Aufgabe 1
Machine Learning for Visual Computing

Philipp Omenitsch, xx and xx

December 11, 2015

Abstract

Datengenerierung, einfacher Klassifikator, Perzeptron

Contents

1 Datengenerierung

2 Einfacher Klassifikator

Der einfache Klassifikator stellt in unserem Fall eine memory Funktion da. Das heißt, dass wenn wir einen Datensatz schon gesehen haben, geben wir dessen Werte aus. Wenn wir diesen noch nicht gesehen haben geben wir einen negativen Wert zurück. Nachdem wir alle möglichen Zahlenkombinationen gesehen haben, haben wir einen perfekten Klassifikator.

2.1 Warum overfitted die Funktion memory? Was bedeutet Overfitting in Zusammenhang mit dem Lernen von Klassifikatoren?

Der Klassifikator hat auf dem Trainingset 100 % da er die selben Werte wiedererkennt. Dafür kann man mit diesem Klassifikator nicht generalisieren. Wenn auch nur einer der Werte minimal abweicht bekommt man immer einen negativen Returnwert obwohl es eigentlich viel wahrscheinlicher ist, dass dieser Wert den selben output wert hat als sein minimal unterschiedlicher Nachbar. Daher kann man mit diesem Klassifikator nicht generalisieren.Durch die fehlende Generalisierung werden viele Daten falsch klassifieziert. Dadurch ist die Performance außerhalb des Testsets, wie man bei unseren Resultaten sehen kann sehr schlecht.

2.2 Overfitting von polynomiellen Klassifikatoren

Overfitting ist ein generelles Problem im maschinellen Lernen. Es ist sehr wichtig, dass ein Lernalgorithmus gut generalisiert um richtig auf neue Daten reagieren zu können. Man muss sich daher bei allen Klassfikitaroen, welche aus einem Polynom mit nachfolgendem Schwellwert bestehen überlegen ob diese nicht Overfitting. Speziell betroffen sind Polynome höherer Ordnungen, da diese dazu tendieren auch Ausreißer irgendwie noch in das Polynom hineinzubiegen und damit nicht mehr gut generalisieren. Das zeigt sich dann daran, dass der Algorithmus auf echten, noch nicht gesehenen Daten sehr schlecht reagiert.

Ein zusätzliches Problem bei Polynomen mit Schwellwert ist, dass man diese vorher fix wählt. Daher man muss genau wissen, wie die Daten später vorkommen werden um eine sinnvolle hypothese abgeben zu können. Das verstärkt das Problem mit Overfitting noch.

3 Perceptron

Das Perceptron ist eine einfache Variante eines neuronalen Netzes. Das Prinzip wurde erstmals im Jahre 1958 von Frank Rosenblatt veröffentlicht[**?**]. Es handelt sich dabei um eine lineare Diskriminantenfunktion. Während des Lernvorganges wird ein Vektor mit Gewichten erstellt, welcher dann anschließend eine Klassifikation vornimmt. Das Ergebnis wird anschließend durch eine Signum-Funktion dargstellt.

3.1 Untersuchen sie den Trainingsalgorithmus: Welche Eigenschaften der Daten beeinflussen die durchschnittliche Anzahl an Iterationen bis eine Lösung w* gefunden wurde?

Wenn die Daten linear separierbar ist kann immer eine obere Grenze für die Anzahl der Iterationen bis zur Konvergenz zu einer optimalen Lösung gefunden werden. Die lineare Separierbarkeit ist eine Eigenschaft die die Anzahl der Schritte beeinflust, je besser separierbar desto schnellere Konvergenz, weil die Verschiebung in jedem Schritt optimal ist. Allerdings ist bei nicht linear separierbaren Daten nicht garantiert, dass immer eine optimale Lösung gefunden wird. Außerdem terminiert der Algorithmus nie, weil die Daten ja nicht linear separiert werden können.

Es kommt auch darauf an, wieviele Elemente welcher Klasse existieren, wenn Klasse 1 viel mehr Elemente als Klasse 2 hat, wird die Entscheidungsgrenze so verschoben dass Klasse 1 öfter richtig klassifiziert wird (weil ja mehr Elemente mehr zum Gewichtsvektor beitragen)

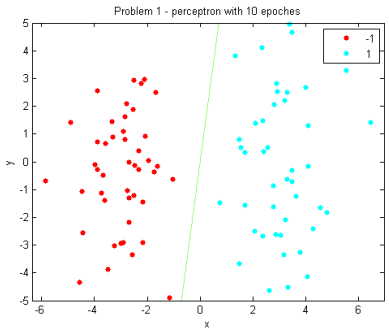
3.2 Welchen Einufluss hat die Schrittweite?

Die Schrittweite oder Lernrate γ kann man so verstehen, dass die Entscheidungsgrenze immer um den Richtungsvektor * lernrate verschoben wird. Wenn die Lernrate zu groß wird, können optimale Lösungen Übersprungen werden (wenn wir uns den Lösungsraum im Dreidimensionalen vorstellen so kann ein Tal einfach übersprungen werden), ist die Lernrate zu klein, so kann es passieren, dass man in einem lokalen Optimal stecken bleiben kann.

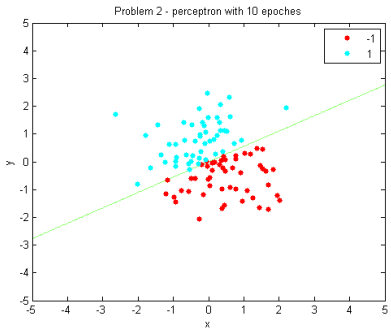
3.3 Plotten Sie Daten und Entscheidungsgrenze (analog zu Punkt 1.1).

3.4 Vergleichen Sie das Perzeptron mit der Funktion memory. Worin liegt der Unterschied?

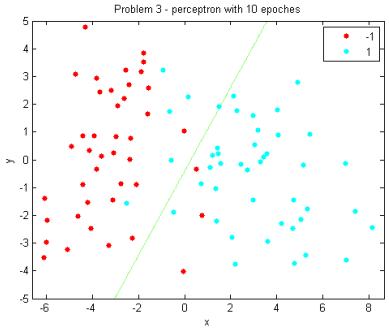
Bei memory werden nur Punkte erkannt die im Trainingsset vorhanden waren. Bei allen anderen Punkte ratet die memory Funktion. Das Perzeptron widerum kann auf Grund der eingegangen Daten generalisieren. Dadurch können neue,



(a) Nicely spread data



(b) seperable data but close together



(c) Not seperable data

Figure 1: Verschiedene plots der Perceptron Klassifier nach 10 Epochen

unbekannte Daten dennoch mit einer höheren Wahrscheinlichkeit richtig klassifiziert werden. Daher führt das Perzeptron eine Entscheidungsgrenze ein, welche dann für die neuen Daten angewandt werden kann. Dadurch wird das Overfitting vermindert oder ganz vermieden.

3.5 Wie ist das Verhalten bei nicht linear separierbaren Daten?

Die Entscheidungsgrenze bildet ein Ausgleichsgerade zwischen den zwei Klassen, wie in der Vorlesung gesehen, minimiert sie theoretisch den Abstand zu beiden Klassen, allerdings osziliert sie in aufeinanderfolgenden Iterationen in der Region in der sich die Klassen überlappen, weil jede falsche Klassifikation die Entscheidungsgrenze verschiebt, weil es aber immer falsche Klassifikationen geben wird, wird auch die Entscheidungsgrenze immer hin und her geschoben.

References

[Ros58] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.