|  |  |
| --- | --- |
| **Machine Learning**  Die drei Arten des Machine Learnings:  1. Überwachtes Lernen  2. Verstärkendes Lernen  3. Unüberwachtes Lernen  **Überwachtes Lernen Vorhersagen**    Modell erlernen anhand von Trainingsdaten, um Voraussagen über unbekannte oder zukünftige Daten treffen können. Die richtige Antwort ist im Vorhinein bekannt.  **Typ 1: Klassifizierung**  Einteilung von Daten in bestimmte Klassen.  Anhand vorgehender Beobachtungen vorhersagen neuer Instanzen.  Die Bezeichnungen dieser Klassen sind eindeutige, ungeordnete Werte, die als Gruppenzugehörigkeit der Instanzen aufgefasst werden können.  Binäre Klassifierung: Spam oder Nicht-Spam  Beispiel: Negative Klasse und positive Klasse  Mehrfachklassifizierung: Beispiel Handschrifterkennung  **Beispiel:** Filtern von E-Mail-Spam  Trainingsdaten: Mails, die entweder als Spam oder Nicht-Spam markiert sind. Die Maschine kann dann sagen, ob ein neues Mail Spam ist oder nicht.  **Typ 2: Regression / Regressionsanalyse**  Ausgabewerte sind im Gegensatz zur Klassifizierung stetig?  Vorgegeben: Regressor-Variablen (unabhängige oder erklärende Variablen) und eine stetige Zielvariable (Ergebnis)  Es wird versucht, bei diesen Variablen eine Beziehung festzustellen.  Wie lineare Regression in Statistik.  Mit dem ermittelten Schnittpunkt der y-Achse und der Steigung der Gerade kann man das Ergebnis für neue Werte vorhersagen. | **Verstärkendes Lernen Interaktive Aufgaben**    Entwickeln eines Systems (den Agenten) das seine Leistung durch Interaktion mit seiner Umgebung verbessert.  Zu den Informationen über den Zustand der Umgebung gehört ein Belohnungssignal. Das ist ein Mass, wie gut eine Aktion war. Wird durch eine Belohnungsfunktion beschrieben.  Beispiel: Schachcomputer, MarI/O  **Unüberwachtes Lernen Verborgene Strukturen**  Umgang mit nicht gekennzeichneten Daten oder Daten mit unbekannter Struktur.  Sinnvolle Informationen über die Daten lernen, ohne dass es Hinweise auf eine Zielvariable oder eine Belohnungsfunktion gibt.  **Aufspüren von Untergruppen durch Clustering**  Exploratives Datenanalyseverfahren. Man teilt Informationen in sinnvolle Untergruppen (Cluster) auf, ohne vorherige Kenntnisse über die Gruppenzugehörigkeit dieser Informationen zu besitzen.  Jeder Cluster hat dabei eine bestimmte Eigenschaft, unterscheidet sich aber hinreichend von anderen Clustern. Darum wird das Clustering auch als unüberwachte Klassifizierung bezeichnet.  **Datenkomprimierung durch Dimensionsreduktion**  Daten haben oft eine hohe Dimensionalität = Jede Beobachtung besteht aus einer Vielzahl von Messwerten.  Bei der Vorverarbeitung von Merkmalen wird häufig eine unüberwachte Dimensionsreduktion eingesetzt, um die Daten von sogenanntem Rauschen zu befreien.  Die Daten werden in kleine Unterräume geringerer Dimensionalität aufgeteilt, wobei der Grossteil der relevanten Informationen erhalten bleibt.  Die kann zu einer Abschwächung der Aussagekraft bestimmter Vorhersagealgorithmen führen. |

|  |  |
| --- | --- |
| **Grundlegende Terminologie und Notation**  Grundsätzlich gleiche Notation wie in lineare Algebra (Matrizen- und Vektornotation)    **Zeile = 1 Objekt**  Hat man etwa eine Blumen-Datensammlung wird jedes Blumenexemplar durch eine Zeile repräsentiert.  Ein Objekt wie eine Blume wird dabei auch als Instanz oder Beobachtung bezeichnet.  **Spalte = 1 Merkmal eines Objekts**  In den einzelnen Spalten stehen die in Zentimeter angegebenen Messdaten = Merkmale der Datenmenge  Ein Merkmal **x** wird auch als Attribut, Messwert oder Dimension bezeichnet.  **Spezielle Spalte: Die Klassenbezeichnung**  Bei der Blume „Iris“ wird jeweils in einer Klassenbezeichungs-spalte angegeben, ob es sich um eine „Iris setosa“, „Iris versicolor“ oder „Iris virginica“ handelt.  Die Klassenbezeichnungsspalte ist hier die Zielvariable bzw. das Ergebnis !  **Matrizen und Vektoren**  Hat man also eine Datensammlung aus 150 Datensätzen mit jeweils vier Merkmalen kann somit eine 150 \* 4 Matrix X geschrieben werden:  Hochgestelltes (i) = i-te Trainingsobjekt  tiefgestelltes j = j-te Dimension bzw. j-tes Merkmal | **Epoche**  = Maximale Anzahl von Durchläufen der Trainingsdaten  **Entwicklung eines Systems für Machine Learning**    **Vorverarbeitung**  Entnahme von sinnvollen Merkmalen aus den Rohdaten. Die Merkmale müssen für die meisten Lernalgorithmen normiert sein (gleiche Einheit, gleicher Massstab)  Oftmals Intervall [0,1] oder Standardnormalverteilung (Mittelwert 0 und Standardabweichung 1)  Verfahren zur Dimensionsreduktion, um Merkmalsraum zu verkleinern. Braucht weniger Speicherplatz und Algorithmus ist schneller.  Aufteilung der Daten in Trainingsdaten und Testdaten. Die Testdaten sind für eine Überprüfung des Modells am Ende.  Trainingsdaten können weiter unterteilt werden in Trainings- und Validierungsteilmenge, um die Verallgemeinerungsfähigkeit des Modells abzuschätzen.  **Trainieren und Auswählen eines Vorhersagemodells**  In der Praxis sollte man eine Handvoll Algorithmen vergleichen, um das am besten funktionierende Modell zu trainieren und auszuwählen.  Ein Modell wird oft bewertet nach der Korrektklassifizierungsrate oder Vertrauenswahrscheinlichkeit des Modells.  Um Algorithmen der Standardbibliotheken zu verbessern, wird das Verfahren zur Hyperparameter-Optimierung gemacht.  **Bewertung von Modellen und Vorhersage**  Wurde ein Modell für gut empfunden, können die Parameter verwendet werden, um anhand neuer, zukünftiger Daten vorhersagen zu treffen. |

|  |  |
| --- | --- |
| **Machine Learning mit Python**  Entweder Python von [www.python.org](http://www.python.org) installieren oder gleich das Gesamtpaket Anaconda auf  <http://continuum.io/>  **Verwendete Python Bibliotheken**  NumPy 1.9.1: Mehrdimensionale Arrays für Datenverarbeitung  pandas 0.15.2: Funktionen für die Verarbeitung von Tabellendaten  matplotlib 1.4.0: Visualisierung quantitativer Daten  SciPy  scikit-learn  **Perzeptron Lernalgorithmus**  **Neuronen**  Bei Neuronen handelt es sich um miteinander verknüpfte Nervenzellen des Gehirns, die an der Verarbeitung und Weiterleitung chemischer und elektrischer Signale beteiligt sind.  Wenn ein bestimmter Schwellwert erreicht ist, erzeugt die Zelle ein Ausgabesignal.  Eine Nervenzelle wird dabei angesehen wie ein einfaches logisches Gatter mit binärer Ausgabe.  **Perzeptron-Lernregel**  Das Ziel ist es, herauszufinden, zu welcher von zwei Klassen ein Objekt gehört = binäre oder dichotome Klassifizierung  1 = Positive Klasse  -1 = Negative Klasse  Daraus wird eine Aktivierungsfunktion (phi) gebildet mit z als Nettoeingabe  x = Eingabewerte als Vektor  w = Gewichtungsfaktoren als Vektor | **Die Aktivierungsfunktion z beim Perzeptron**  Beim Perzeptron ist die Aktivierungsfunktion eine einfache Sprungfunktion oder Heaviside-Funktion:  (Theta) = Schwellwert  Der Einfachheit halber wird eine Nullgewichtung mit  somit ist z das Skalarprodukt der zwei Vektoren  **Repetition: Matrix transponieren**  **Schritte des ursprünglichen Perzeptron Modelsl**  Die Idee besteht darin, einen reduzierten Ansatz zu verwenden, um zu simulieren, wie ein einzelnes Neuron im Gehirn funktioniert: Entweder es feuert oder es feuert nicht.  1. Die Gewichtungen werden mit 0 oder kleinen zufälligen Werten initialisiert  2. Bei jedem zum Training eingesetzten Objekt :  a. Berechnung des Ausgabewertes  Gemäss vorher definierter Sprungfunktion  b. Aktualisierung der Gewichtungen  Das Delta kann dabei mit der Perzeptron-Lernregel berechnen:  = Eta = Lernrate, Konstante zwischen 0 und 1  = Tatsächliche Klassenbezeichnung  = vorhergesagte Klassenbezeichnung  x = aktuelles Merkmal des Objekts  Alle Gewichtungen im Vektor werden gleichzeitig aktualisiert. |

|  |  |
| --- | --- |
| **Beispiele zur Perzeptron-Regel**  ist die effektive Klasse, die vorhergesagte.  Wenn das Perzeptron die Klasse korrekt vorhersagt, kommt am Ende als Gewichtsupdate 0 heraus:  oder  Nur wenn sich die Vorhersage von der tatsächlichen Klasse unterscheidet, gibt es ein Delta Gewicht w:  oder  **Der Faktor**  Als Beispiel nehmen wir folgende Werte:  Das Objekt wäre also eine 1, wir denken aber es sei eine -1.  Somit ergibt sich ein Delta der Gewichtung von  Die Gewichtung wird also um 1 erhöht. Dadurch wird auch die Nettoeingabe erhöht, und somit die Wahrscheinlichkeit, dass der Schwellwert der Sprungfunktion überschritten wird und somit das Objekt als 1 klassifiziert wird.  Bei einem weiteren Objekt mit , welches ebenfalls fälschlicherweise als klassifiziert wird, ergibt das Delta der Gewichtung:  Durch Gewicht wird das Objekt noch mehr Richtung „1“ geschoben.  Je grösser desto grösser das Delta Gewicht desto mehr wird das Objekt in die korrekte Richtung geschoben. | **Perzeptron muss linear trennbar sein!**  Die Konvergenz ist nur gegeben, wenn die Klassen lineare trennbar sind. Gäbe es keine lineare Entscheidungsgrenze, würde das Perzeptron endlos weiteraktualisieren.    **Konzept des Perzeptron grafisch**    Die Eingabe x wird entgegengenommen und mit der Gewichtung w kombiniert um die Nettoeingabefunktion zu berechnen. Diese wird dann der Aktivierungsfunktion übergeben, um eine binäre Ausgabe zu erhalten (1 oder -1), was der Vorhersage der Klassenbezeichnung des Objekts entspricht.  Während der Lernphase wird diese Ausgabe genutzt, um Fehler festzustellen und die Gewichtungen zu aktualisieren. |

|  |  |
| --- | --- |
| **Perzeptron-Lernalgorithmus in Python**  **Variablen**  Unterstrich am Ende bedeutet, dass das Attribut nicht bei der Initialisierung des Objekts gesetzt wurde  **Methoden**  fit – Lernt aus Daten  predict – Trifft eine Vorhersage  **Bibliothek NumPy**  sdfdddd  **Bibliothek pandas**  asdf  **Bibliothek matplotlib**  asdf |  |

|  |  |
| --- | --- |
| **Numpy Bibliothek**  **Array mit zehn Nullen erstellen**  w = numpy.zeros(10)  Achtung: Zeros nicht Zeroes! |  |

|  |  |
| --- | --- |
| **Sklearn Bibliothek**  **Decision Tree**  from sklearn import tree |  |