf Fehler und Sicherheitslücken, bevor du ihn verwendest. Falls Code einer Open-Source-Lizenz unterliegt, gibt es in Bard einen entsprechenden Hinweis.

Was dürfen wir mit dem Code machen?

- Class action lawsuit gegen GitHub, Microsoft und OpenAI vor dem US District Court for the Northern District of California
 - Antrag auf Zurückweisung abgelehnt
 - DMCA violation
 - breach of OSS licenses
 - unjust enrichment
 - unfair competition