
Perzeptron-Lernalgorithmus

FACHHOCHSCHULE VORARLBERG
MASTER INFORMATIK

DOZENT

HANS-GEORG BEYER

VORGELEGT VON

GRIESSER, HÄMMERLE, RIEDMANN

DORNBIRN, 11.12.2016

Kurzreferat

Diese Übung Dient als Einstieg in Oktave und es wird der in der Vorlesung vorgestellte Perzeptron-Lernalgorithmus implementiert. Durch Experimente mit unterschiedlichen Eingangsvektoren sollen Wege gefunden werden, den Algorithmus zu verbessern.

Inhaltsverzeichnis

1 Aufgabenstellung	4
2 Perzeptron-Lernalgorithmus	5
2.1 Octave Implementierung	5
3 Experimente	6
3.1 Verhalten bei Nicht-Separierbarkeit	7
3.2 Mehrdimensionales Experiment	7
4 Verbesserung des Algorithmus	8
Literaturverzeichnis	10

1. Aufgabenstellung

1. Machen Sie sich mit Octave vertraut. Dazu re-implementiere man den Perzeptron-Lernalgorithmus von Folie 32 in Octave. (Implementation von Folie 33f kann als Ausgangslösung verwendet werden.)
2. Man experimentiere mit weiteren selbstgewählten Datensätzen (2D, 3D und höhere Merkmalsräume) und untersuche Fälle, in denen die lineare Separierbarkeit der Daten nicht möglich ist.
3. Für einen derartigen Fall stelle man die Anzahl der Missklassifikationen über die Iterationen dar. Was beobachtet man?
4. Wie kann diese Beobachtung zur Verbesserung der Separationsleistung des Algorithmus verwendet werden? Schlagen Sie auf Basis dieser Überlegungen eine verbesserten Lernalgorithmus vor.

2. Perzeptron-Lernalgorithmus

2.1 Octave Implementierung

Für die Implementierung eines Perceptrons in Octave wird die bekannte Implementierung (vgl. (1)) als Basis herangezogen. Zusätzlich werden kleinere Modifikationen vorgenommen. Zum einen wird der Skalarwert `w_found` eingefügt, der klar angibt, ob die Funktion durch den Fund einer passenden Gewichtung oder durch das Überschreiten der maximalen Iterationsanzahl beendet wurde. Zum anderen wird der Vektor `w_history` hinzugefügt, um Aufschluss auf die Entwicklung des Vektors `w` zu geben.

3. Experimente

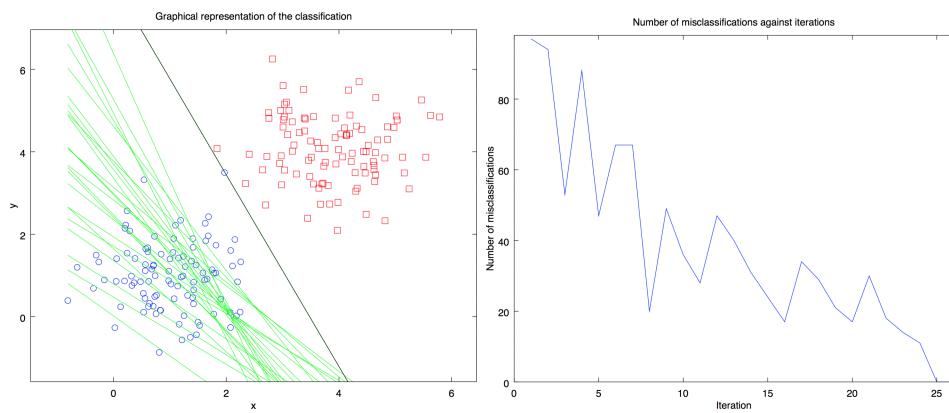


Abbildung 1: separierbar

Bei einer Eingabe eines Datensets mit einer möglichen linearen Separation findet der Lernalgorithmus nach einer nicht festgelegten Anzahl an Iterationen eine passende Gewichtung. Als Voraussetzung dafür gilt, dass dem Algorithmus eine genügende Anzahl an Iterationen gewährt werden, welche sich aber im Voraus nicht eindeutig bestimmen lässt, da zum einen die Initialgewichtung zufällig gewählt wird und zum anderen keine zielgerichtete Suche angewandt wird, d.h. der Algorithmus wählt teilweise auch eine weniger optimale Gewichtung für den nächsten Iterationsschritt.

3.1 Verhalten bei Nicht-Separierbarkeit

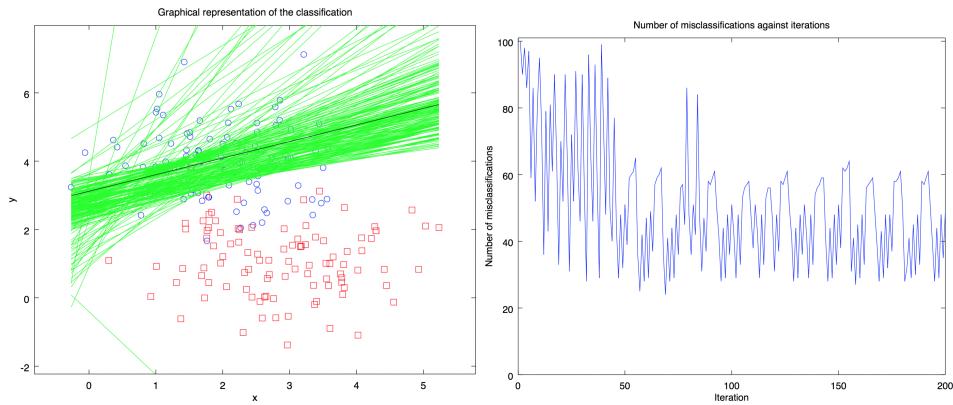


Abbildung 2: nicht separierbar

Eine Untersuchung von einem Lerndatenset mit hingegen keiner möglichen, linearen Separierung (vgl. Abbildung 2) ist auffallend, dass der Algorithmus zu keinem Ergebnis kommt und daher nur durch Erreichen der maximalen Iterationsanzahl zu einem Ende kommt. Mit einem ausreichend großen oberen Limit, stellt sich dann nach einigen Iterationen eine Art Gleichgewicht ein, in dem die gefundenen Gewichtungen w regelmäßig zwischen ähnlichen Werten hin- und herspringen.

3.2 Mehrdimensionales Experiment

- Eingangsvektoren: 100 Vektoren mit 0.8 Standardabweichung um $(1, 1, 1, 1, 1)$ und $(4, 4, 4, 4, 4)$ verteilt.
- Ergebnisse:
 - Gewichtung $w = (-0.044289, -0.056365, -0.167347, 0.043915, -0.144743, 0.971585)$
 - Anzahl Iterationen: $n_i = 11$

Da zwei-dimensionale Daten leicht darstellbar und vorstellbar sind werden im Folgenden nur zwei-dimensionale Daten betrachtet.

4. Verbesserung des Algorithmus

Wie bereits in Kapitel 3 angemerkt, ist das Verhalten des Algorithmus nicht zielgerichtet. Eine Verbesserung dieses Verhaltens ist aber möglich durch eine Vorverarbeitung der Eingangsdaten (vgl. (2), (3)). Im Detail wird dabei das komplette Datenset “normalisiert” indem jeder der Vektoren im Datenset auf eine Länge innerhalb des Intervalls $[0, 1]$ skaliert wird.

Wie in den Abbildungen 3 und 4 verhält sich der Algorithmus mit dieser Normalisierung deutlich zielgerichtetet bei der Findung einer anwendbaren Gewichtung sofern diese möglich ist. Bei Datensets die nicht-separierbar sind, gibt dieser Algorithmus ebenfalls den Vorteil, dass er in der Lage wäre zu erkennen, wenn eine Separation nicht möglich ist. Dies wird nämlich erkannt, wenn der Algorithmus zu einer Verschlechterung bei der Anzahl der Missklassifikationen führt (vgl. Abbildung 5).

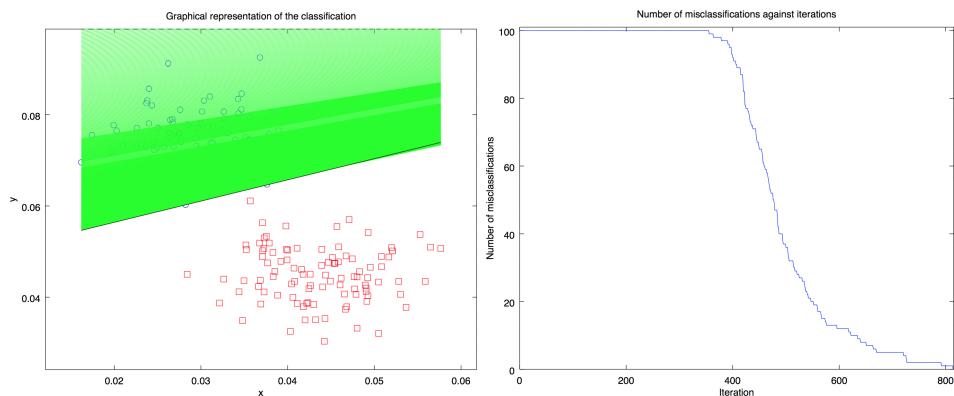


Abbildung 3: separierbar

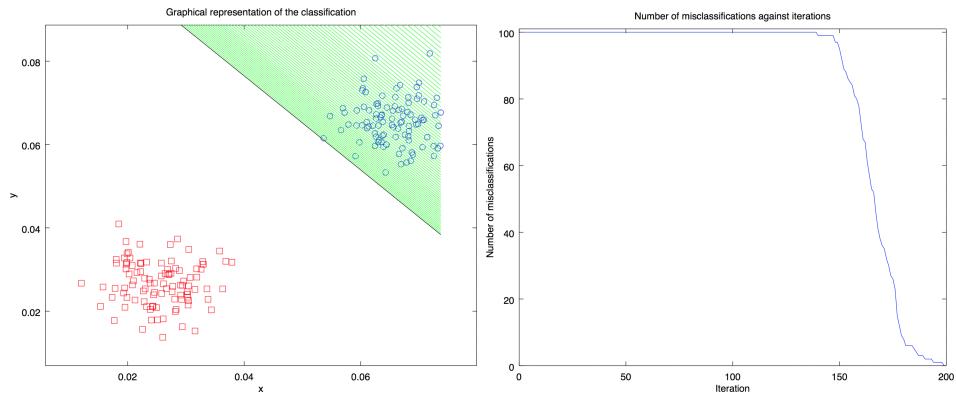


Abbildung 4: separierbar

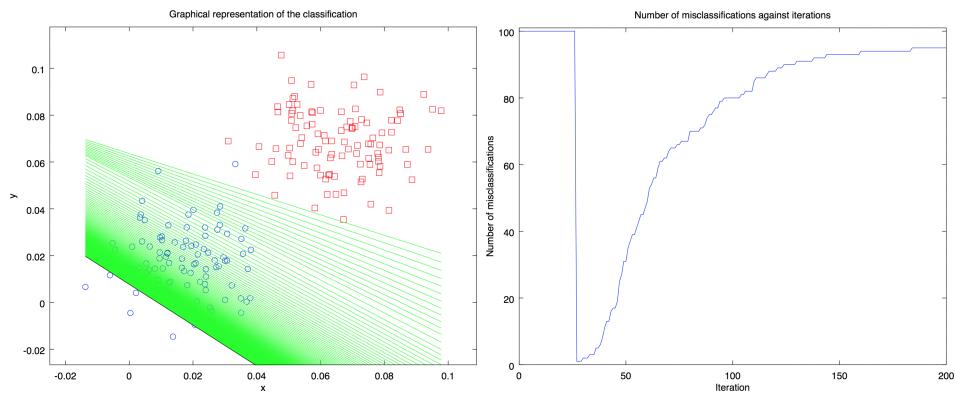


Abbildung 5: nicht separierbar

Literaturverzeichnis

- [1] H.-G. Beyer, “Computational Intelligence and Optimization - An Introduction.”
- [2] K. N. Rotich, “Forecasting of Wind Speeds and Directions with Artificial Neural Networks.” <https://www.doria.fi/bitstream/handle/10024/98414/Rotich%20%28M.%20Sc%20Thesis%29.pdf> abgerufen am 09.12.2016.
- [3] R. Rojas, “Theorie der neuronalen Netze: Eine systematische Einführung.” <https://books.google.at/books?id=YoPLBgAAQBAJ&lpg=PA88&ots=twxLUIjDT5&dq=perzeptron%20ernalgorithmus%20normalisierte%20vektoren&hl=de&pg=PA88#v=onepage&q=perzeptron%20ernalgorithmus%20normalisierte%20vektoren&f=false> abgerufen am 09.12.2016.