1 Das MCP-Modell

Dieses sehr frühe Modell des maschinellen Lernens beherrscht eine binäre Klassifizerung der Objekte. Entweder ist das zu-klassifizierende Objekt ein Objekt dieser Klasse oder es ist ein Objekt einer anderen Klasse. Ein Modell dieser Art der Klassifizierung ist das MCP-Modell. Das MCP-Neuronenmodell basiert auf dem Modell einer Nervenzelle. Eine biologische Nervenzelle wird erst dann bei elektrischen Signalen aktiviert, wenn der Stromfluss einen gewissen Schwellenwert erreicht hat und mathematisch wird dies mit der Heaviside-Funktion modelliert.

$$\phi(z) = \begin{cases} 1, & z \geq 0 \\ -1, & z < 0 \end{cases} \tag{1}$$

Kann das zu-klassifizierende Objekt in quantitative Eigenschaften übersetzt und Diese normiert werden, kann hieraus eine Matrix $t \in \mathbb{R}^{n+1\times 1}$gemacht werden. Dabei wird jedes Feature, also jede beschriebene Eigenschaft einer Reihe zugeordnet. Damit beinhaltet jede Zeile die gesamten Informationen eines Datenpakets. Dabei ist x_i die i-te Eigenschaft des Objektes. In dieser Variante wird x_0 als 1 definiert. Dieser hat den Zweck in dem Skalarprodukt den Schwellenwert wieder auszugeben.

$$t = (1, x_1, ..., x_{n-1}, x_n) \tag{2}$$

Diese Matrix wird nun mit einem Gewichtsvektor $\omega \in R^{1\times n+1}$multipliziert. Dieser Gewichtsvektor ist ein Vektor , wobei jeder Eintrag ein individuelles Gewicht für jede Eigenschaft festlegt. Der nullte Eintrag ist hierbei der Schwellenwert z, der so gespeichert wird.

$$\omega = \begin{pmatrix} -z \\ y_1 \\ ... \\ y_{n-1} \\ y_n \end{pmatrix} \tag{3}$$

Für die Multiplikation gilt also nach den Regeln der linearen Algebra:

$$V^{m\times n}.W^{n\times k} = P^{m\times k} : \forall P_{g,h} : P_{g,h} = \sum_{1\leq i\leq n} V_{g,i}W_{i,h} \implies \omega{\cdot}t = -z + \sum_{1\leq i\leq n} x_i y_i \tag{4}$$

Damit lässt sich aus den Gewichten die Heaviside-Funktion neu definieren.

$$\phi(\omega \cdot t) = \begin{cases} 1, & \sum_{1 \leq i \leq n} x_i y_i \geq z \\ -1, & \sum_{1 \leq i \leq n} x_i y_i < z \end{cases} \tag{5}$$

Nun werden diese Schritte iterativ mit der Perzeptronen-Lernregel wiederholt und die Gewichte in ω angepasst. Die Berechnung des $k+1$-ten Gewichtes ist gegeben durch, wobei y das richtige Ergebnis ist und η die Lernrate ist:

$$w_{i,k+1} := w_{i,k} + \eta(y - \phi(\omega \cdot t))x_i \tag{6}$$

Wird hier also die richtige Eingabe durch die Heaviside-Funktion gegeben, ergbt sich automatisch eine Änderung des Gewichts von Null.

$$w_{i,k+1} := w_{i,k} + \eta(y - y)x_i \tag{7}$$

$$w_{i,k+1} := w_{i,k} \tag{8}$$

Ist das Ergebnis allerdings nicht das Richtige, so gibt es zwei Fälle. Wurde es als richtig vorhergesagt, war aber falsch, dann gilt:

$$w_{i,k+1} := w_{i,k} + \eta(-1 - 1)x_i \tag{9}$$

$$w_{i,k+1} := w_{i,k} - 2\eta x_i \tag{10}$$

Wurde es als falsch vorhergesagt, war aber richtig, dann gilt:

$$w_{i,k+1} := w_{i,k} + \eta(1 - -1)x_i \tag{11}$$

$$w_{i,k+1} := w_{i,k} + 2\eta x_i \tag{12}$$

Die jeweilige Änderung hängt also von der Lernrate und dem jeweiligen Wert der Eigenschaft ab. Umso größer beide Werte sind, umso größer und umso radikaler ist also die Änderung. Durch diesen Algorithmus für die Berechnung werden die Werte der Gewichte über die Iterationen verfeinert. Nach dem Trainieren kann nun mithilfe der Gewichte eine Eingabe bewertet werden. Durch die Verwendung validierender Daten wird das System getestet. Ist es dann erfolgreich genug, so wird das System angenommen.

2 Konvergenzbedingung der linearen Separabilität

Damit dieser Lernprozess konvergent ist, muss jeder einzelne Wert der Gewichte auch gegen einen bestimmten Wert konvergieren. Dafür müssen die beiden Punktmengen durch eine Hyperebene getrennt werden. Es seien zwei Punktmengen $P_1 \in \mathbb{R}^n$ und $P_2 \in \mathbb{R}^n$. Unter der Annahme es gibt eine Hyperbene H, die die beiden Punktmengen trennt, gibt es also auch zu jedem Punkt der beiden Mengen eine parallele Ebene G.

$$E : b = \sum_{i=1}^{n} a_i x_i : \forall x \in P_1 \wedge \forall y \in P_2 : \exists G : c = \sum_{i=1}^{n} a_i x_i \tag{13}$$

Da der Punkt also in der parallelen Ebene liegen muss gilt also nach Einsetzen des Punktes:

$$x \in G \implies c = \sum_{i=1}^{n} a_i x_i^i \tag{14}$$

Nun wählen wir einen der Achsenschnittpunkte beider Ebenen aus. Da sie parallel sind und die mindestens einen Schnittpunkt mit der Achse haben, da nicht alle Koeffizienten Null sind, gilt also für die Achsenschnittpunkte:

$$E : b = a_k x_k \tag{15}$$

$$G : a_k x_k = \sum_{i=1}^{n} a_i x_i^i \tag{16}$$

Nun kann man die beiden Schnittpunkte der parallelen Ebenen vergleichen. Liegt jeder Punkt x aus P_1 nun in einer Ebene, die eine negative Koordinatentransformation nutzt, so gilt:

$$b < \sum_{i=1}^{n} a_i x_i \tag{17}$$

Dann gilt das Gegenteil für die andere Punktmenge.

$$b > \sum_{i=1}^{n} a_i y_i \tag{18}$$

Setzt man nun beide Gleichungen ein, so gilt:

$$\sum_{i=1}^{n} a_i y_i^i < \sum_{i=1}^{n} a_i x_i^i \tag{19}$$

Da dies nun für jedes Punktepaar gelten muss, gilt:

$$\forall x \in P_1 \wedge \forall y \in P_2 : \sum_{i=1}^{n} a_i y_i < \sum_{i=1}^{n} a_i x_i \tag{20}$$

Dies ist nun die Vorraussetzung für die Konvergenz dieses Lernmodells.

3 Der Beweis

Dafür teilen wir bei den Lerniterationen die Kalibirierung in zwei Teilen ein. Zuerst wird der Teil analysiert, der als richtig bestimmt werden soll. Für die Änderung der Gewichte gilt:

$$\omega_{i,k+1} = \omega_{i,k} + \eta(1 - \phi(\omega \times X))x_i \tag{21}$$

Wenn nun hier jeder Änderungsterm Null ist, ändert sich also nichts mehr, da dann gilt:

$$\forall i : \omega_{i,k+1} = \omega_{i,k} + 0 \tag{22}$$

$$\forall i : \omega_{i,k+1} = \omega_{i,k} \tag{23}$$

Damit gilt also für die Konvergenz dieser Teilsumme:

$$\forall X : 0 = \eta(1-\phi(\omega \times X))x_i \implies \forall X 1 = \phi(\omega \times X) \implies \forall x \omega \times X \geq 0 \tag{24}$$

Wendet man sich nun der anderen Teilmenge zu, gilt also:

$$\omega_{i,k+1} = \omega_{i,k} + \eta(-1 - \phi(\omega \times Y))y_i \tag{25}$$

Wenn nun hier jeder Änderungsterm Null ist, ändert sich also nichts mehr, da dann gilt:

$$\forall i : \omega_{i,k+1} = \omega_{i,k} + 0 \tag{26}$$

$$\forall i : \omega_{i,k+1} = \omega_{i,k} \tag{27}$$

Damit gilt also für die Konvergenz dieser Teilsumme:

$$\forall Y : 0 = \eta(-1-\phi(\omega \times Y))y_i \implies \forall Y : -1 = \phi(\omega \times X) \implies \forall Y : \omega \times Y < 0 \tag{28}$$

Damit also nun beide Teilsummen konvergieren, gilt:

$$\forall Y \wedge \forall X : \omega \times Y < 0 \wedge \omega \times X > 0 \implies \forall i : \omega_{i,k+1} = \omega_{i,k} \tag{29}$$

Diese Bedingungen werden zusammengefasst.

$$\forall Y \wedge \forall X : \omega \times Y < \omega \times X \forall i : \omega_{i,k+1} = \omega_{i,k} \tag{30}$$

Damit ist gezeigt, dass dieses Modell konvergiert, sobald die zwei Datenmengen separabel sind.

4 Logische Gatter mit dem Perzeptron

Es gibt drei logische Gatter, die es zu testen gilt. Dabei definieren wir die zwei Outputs als ihre binären Wahrheitswerte. Um nun zu beweisen, dass es keinen Gewichtsvektor gibt, der die oben-genannte Konvergenzbedingung erfüllt.

4.1 AND-Gatter

Die Punkte sind nun:

$$\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{array} \tag{31}$$

Daraus folgen die vier Gleichungen:

$$0a_1 + 0a_2 < a_1 + a_2 \tag{32}$$

$$1a_1 + 0a_2 < a_1 + a_2 \tag{33}$$

$$0a_1 + 1a_2 < a_1 + a_2 \tag{34}$$

Daraus folgen die Gleichungen:

$$0 < a_1 + a_2 \tag{35}$$

$$0 < a_2 \tag{36}$$

$$0 < a_1 \tag{37}$$

Da diese Bedingungen alle miteinander konform sind, gibt es eine Konfiguration, die die Konvergenzbedingung erfüllt.

4.2 OR-Gatter

Die Punkte sind nun:

$$\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{array} \tag{38}$$

Daraus folgen die vier Gleichungen:

$$0a_1 + 0a_2 < a_1 + a_2 \tag{39}$$

$$0a_1 + 0a_2 < a_2 \tag{40}$$

$$0a_1 + 0a_2 < a_1 \tag{41}$$

Daraus folgen die Gleichungen:

$$0 < a_1 + a_2 \tag{42}$$

$$0 < a_2 \tag{43}$$

$$0 < a_1 \tag{44}$$

Da diese Bedingungen alle miteinander konform sind, gibt es eine Konfiguration, die die Konvergenzbedingung erfüllt.

4.3 XOR-Gatter

Die Punkte sind nun:

$$\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{array} \tag{45}$$

Daraus folgen die vier Gleichungen:

$$0a_1 + 0a_2 < a_1 \tag{46}$$

$$0a_1 + 0a_2 < a_2 \tag{47}$$

$$a_1 + a_2 < a_1 \tag{48}$$

$$a_1 + a_2 < a_2 \tag{49}$$

Daraus folgen die Gleichungen:

$$0 < a_1 \tag{50}$$

$$0 < a_2 \tag{51}$$

$$0 > a_1 \tag{52}$$

$$0 > a_2 \tag{53}$$

Da diese Bedingungen nicht miteinander konform sind, gibt es keine Konfiguration, die die Konvergenzbedingung erfüllt.