

Blatt 7

Vincent Kümmerle und Elvis Gnaglo

5. Dezember 2025

1 Zotero

Machinelles Lernen wird verwendet, um Muster in Datensammlungen herauszufinden. [1]

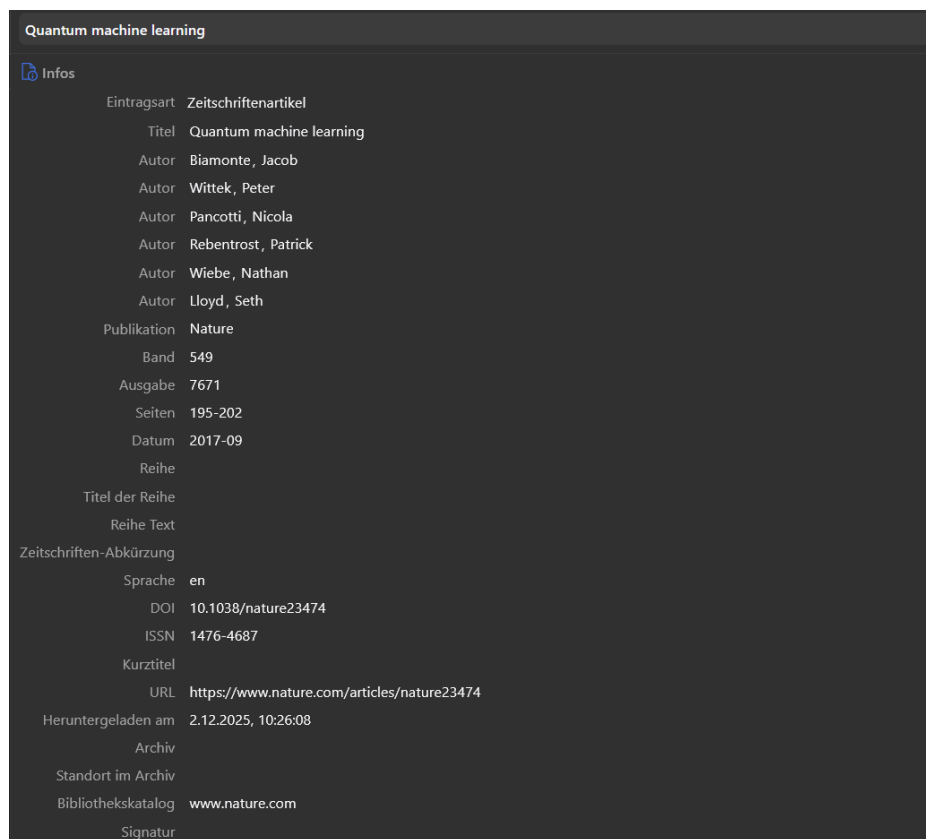


Abbildung 1: Zeitschriftenartikel über Quantum Machine Learning in der Zotero-Bibliothek.

Das Buch "Experimentalphysik 1" von Wolfgang Demtröder erklärt die Grundlagen der Mechanik und Wärmelehre. [2]

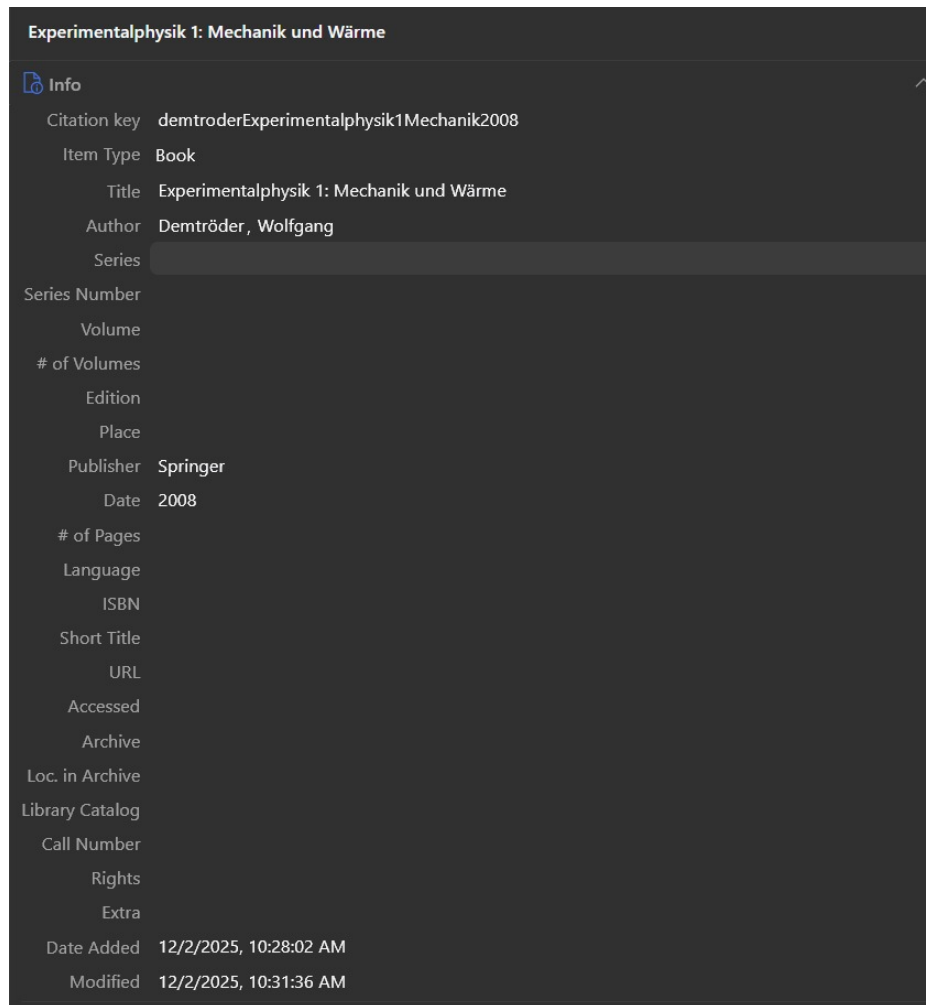


Abbildung 2: Buch über die Mechanik und Wärmelehre in der Physik.

2 List Comprehension

```
1 import random
2 import matplotlib.pyplot as plt
3 import math
4
5 def generate_random_point_in_square(length):
6     """
7     Generate a random point in a square of a given side length
8     centered around the origin with the sides aligned with x- and
9     y-axis.
10    """
11    return tuple([length*random.uniform(-0.5, 0.5), length*random.
12                  uniform(-0.5, 0.5)])
13
14 def distance(p1, p2):
```

```

12     """
13     Measures the Euclidean distance between the points p1 and p2 in 2
        D.
14     """
15     # TODO: wurzel((x2-x1)^2 + (y2-y1)^2)
16     return math.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)
17
18 def point_in_circle(p, center=(0,0), radius=1.0):
19     """
20     Checks if the provided point p lies in a circle of a given radius
        around the provided center.
21     """
22     # TODO: Abstand zum Mittelpunkt kleiner/gleich dem Radius
23     return distance(p, center) <= radius
24
25
26 # TODO: generate a list of 10000 random points in a square of side
        length 2.0 using generate_random_point_in_square and list
        comprehension
27 # Liste erzeugen mit _ als Laufvariable, kein Index nötig
28 full_list_of_points = [generate_random_point_in_square(2.0) for _ in
        range(10000)]
29
30 # TODO: generate a list of those points in full_list_of_points that
        are within a radius of 1.0 of the origin (again using list
        comprehension)
31 # Liste filtern basierend auf der Bedingung point_in_circle
32 filtered_list_of_points = [p for p in full_list_of_points if
        point_in_circle(p, center=(0,0), radius=1.0)]
33
34 # TODO: print the ratio of the number of points in
        filtered_list_of_points to the total number of generated points
35 ratio = len(filtered_list_of_points) / len(full_list_of_points)
36 print(f"Verhältnis (Punkte im Kreis / alle Punkte): {ratio}")
37 print(f"Annäherung an Pi (Verhältnis * 4): {ratio * 4}")
38 # Verhältnis Punkte im Kreis (in gefilterter Liste) / alle Punkte =
        Fläche Kreis / Fläche Quadrat = (pi*r^2) / (2r)^2 = pi/4
39 # Somit konvergiert es für n gegen Unendlich gegen pi/4.
40
41 # TODO: Generate separate lists of the x- and y-values of
        full_list_of_points and filtered_list_of_points
42 # x-Werte sind am Index 0, y-Werte am Index 1 jedes Tupels
43 x_values_full = [p[0] for p in full_list_of_points]
44 y_values_full = [p[1] for p in full_list_of_points]
45

```

```

46 x_values_filtered = [p[0] for p in filtered_list_of_points]
47 y_values_filtered = [p[1] for p in filtered_list_of_points]
48
49 # Plotten
50 plt.figure(figsize=(6, 6)) # Quadratische Darstellung für korrekte
    Proportionen
51 plt.scatter(x_values_full, y_values_full, s=1, label='außerhalb',
    color='blue')
52 plt.scatter(x_values_filtered, y_values_filtered, s=1, label='
    innerhalb', color='red')
53 plt.legend(loc='upper right')
54 plt.title(f'Monte Carlo Simulation (Verhältnis = {ratio})')
55 plt.axis('equal') # damit der Kreis nicht verzerrt aussieht
56 plt.show()

```

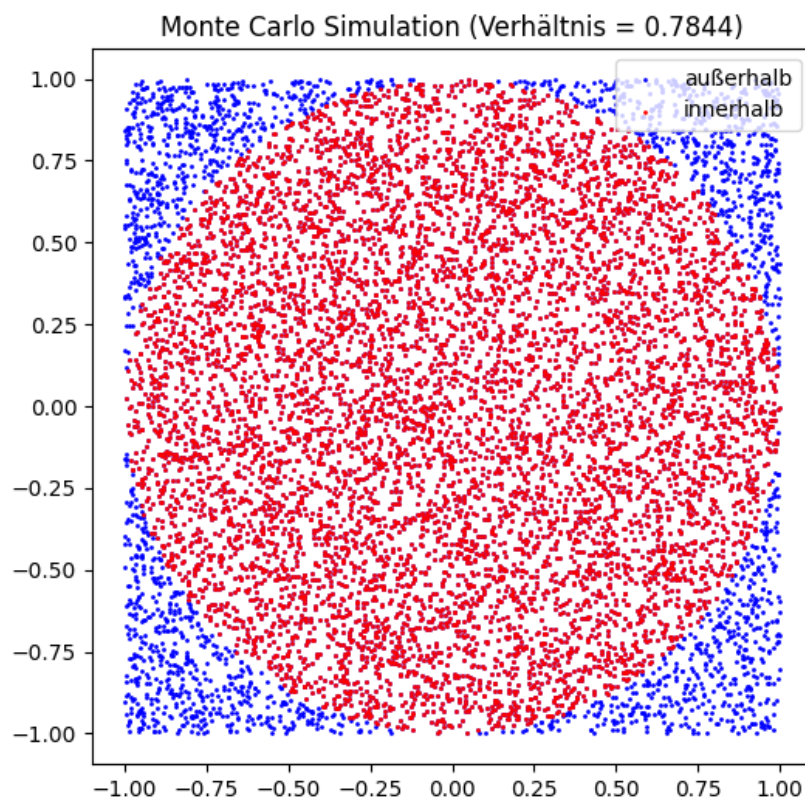


Abbildung 3: Monte Carlo Simulation

3 Dateien lesen und Daten plotten

```
1 import matplotlib.pyplot as plt
2 import os
3
4 dateiname = r"C:\Studium\5. Semester\AC II lab\Protokolle\CGL\Blatt
5       7\messdaten.txt"
6 x_werte = []
7 y_werte = []
8
9 # 1. Datei öffnen und Schleife über Zeilen
10 try:
11     with open(dateiname, 'r') as datei:
12         print(f"Verarbeite Daten aus '{dateiname}'...")
13
14         for zeilennummer, zeile in enumerate(datei, 1):
15
16             # 2. .strip() verwenden
17             bereinigte_zeile = zeile.strip()
18
19             # 3. Filtern von leeren Zeilen und Kommentaren
20             if not bereinigte_zeile or bereinigte_zeile.startswith('#
21                 '):
22                 # Ignoriere auch Zeilen, die mit '#' (Kommentar)
23                 # beginnen
24                 continue
25
26             # 4. Sonst Zeile ausgeben und Werte extrahieren
27             print(f"Verarbeite: '{bereinigte_zeile}'")
28             # Teile die bereinigte_zeile in Einzelteile auf
29             teile = bereinigte_zeile.split()
30
31             if len(teile) == 2:
32                 try:
33                     # Wandel die Teile in Gleitkommazahlen um
34                     x = float(teile[0])
35                     y = float(teile[1])
36
37                     # Speicher die Werte in den Listen x_werte und
38                     # y_werte
39                     x_werte.append(x)
40                     y_werte.append(y)
41                 except ValueError:
42                     print(f"Warnung in Zeile {zeilennummer}: Konnte
43                           Werte nicht in Zahlen umwandeln: '{
```

```

        bereinigte_zeile}''")
39     else:
40         print(f"Warnung in Zeile {zeilennummer}: Unerwartetes
        Format (erwarte 2 Werte): '{bereinigte_zeile}''")
41
42 except FileNotFoundError:
43     print(f"Fehler: Die Datei '{dateiname}' wurde nicht gefunden.")
44
45 # 5. Am Ende alle x-y Werte plotten
46 if x_werte:
47     print(f"Erfolgreich {len(x_werte)} Datenpunkte gesammelt.
        Erstelle Plot.")
48
49     plt.figure(figsize=(10, 6))
50     folder = r'C:\Studium\5. Semester\AC II lab\Protokolle\CGL\Blatt
        7'
51     # Verwenden Sie plt.scatter() um x_werte gegen y_werte zu plotten
52     plt.scatter(x_werte, y_werte, color='red', marker='x', label='
        Messpunkte')
53
54     plt.xlabel('X-Koordinate')
55     plt.ylabel('Y-Koordinate')
56
57     plt.title('Messdaten aus Datei')
58     plt.xlabel('X-Koordinate')
59     plt.ylabel('Y-Koordinate')
60     plt.grid(True)
61     plt.savefig(os.path.join(folder, 'messdaten_plot.pdf'))
62     plt.show()
63 else:
64     print("Keine gültigen Daten zum Plotten gefunden.")

```

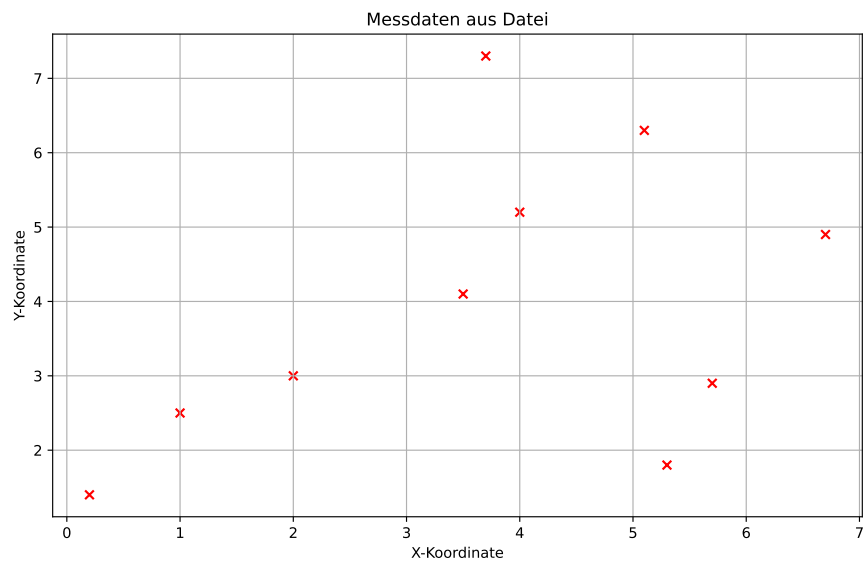


Abbildung 4: Plot der Messdaten

Literaturverzeichnis

- [1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, *Nature* **2017**, 549, 195–202.
- [2] W. Demtröder, *Experimentalphysik 1: Mechanik Und Wärme*, Springer, **2008**.