Computerpraktikum Maschinelles Lernen

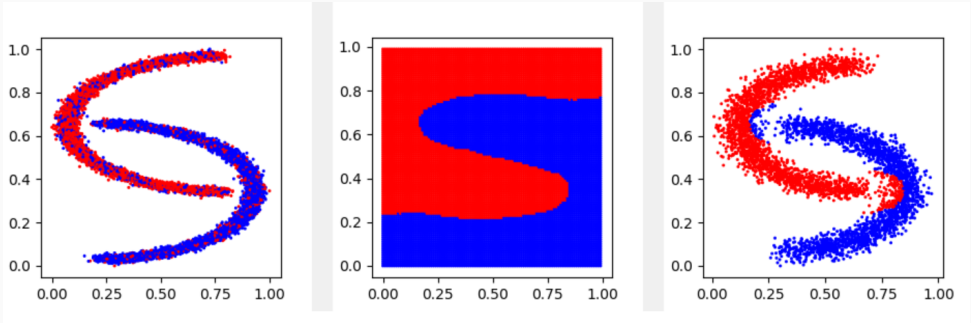
Thema 4 - Klassifikationsverfahren

Pascal Bauer, Raphael Millon, Florian Haas

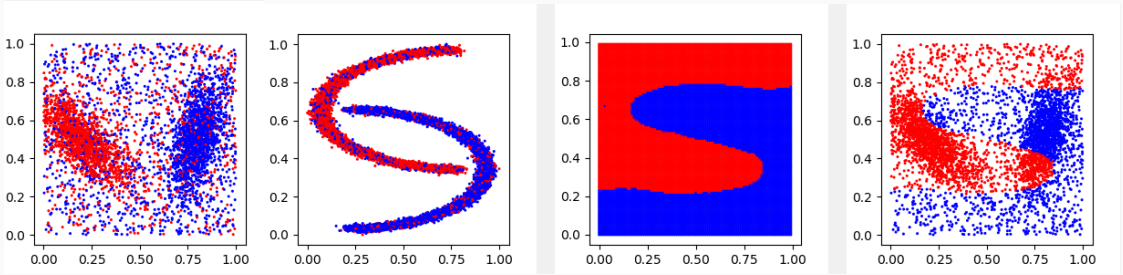
Sommersemester 2020

Table of contents

Frage: Was passiert, wenn wir als Testdaten andere Datensätze verwenden?



(Von links nach rechts: Trainingsdaten (bananas-1-2d), Gitter, Ergebnis (mit Testdaten bananas-2-2d))



(Von links nach rechts: Testdaten (toy-2d), Trainingsdaten (bananas-1-2d), Gitter, Ergebnis)

Codebeispiele Struktur und Module

Code ist Open-Source auf Github:
`https://github.com/raphaelMi/computerpraktikum-maschinelles-lernen`

Unser Programm ist in folgende Module aufgeteilt:

- **main.py**: Hauptmodul mit wesentlichen Algorithmen
- **dataset.py**: Datensatz-Import/-Export
- **gui.py**: Grafische Oberfläche
- **kd_tree.py**: Hilfsmodul für k-d-Search
- **visual.py**: Plotting der Datensätze

Verwendete Bibliotheken:

- **numpy**: Effizientes (vektorisiertes) Rechnen
- **matplotlib**: Generieren der Plots
- **tkinter**: Grafische Benutzeroberflächen
- **scikit-learn**: Ein dritter Algorithmus zum Vergleich

Codebeispiele Klassifikation

Die **classify**-Funktion ist das "Herz" unseres Programmes:

```
def classify_gui(train_data, test_data, output_path, kset=K, l=5, algorithm='brute_sort'):
    if algorithm == 'brute_sort':
        dd, k_best = train_brute_sort(train_data, kset, l)
        print('k* =', k_best)
        f_rate, result_data = test(dd, test_data, k_best, output_path)
        return k_best, f_rate, result_data, dd
```

Parameter:

- **train_data**: Trainingsdaten
- **train_data**: Testdaten
- **output_path**: Ausgabedatei der Ergebnisdaten
- **kset**: Menge der k
- **l**: Partitionsanzahl
- **algorithm**: Suchalgorithmus für Nachbarn

Ablauf:

1. Training mit gegebenen Trainingsdaten und Sortieralgorithmus
2. Klassifikation und der Testdaten und Darstellung der Resultate

Codebeispiele Training