

MachineLearning for Visual Computing

Aufgabenblock 1

Christian Brändle - Gruppe 5

November 26, 2014

0.1 Einfaches Perceptron - Datengeneration

Gegeben sind vier Datensets a 100 Beobachtungen von 2-dimensionalen Eingangsdaten, welche normalverteilt sind. Die Figures in Tabelle 1 verdeutlichen dies.


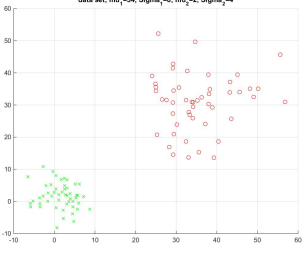
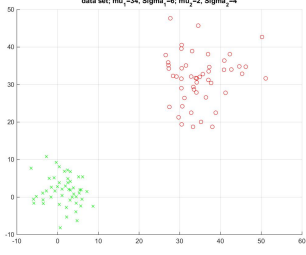
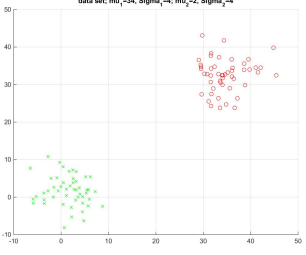
 <p><i>set</i>₁: $\mu_1 = 34, \Sigma_1 = 10$ <i>set</i>₂: $\mu_2 = 2, \Sigma_2 = 2$</p>	 <p><i>set</i>₁: $\mu_1 = 34, \Sigma_1 = 8$ <i>set</i>₂: $\mu_2 = 2, \Sigma_2 = 3$</p>
 <p><i>set</i>₁: $\mu_1 = 34, \Sigma_1 = 6$ <i>set</i>₂: $\mu_2 = 2, \Sigma_2 = 4$</p>	 <p><i>set</i>₁: $\mu_1 = 34, \Sigma_1 = 4$ <i>set</i>₂: $\mu_2 = 2, \Sigma_2 = 5$</p>

Table 1: data sets of separatable data

Chapter 1

Einfaches Perceptron - Perceptrontraining

1.1 Anzahl Iterationen

Die Anzahl der Iterationen hängt maßgeblich von der *margin* der separierbaren Daten ab. Je kleiner die margin, das heißt der Minimalabstand zwischen den beiden Mengen, desto mehr Iterationen sind notwendig, bis der Algorithmus terminiert. Dies ist aus der Angabe der Iterationen aus 1.2 ersichtlich. Weiters ist zu sehen, dass nur der online Percetron-Trainingsalgorithmus seine Iterationszahl ändert, die Werte für den batch Percetron-Trainingsalgorithmus bleiben konstant.

Bei größeren Datensätzen und extrem geringer margin ist ein deutlicher Sprung in der notwendigen Anzahl der Iterationen zu sehen.

1.2 Welchen Einfluß hat die Schrittweite γ

Der Einfluß der Schrittweite γ hat keinerlei Einfluß auf die Anzahl der notwendigen Iterationen sowohl der online als auch der batch Variante des Percetron-Trainingsalgorithmus. Die konstante Verteilung der Iterationen über ein γ von 0.1 bis 4 für die jeweiligen unterschiedlichen Datensätze in Tabelle 1.1 sowohl für online als auch batch Variante des Percetron-Trainingsalgorithmus zu sehen.

1.3 Daten und Entscheidungsgrenzen im \mathbb{R}^2

Die ermittelten Entscheidungsgrenzen des batch-learning sowie des online-learning Algorithmus sind für die einzelnen Datensätze und ein gegebenes $\gamma = 1$ in Tabelle 1.2 zu sehen.

Es ist zu sehen, daß meistens wesentlich bessere Entscheidungsgrenzen gefunden werden könnten, welche quasi normal auf die margin stehen würden und

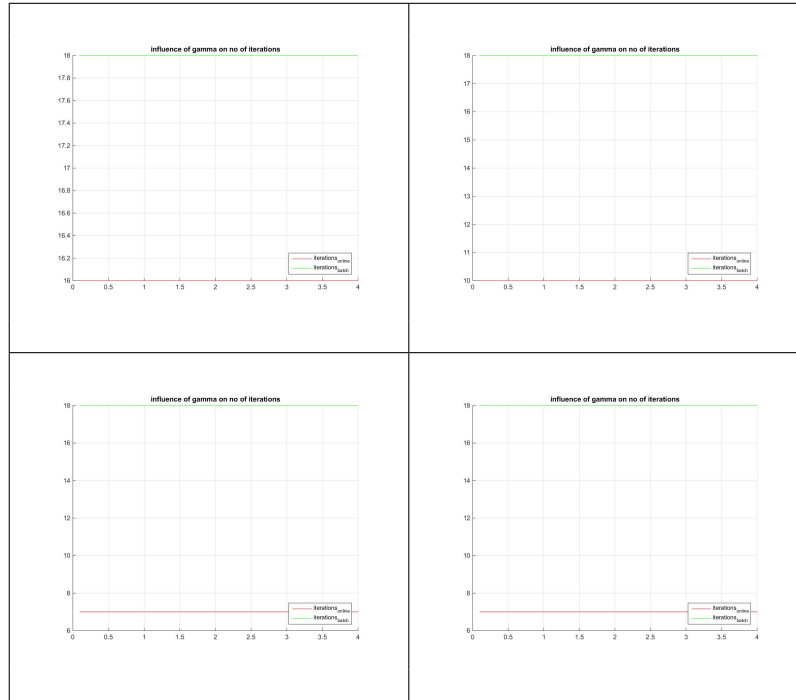


Table 1.1: iteration distribution over gamma

somit besser noch unbekannte Samples einer korrekten Klasse zuweisen würden.

1.4 Wie ist das Verhalten bei nicht linear separierbaren Daten

Der Algorithmus terminiert nicht da bei jedem Durchlauf Daten gefunden werden, welche in der falschen Klasse landen.

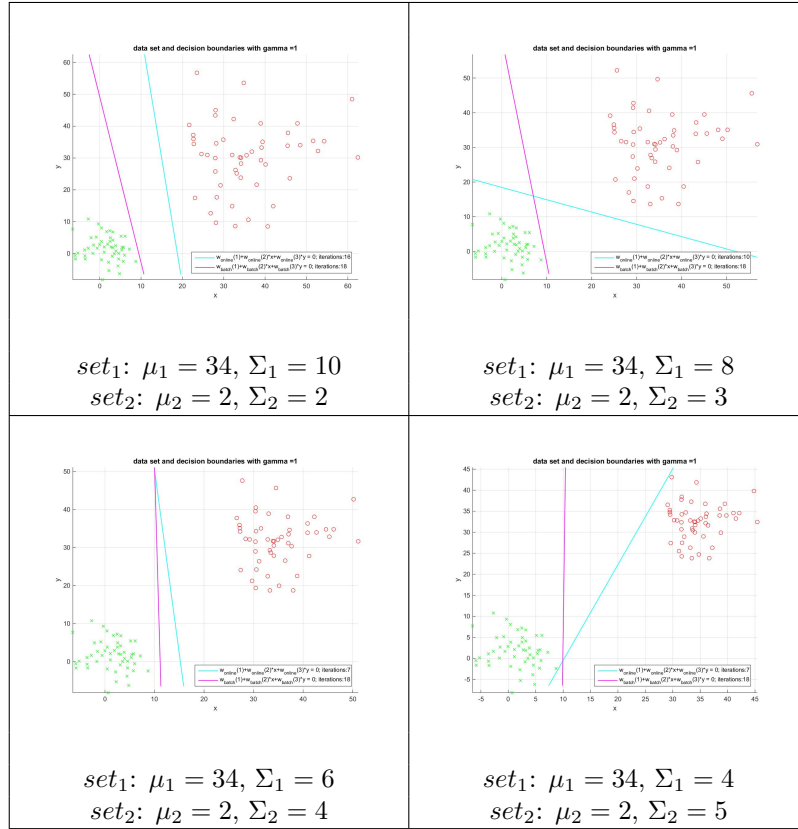


Table 1.2: data sets and decision boundaries

Chapter 2

Lineare Regression

Wir erstellen Daten im Bereich von $[0, 5]$ mit einer Schrittweite von 0.1 aus der Funktion:

$$y = x^2 - Gx + 1 \quad (2.1)$$

wobei $G = 10$ angenommen wird.

Des weiteren erstellen wir eine Trainingsmenge aus jedem sechsten Datenpunkt und fügen je einen Zufallswert aus $N(\mu = 0, \Sigma = 0.7)$ hinzu.

2.1 Gewichtsvektor mittels Gradientenabstieg bei quadratischer Fehlerfunktion

Der Gewichtsvektor w_{online} welcher aufgrund des online Verfahrens für *LMS* (least mean square) ermittelte wurde kommt in der Anwendung auf die Funktion $f(x) = x^2 - Gx + 1$ dem optimalen w^* zwar nahe, erreicht dessen Qualität aber nicht. Eine Verringerung der Fehlertoleranz würde hier sicher Abhilfe schaffen. In der Anwendung auf die Funktion $f(x)$ ergeben sich wie in Figur 2.1 gezeigt für die Dimensionalität 3 die gezeigten Kurven.

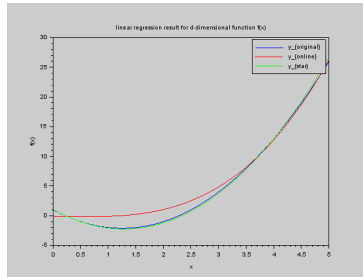


Figure 2.1: online vs batch LMS

2.2 Optimaler Gewichtsvektor w^*

Den optimalen Gewichtsvektor w^* kann man über die Bestimmung der Pseudoinversen A^+ für die Gleichung $Aw^* = b$ ermitteln. Die Gleichung $w^* = A^+b$ bestimmt das optimale w^* . Dabei berechnet sich A^+ wie folgt:

$$A^+ = (AA^T)^{-1} \quad \text{für invertierbare } AA^T \quad (2.2)$$

$$A^+ = (AA^T)^{-1} + \lambda I \quad \text{für nicht invertierbare } AA^T \text{ mit } \lambda \ll 1 \quad (2.3)$$

Das ermittelte Ergebnis für w^* mit normalisierten Daten aus dem \mathbb{R}^2 lautet: $w^* = (1.0106678, -10.0086, 1.8960336, 0.0262634)$.

2.3 Konvergenzverhalten in Abhängigkeit von γ

In Figur 2.2 ist zu sehen, daß die Anzahl der notwendigen Iterationen um zu einem Optimum zu gelangen mit der Größe von γ exponentiell abnimmt bis sie bei einem Wert von $\gamma^{hat} \approx 0.0001$ ein Minimum erreichen. Danach steigt die Anzahl der notwendigen Iterationen wieder an.

In Figur 2.3 ist zu sehen, daß bei Erhöhung von γ über den optimalen Wert von $\gamma^{hat} \approx 0.0001$ hinaus die Anzahl der Iterationen wieder zunimmt um anschließend einen etwas willkürlichen Verlauf anzunehmen. Auf jeden Fall ist festzustellen, dass nach dem erneuten Fallen der Iterationen kein brauchbares Resultat mehr zustande kommt. Exemplarisch wurde dies für ein $\gamma = 0.0035$ durchgerechnet. Eigentlich wäre erwartet worden, daß der Algorithmus über einem gewissen γ einfach die maximal zulässige Anzahl an Iterationen durchführt und dann ohne konvergiertes Ergebnis terminiert.

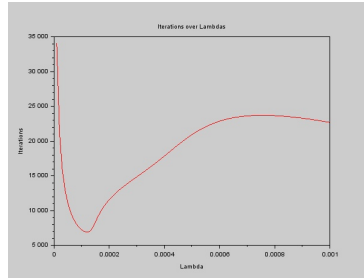


Figure 2.2: convergence over gamma

2.4 Erwartungswert und Varianz von w^* im Bezug zu Dimensionalitt d von $f(x)$

Erwartungswert und Varianz von w^* im Bezug zu Dimensionalitt d von $f(x)$ zeigt, dass je höher die Dimensionalitt d steigt, desto höher auch die Vari-

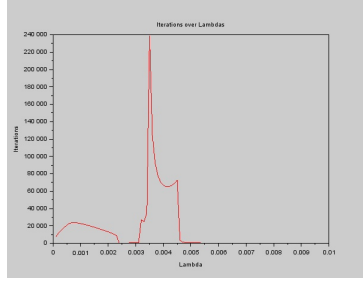


Figure 2.3: convergence over gamma beyond optimal region

anz Σ der ermittelten Werte für w^* für verrauschte Eingangsdaten wird. Dies ist auch klar, da höherdimensionale Koeffizienten von w^* höhere Frequenzen repräsentieren, welche durch das Verrauschen der Eingangsdaten stärker betroffen sind als niedrigere Frequenzen. Interessanterweise steigt die Varianz aller Koeffizienten von w^* in Tabelle 2.1 nur bis zu einer Dimensionalität des Problems von 6, danach fällt sie wieder. Es ist aber anzunehmen, dass dies zum Einen aufgrund von Unzulänglichkeiten beim generieren der Pseudoinversen zur Berechnung von w^* liegt und zum Anderen von der fehlenden Betrachtung noch höherer Dimensionen, wo sich globale Anstieg der Varianz sicherlich fortsetzt.

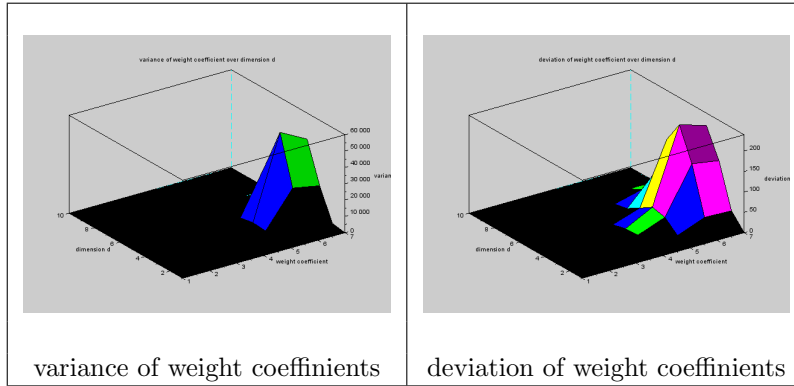


Table 2.1: variance and deviation of w^* over dimension

Des weiteren ist in Figur 2.4 festzustellen, dass für höherdimensionale Komponenten für w^* durch overfitting der Kurve $f(x)$ die Varianz zunimmt, zu sehen in den Schwingungen der Kurven.

2.5 Mittlere quadratische Abweichung von $f_{w^*}(x^*)$

Wie erwartet, steigt auch die mittlere quadratische Abweichung von $f_{w^*}(x^*)$ mit der Zunahme der Dimension d an, wie in Figur 2.5 zu sehen ist.

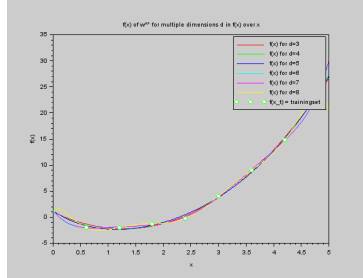


Figure 2.4: overfitting of $f_{w^*}(x)$ curves over dimension

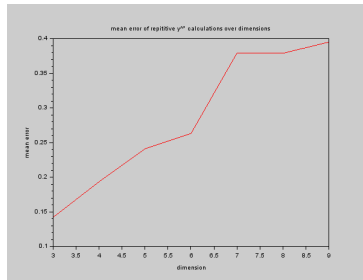


Figure 2.5: mean error of $f_{w^*}(x^*)$ curves over dimension

2.6 Gewichtsvektor w^* bei ungestörter Trainingsmenge über Dimension d

Die resultierenden Kurven für $f_{w^*}(x)$ welche aus der ungestörten Trainingsmenge ermittelt wurden fitten die Originalpunkte aus der Originalkurve $y = 2x^2 - Gx + 1$ naturgemäß wesentlich besser als die Kurven, welche auf der verrauschten Trainingsmenge basieren. Dies ist in Tabelle 2.2 zu sehen. Der Ausreißer bei Dimension 9 ist wiederum auf Schwächen in der Bestimmung der Pseudoinversen zurückzuführen.

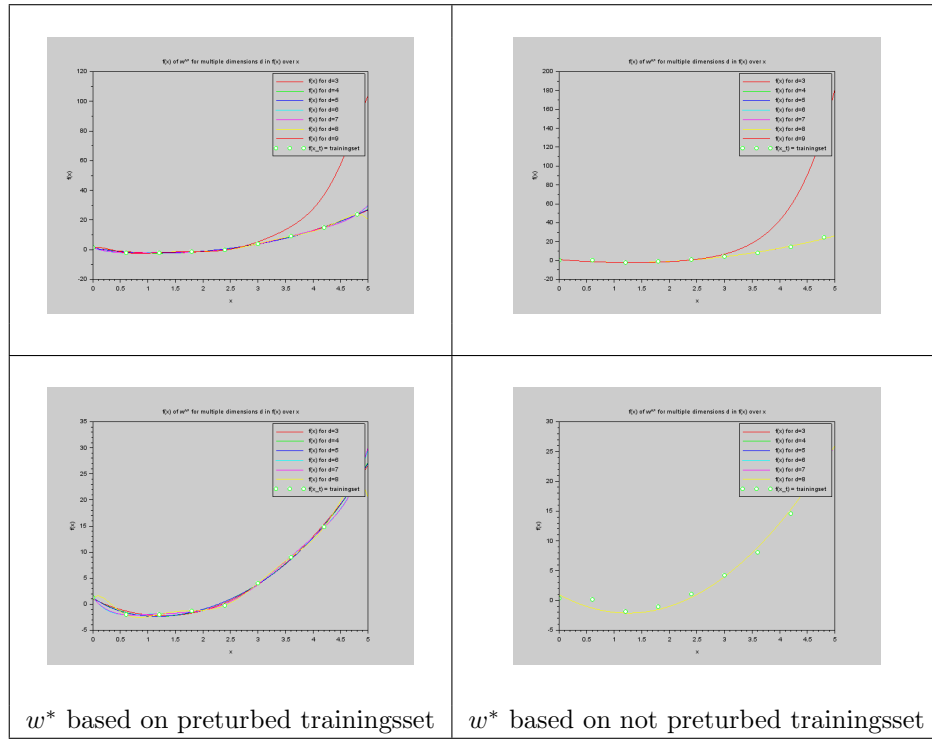


Table 2.2: $f_{w^*}(x)$ over dimension