Mathematische Grundlagen des maschinellen Lernens

Ziel der Versuche:

In diesem Praktikumsversuch sind 3 unabhängige Teilaufgaben zu lösen:

1. Polynome unterschiedlichen Grades (Gerade, kubisches Polynom, Polynom 9. Grades) sind in eine Wolke von Datenpunkten bestmöglich (Ausgleichsrechnung) einzupassen.

2. In einem Funktionsgebirge ist durch Gradientenabstieg ein lokales Minimum zu suchen und der Pfad anzuzeigen.

3. Ein einzelnes Neuron mit logistischer Aktivierungsfunktion ist zu trainieren und der trainierte Eingangsvektorraum ist darzustellen.

Zu jeder Teilaufgabe gibt es bereits ein Jupyter-Notebook mit vorbereitendem Code, welches zu vervollständigen ist.

Darüber hinaus gibt es zu jeder Teilaufgabe ein numpy-Tutorial (ebenfalls ein Jupyter-Notebook), in dem hilfreichen Codesnippets (z.B. Plotten von Funktionen, Lösen überbestimmter Gleichungssysteme, Vektor in Matrix umwandeln usw.) zu finden sind.

Schauen Sie sich zuerst die Tutorials zu den jeweiligen Teilaufgaben an und vollziehen Sie diese nach. Erst danach sollten Sie mit den Teilaufgaben beginnen.

Teilaufgabe 1: Polynomapproximation

In der Vorlesung wurde gezeigt, wie mit Hilfe von Polynomen

$$y = p_m x^m + p_{m-1} x^{m-1} + ... + p_2 x^2 + p_1 x + p_0$$

eine Approximationsfunktion gefunden werden kann. Hierzu wird für jeden Datenpunkt (Messwert)

$$(x_1, y_1),\ (x_2, y_2),\ (x_3, y_3),\ .....\ (x_{n-1}, y_{n-1}),\ (x_n, y_n)$$

mit Hilfe des Polynoms genau eine Gleichung eines (typ. überbestimmten) Gleichungssystems aufgestellt. Auf diese Weise erhält man ein Gleichungssystem der Form

Mathematische Grundlagen des maschinellen Lernens

$$\begin{aligned} y_1 &= p_m x_1^m + p_{m-1} x_1^{m-1} + ... + p_2 x_1^2 + p_1 x_1 + p_0 \\ &\vdots \quad \vdots \qquad\qquad\qquad\qquad\qquad\qquad \vdots \\ y_n &= p_m x_n^m + p_{m-1} x_n^{m-1} + ... + p_2 x_n^2 + p_1 x_n + p_0 \end{aligned}$$

bzw. in Matrixform

$$\begin{pmatrix} x_1^m & x_1^{m-1} & . & . & . & x_1^2 & x_1 & 1 \\ x_2^m & x_2^{m-1} & . & . & . & x_2^2 & x_2 & 1 \\ \vdots & \vdots & . & . & . & \vdots & \vdots & \vdots \\ x_n^m & x_n^{m-1} & . & . & . & x_n^2 & x_n & 1 \end{pmatrix} \cdot \begin{pmatrix} p_m \\ p_{m-1} \\ . \\ p_2 \\ p_1 \\ p_0 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

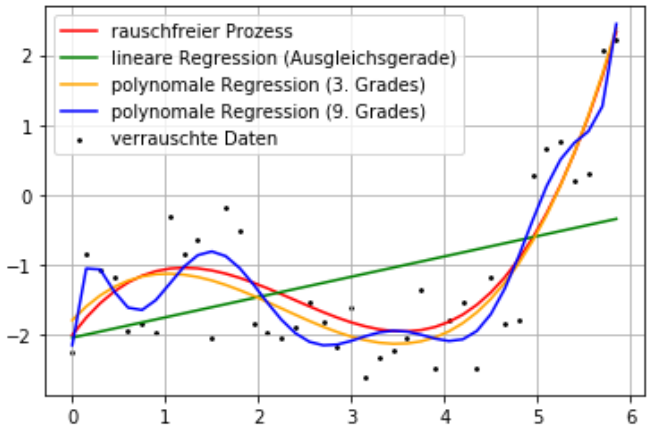
Für den in der Praxis häufig auftretenden Fall n>m erhält man für ein lin. Gleichungssystem

$$\underline{A} \cdot \vec{\xi} = \vec{b}$$

die ausgeglichene Lösung durch Lösung des Gleichungssystems

$$\underline{A}^T \underline{A} \cdot \vec{\xi} = \underline{A}^T \vec{b}$$

Genau das ist in dieser Teilaufgabe für verschiedene Polynome (Gerade, kub. Parabel, Polynom 9. Grades) zu realisieren.



Mathematische Grundlagen des maschinellen Lernens

Teilaufgabe 2: Gradientenabstieg

Vorgegeben ist eine Funktion *f(x,y)* und ein Startpunkt. Durch Gradientenabstieg soll ein lokales Minimum der Funktion gefunden werden. Mit dem Gradienten der Funktion

$$\vec{\nabla} f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

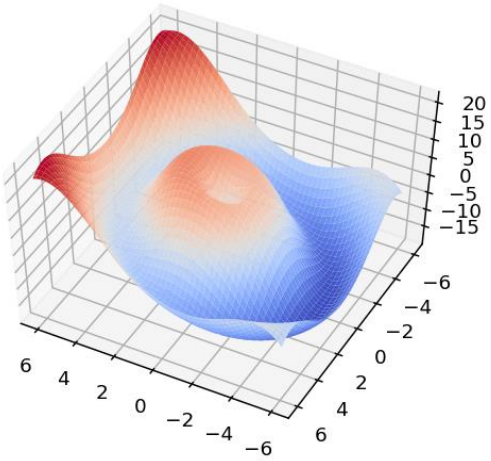
und der Iteration

$$\vec{x}_{n+1} = \vec{x}_n - \eta \cdot \vec{\nabla} f(x,y) \qquad \vec{x} = (x,y)^T$$

wird der Gradientenabstieg durchgeführt.

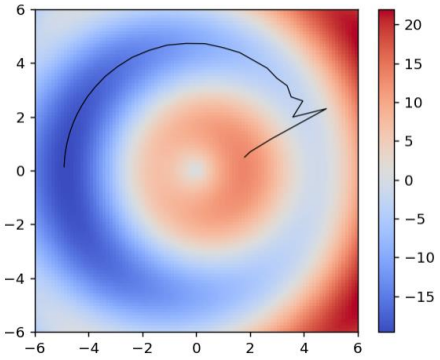
Vorbereiten: Im Versuch wird die folgende Funktion verwendet.

$$f(x,y) = 10 \cdot \sin\left(\sqrt{x^2 + y^2}\right) + 2x$$



Leiten Sie den Gradienten her und schreiben Sie dafür eine Funktion.

Ein Gradientenabstieg könnte dann z.B. so aussehen:



Mathematische Grundlagen des maschinellen Lernens

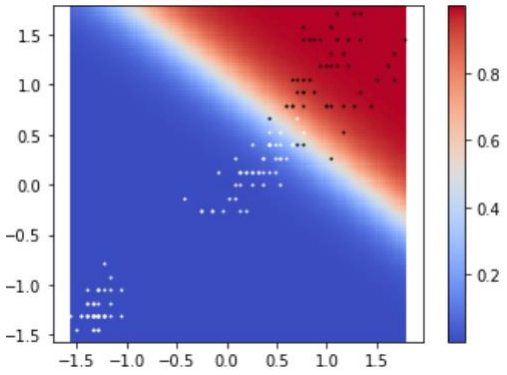
Versuchsdurchführung:

a) Führen Sie den Versuch mit den Schrittweitenfaktoren η = 0.1, 0.2, 0.3, 0.4 durch.
b) Verwenden Sie in der Iterationsformel für den Gradientenabstieg zusätzlich einen Momentumterm (s.u.) und führen Sie den Versuch mit η = 0.25 und den Momentumfaktoren α = 0, 0.1, 0.2, 0.3…. 0.8 durch.

$$1.\quad \vec{x}_{n+1} = \vec{x}_n - \eta \cdot \vec{\nabla} f(x,y) + \alpha \cdot \Delta\vec{x}_n$$
$$2.\quad \Delta\vec{x}_n = (\vec{x}_n - \vec{x}_{n-1})$$

Teilaufgabe 3: Training eines einzelnen Neurons

Anhand eines gegebenen Datensatzes (Iris-Datensatz) soll ein einzelnes Neuron mit logistischer Aktivierungsfunktion so trainiert werden, dass die Lilienart („iris virginica“) von anderen Lilienarten („iris setosa“ und „iris versicolor“) anhand des Neurons unterschieden werden kann.
Ausgegeben werden soll der Fehlerverlauf beim Training $E = \frac{1}{2}(t-y)^2$ sowie der Eingangsvektorraum mit den eingezeichneten Datenpunkten, z.B. so:



Versuchsdurchführung:

- Trainiert werden die Gewichte w_1 und w_2 sowie der Biaswert b.
- Die Fehlerverläufe der Durchführungsvarianten a)-c) sind zu dokumentieren.
- Die Fehlerverläufe sind mit einem gleitenden Mittelwert über m=50 Werte zu mitteln.
  Dies kann durch Faltung (*convolve*) mit einen Faltungskern [1, 1, 1, 1, ….., 1]/m erreicht werden.
- Der Schrittweitenfaktor sei η=0.1.

a) Führen sie das Training durch mit folgenden Randbedingungen:
   - Backpropagation mit quadratischem Fehlermaß (s.o.)
   - Daten wie gegeben.
   - Epochen = 15000

b) Führen sie das Training durch mit folgenden Randbedingungen:
- Backpropagation mit quadratischem Fehlermaß (s.o.)
- Daten (pro Spalte) vom Mittelwert befreien und auf Standardabweichung=1 bringen.
  Mittelwertbefreiung einer Datenspalte durch: $x = x - \text{mean}(x)$
  Standardabweichung=1 einer Datenspalte durch : $x = x/\text{std}(x)$
- Epochen = 15000

c) Führen sie das Training durch mit folgenden Randbedingungen:
- Backpropagation mit Crossentropie-Fehlermaß (s.o.)
- Daten (pro Spalte) vom Mittelwert befreien und auf Standardabweichung=1 bringen.
- Epochen = 5000