Patrick Malik
5BHIT

Data Mining-Ausarbeitung

Contents

Allgemein

Data Mining ist eine Vorgehensweise bei der verschiedene gesammelte Daten über Personen, in den meisten Fällen Kunden, analysiert werden. In diesem Vorgang wird versucht Muster zu erkennen um Aussagen für bzw. über die Zukunft treffen zu können. Data Mining verwendet verschiedene darauf optimierte Algorithmen der Satistik und Techniken der künstlichen Intelligenz um Analysen zu treffen. Bei den Unmengen an Daten mit denen Firmen heutzutage Arbeiten, ist es unmöglich für Menschen diese zu analysieren und Data Mining Systeme helfen beim Entschlüsseln der Bedeutung dieser Daten. [1]

Domain Knowledge

Bei Domain Knowledge handelt es sich um Information die notwendig ist um Daten sinnvoll zu interpretieren. Sollte sich bspw. eine Lücke in den Daten bilden, kann dieses Wissen helfen zu verstehen, dass das System aufgrund von Wartung zu dieser Zeit ausgefallen ist. Dieses Wissen kann nicht durch das System erkannt werden, außer es handelt sich natürlich um eine Regelmäßigkeit. Prinzipiell handelt es sich bei Domain Knowledge um Meta-Daten, welche bei der Interpretation der Daten helfen.[1]

Sampling

Je nach Menge an Daten arbeitet man entweder mit allen zur Vergügung gestellten Daten oder bloß einem Sample. Hierfür ist es notwendig und bedeutend, dass es sich um sinnvolle und gute Samples handelt, da die Ergebnisse für das Sample als Ergebnis für die ganze Menge an Daten, auch Population genannt, gilt.[1]

Außerdem ist es bei Data Mining wichtig, lediglich Daten in die Analyse zu beziehen, die relevant sind. Irrelevante Daten können zu Missinterpretationen führen und die Ergebnisse verfälschen. [1]

Data Warehousing

Bei Data Warehouses handelt es sich um Datenbanksysteme, die Daten sammeln, welche für bspw. Data Mining Prozesse verwendet werden können. DWH sind nicht zu jedem Zeitpunkt auf dem neuesten Stand sondern werden periodisch upgedated und sind auf viele jedoch beinahe ausschließlich Lesezugriffe optimiert. Außerdem greifen nur wenige User auf dieses Sytem zu.[3]

Bei DWH ist es wichtig, dass erkennbar ist, woher bzw. von wann diese Daten stammen. Abfragen liefern eine große Menge an Daten zurück.[3]

Die Irrelevanz der Verfügbarkeit und Performance, machen diese Systeme enorm effizient auf ihrem Gebiet und bieten somit gute Leistungen, insbesondere bei großen Anfragen mit großen Antwortmengen. [3]

Der Vorteil der DWH-Systeme verglichen mit OLTP-Systemen ist, dass hier mit koplexen Abfragen und enormen Datenmengen gearbeitet werden kann. Es handelt sich zudem um zeitliche Konsistente Daten und die zentrale Sammlung der Daten verschiedener Applikationen bietet einen guten Ausgangspunkt für DM-Prozesse. [3]

Beschreibung der Daten

Um die gesammelten Daten weiter verwenden zu können, müssen sie zunächst aufbereitet werden. Es ist notwendig, dass die zugehörigen Attribute richtig beschrieben werden. [5]

Jede Branche sammelt Daten über ihre Kunden. Daten die für die Branche von Bedeutung sind. Diese Daten werden in Datenbanken abgelegt und jede Reihe entspricht einem Datensatz. Dieser Datensatz besteht aus den verschiedenen Beobachtungen zu den Eintragungen (häufig Kunden oder Produkte), welche spezielle Informationen preis geben wie bspw. das Gewicht oder Alter eines Kunden. Diese Daten werden üblicherweise als raw bezeichnet. [5]

Zunächst ist es notwendig zu bestimmen, welcher Art die Daten entsprechen. Wenn es sich um einen numerischen Wert handelt bspw. ob es sich um einige wenige Nummern, oder auch Klassen handelt oder das Attribut jeden Wert annehmen kann (discrete oder continous). Eine Jobbranche wäre ein gutes Beispiel für eine diskrete Variable, hier gibt es vorgegebene Möglichkeiten. Das Gewicht eines Patienten bspw. ist continous, es kann jeden beliebigen numerischen Wert annehmen. Außerdem ist es wichtig, die Skala zu kennen auf der sich die Variable bewegt. Diese ist notwendig um ein sinnvolles Tool zu wählen oder bei der Visualisierung zu wissen wie die Daten darzustellen sind. Hierzu gibt es verschiedene Einteilungen: [5]

- Nominal scale
  Hier sind die Möglichkeiten beschränkt. Ein Industriesektor oder eine Jobbranche zum Beispiel
- Ordinal scale
  diese beschreibt Variablen mit einer ebenfalls festgelegten Anzahl an Möglichkeite, jedoch sind diese geordnet. Bspw. High, Medium und Low. Hier gibt es lediglich 3 Optionen, es ist jedoch bekannt, das High höher als Medium, Medium höher als Low und Low die niedrigste Kategorie ist. Es ist jedoch nicht möglich den Unterschied zwischen den Kategorien zu verlgeichen.
- Interval scale
  bei dieser Skalierung ist es möglich die Unterschiede zwischen den Kategorien zu vergleichen. 5°F, 10°F und 15°F haben jeweils 5°F unterschied, jedoch sind die Werte dieser Kategorien nicht zu vergleichen. 10°F ist nicht doppelt so warm wie 5°F.
- Ratio scale
  Diese Skalierung lässt sowohl die Werte als auch die Intervalle vergleichen. Bspw. sind zwischen 5€, 10€ und 15€ jeweils 5€ Abstand aber 10€ sind auch doppelt so viel wie 5€. Diese Skala lässt Intervall als auch Wert sinnvoll vergleichen.
- Dichotomous (Binär)

Zudem gibt es jedoch auch Attribute die lediglich für die Interpretation oder Beschreibung der Ergebnisse von Bedeutung sein können. IDs o.ä. sollten nicht als Variable einbezogen werden, sie geben jedoch aufschluss darüber welcher Kunde bspw. hier zugehörig ist. Auch Daten die bspw. Information zur Kalibrierung einer Maschine geben sind wichtig, werden allerdings nicht in den Prozess miteinbezogen. Diese Information entscheidet möglicherweise darüber, ob die Daten einer anderen Maschine inkludiert werden können. Bei anderer Kalibrierung muss diese entweder berücksichtigt oder die ganzen Daten außen vor gelassen werden. [5]

Variablen können auf verschiedene Arten beschrieben werden. Häufig ist es interessant oder notwendig bspw. die Mitte zu berechnen, um die der Großteil der Werte liegt. Hierzu gibt es 3 übliche Methoden der Statistik: Median, Durchschnitt und Modus. [5]

> *Der Modus nimmt die am häufigsten auftretende Zahl einer Reihe. In der Folge 3,3,4,5,7,7,7,8,9 ist der Wert 7 der am häufigsten auftretende. Dieser würde nun als Wert verwendet werden. Sollten 2 Werte die gleiche Anzahl an Auftritten haben, wird ein Mittelwert gebildet (7 und 8 bspw 7.5)*

Darzustellen wie die Werte dieses Attributs verteilt sind, bspw deren Frequenz (Histogramm) visualisieren ist eine weitere Möglichkeit Informationen über Daten zu gewinnen. [5]

Weitere statistische Werkzeuge wie die Spannweite, Quartile, Varianz und Standard Abweichung geben zusätzliche Information. Viele dieser Daten können bspw. mittels eines Boxplots dargestellt werden.
Neben diesen Mitteln kann es auch von Bedeutung sein die Form der Verteilung zu kennen. Informationen über die Verteilung der Werte um den Mittelwert können von Bedeutung sein. [5]

Vorbereitung der Daten

Der folgende Punkt ist bei weitem der aufwändigste. Um die gesammelten und mittlerweile beschriebenen Daten auch sinnvoll verarbeiten zu können, müssen sie nun gesäubert werden, indem redundante, irrelevante und problematische Daten entfernt und Fehler behoben werden. Möglicherweise ist es auch nowendig neue Attribute zu berechnen. [5]

Zudem kann es sinnvoll sein diese Daten in Subsets zu unterteilen, um schnellere und leichtere Abfragen zu ermöglichen. Es ist auch immer wichtig, alle Schritte zu dokumentieren. Die Dokumentation bietet so nicht nur Nachvollziehbarkeit, sondern auch eine Methode, wie sie bei kommenden Projekten verwendet werden kann.[5]

Säubern der Daten

Variablen die in das Gebiet der nominal oder ordinal scale fallen, sollten insbesondere auf deren Möglichkeiten überprüft werden. Es sollten alle Optionen auf Redundanzen überprüft werden, um bspw. gleiche Firmen mit verschiedenen Anschriften („General Electric Company“, „General Elec. Co.“) herauszufiltern. Bei numerischen Werten kann es vorkommen, dass nicht numerische Werte wie NaN o.ä. auftauchen. Auch diese müssen entfernt oder durch Nummern ersetzt werden.[5]

Bei fehlenden Daten ist Expertise notwendig. Diese Daten können unter Anderem durch Fehler fehlen, jedoch auch Bedeutung haben. Auch diese müssen entweder ersetzt oder entfernt werden. Daten des Typs Intervall oder Ratio können fordernder sein. Bei diesen Daten kann es durchaus zu Ausreißern kommen. Diese werden am Besten durch Histogramme oder ähnliche Visualisierungen erkannt und entstehen häufig durch Messfehler. Diese Problemfälle sollten entfernt werden. [5]

Bei der Zusammenführung verschiedener Datensammlungen kann es unter Anderem zu Diskrepanzen bei den Einheiten geben. Meilen statt Kilometer bspw. werden auf Anhieb nicht erkannt, führen aber zu großen Unterschieden. Diese müssen bei der Überprüfung erkannt und standardisiert werden. [5]

Aufzeichnungen denen eine bedeutende Variable fehlt sollten in den meisten Fällen entfernt werden. Sollte ein Attribut auffällig viele fehlende oder fehlerhafte Werte haben, sollte auch hier die Entfernung in Betracht gezogen werden. [5]

Normalisierung

Viele Daten unterscheiden sich im Bereich in dem sie liegen können. Das Alter ist häufig wesentlich kleiner als die getätigten Ausgaben in € die einem Kunden zugeordnet sind. Jedoch ist es notwendig, diesen Daten nicht mehr Einfluss zu geben, als anderen. Solche Attribute sollten Normalisiert werden. Eine einfache Art der Normalisierung ist die Min-Max-Normalisierung, hier werden die Werte eines Attributs auf einen Wert zwischen 0 und 1 gebracht. Die Formel dafür sieht aus wie folgt: [5]

$$x'_i = \frac{x_i - OriginalMin}{OriginalMax - OriginalMin} * (NewMax - NewMin) + NewMin$$

Wobei für x'_i der neue Wert zwischen NewMax und NewMin liegt, OriginalMin der kleinste Wert der ursprünglichen Daten ist, OriginalMax der größte und NewMax und NewMin die 2 Werte sind, in unserem Fall 1 und 0, zwischen denen der neue Wert liegen soll. [5]

Text zu Nummern

Variablen der Skala nominal oder ordinal können auf verschiedene Arten zu Nummern konvertiert werden. Es ist möglich, jedem nominal Wert (Rot, Grün, Blau) eine Zahl zuzuordnen (0, 1, 2). Außerdem kann man für jede Option eine eigene Spalte erzeugen, wobei der Wert dieser Attribute binär ist und bei einem roten Produkt ist lediglich der Wert der Spalte Rot 1, die 2 anderen Spalten 0. [5]

Kontinuierliche Daten zu diskreten Werten

Bspw. ist es für manche Modelle notwendig, dass Kategorien und keine Zahlen angegeben werden. Hier erfolgt die Einteilung der Kategorien wie die Klasseneinteilung in der Statistik. Möglicherweise kann man jedoch auch selbst Kategorien festlegen, wenn diese einem gewissen Prinzip folgen, insbesondere wenn sie von einer Verteilung suggeriert werden. Dieser Schritt kann auch bei nominalen Werten durchgeführt werden. Sind bspw. zuviele Kategorien vorhanden oder zuviele Werte in einer Kategorie, können diese sinnvollerweise zu anderen diskreten Werten umgewandelt werden. [5]

Kombinieren von Variablen

Wie bereits erwähnt kann es auch notwendig sein, neue Attribute zu berechnen. Dieser eigenartig klingende Schritt kann Variablen entweder zusammenfassen und somit sowohl Platz als auch Rechenleistung sparen oder auch bessere Übersicht geben. Die Treibstoffeffizienz kann aus (TreibstoffstandEnde – TreibstoffstandBeginn)/Distanz berechnet werden. Würde man diese 3 Spalten als irrelevant betrachten, könnte man sie auf eine Spalte vereinigen oder lediglich in dieser zusätzlichen Spalte mehr Information finden. [5]

Erzeugen von Samples

Das Generieren von einzelnen Gruppen aus den gegebenen Daten kann aus verschiedenen Gründen zielführend sein. Das Verwenden vieler Daten führt nicht immmer zu einem enorm gesteigerten Ergebnis, kann jedoch die Rechenleistung außerordentlich erhöhen. Somit können zufällig ausgewählte Datensätze verwendet werden, um die Analyse auszuführen. Dies funktioniert natrülich nur, wenn das Zielset dem ursprünglichen ähnlich sein soll. [5]

Ein Sample kann jedoch auch dazu dienen, einer Anwendung zu entsprechen. Sammelt man bspw. seit Jahren alle Daten über seine Kunden und benötigt eine spezielle Abfrage die nur auf eine spezielle Gruppe Kunden zutrifft, kann man diese Kunden als Sample herausfiltern. Diese Art kann jedoch auch in anderen wirtschaftlichen Fällen Anwendung finden. Es macht z.B. Sinn, Gruppen für spezielle Umstände zu bilden. Die Preise für Immobilien die in Küstennähe liegen werden anders auf spezielle Umstände ansprechen als Immobilien in Bergregionen. Hier für solche Unterschied Subsets zu bilden ist durchaus stringent. [5]

Knowledge Discovery Process

Beim KDP handelt es sich um eine dem Data Mining übergeordnete Stufe, welcher in mehrere Schritte aufgeteilt ist. Hier gibt es sowohl verschiedene Ansätze als auch Modelle, welche auf verschiedene Einsatzgebiete optimiert sind. Als Input werden jegliche Art von Daten verwendet (Bilder, Videos, Zahlen, usw.) und der Output ist das neu erlangte Wissen in Form von Regeln, Mustern, Trends, Statistiken usw.. [2]

Forschungsmodelle

Das KDP-Modell nach Fayyad et al. Erfolgt in den folgenden 9 Schritten: [2]

1. Entwickeln und verstehen der Application Domain (Finden der Domain Knowledge, der zu verwendenden Daten und der Ziele)
2. Erstellen eines target data sets (mit dem discovery tasks ausgeführt werden können)
3. Säubern und Vorbereiten der Daten (Irrelevante Daten entfernen, Fehlende Daten ergänzen bzw. erklären,…)
4. Reduzierung und Projektion (Sinnvolle Attribute und unmissverständliche Darstellung der Daten finden)
5. Wählen eines DM-Tasks (passend zu dem in Schritt 1 erkannten Ziel eine DM-Methode finden: Klassifizierung, Clustering, Regression,…)
6. Wählen eines DM-Algorithmus
7. Data mining
8. Interpretieren der gefundenen Muster (Visualisierung)
9. Konsolidieren der gefundenen Ergebnisse (Reporting, Einarbeiten in das System, …)

Hierzu ist jedoch zu sagen, dass Wiederholungen erlaubt sind. Bei diesem Modell handelt es sich um eines, das als Grundlage für viele folgende verwendet wurde. [2]

Industrial Models

Bei dem folgenden Modell handelt es sich um das CRISP-DM-Modell, welches eines der industrial models ist: [2]

1. Verstehen der Ziele und Anforderungen aus geschäftlicher Sicht
    a. Festlegen der geschäftlichen Ziele
    b. Beurteilung der Situation
    c. Festlegen der DM-Ziele
    d. Generieren eines Projekt-Plans
2. Verstehen der Daten
    a. Sammeln der Initialdaten
    b. Beschreibung dieser
    c. Erforschung der Daten (Erkennen interessanter Subsets,…)
    d. Verifikation der Qualität der gesammelten Daten
3. Vorbereitung der Daten
    a. Selektion der Daten
    b. Säuberung der Daten
    c. Konstruktion neuer Attribute
    d. Integration der Daten
    e. Mögliches Anpassen und wiederholen dieser Schritte
4. Modeling
    a. Selektion der Modeling Techniken
    b. Generierung eines Test designs
    c. Erzeugung von Models
    d. Beurteilung der verwendeten Modelle
5. Evaluation
    a. Evaluierung der Ergebnisse
    b. Prozess Review
    c. Festlegung des nächsten Schritts
6. Deployment
    a. Planen des Deployments
    b. Überwachung und Wartung
    c. Generierung des endgültigen Reports

d. Review dieser Schritte

Insbesondere die Wiederholung der Schritte 3 und 4 ist durchaus erlaubt und erwünscht, bis brauchbare Modelle erkannt wurden. Dieses Modell fand bereits Anwendung in der Medizin, im technischen Bereich als auch zu Marketing- und Verkaufszwecken.[2]

Hybriden

Das folgende Modell nach Cios et al. Wurde anhand des CRISP-DM Modells entwickelt und an akademische Modelle (siehe Forschungsmodelle) angepasst: [2]

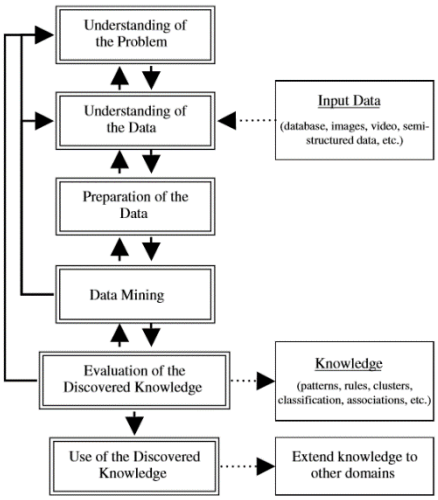

Abbildung 1: Cios et al. Prozess [2]

1. Verstehen der problem domain
    a. In diesem Schritt sollen die Ziele, Schlüsselpersonen und derzeitige Lösungen erkannt werden. Projektziele sollen zu DM-Zielen werden und eine prinzipielle Auswahl an DM-Tools für die spätere Verwendung soll getroffen werden.
2. Verstehen der Daten
    a. Hier werden Sampledaten gesammelt und es wird entschieden welche Daten tatsächlich von Nutzen sind und verwendet werden. Daten werden auf ihre Plausibilität und Vollständigkeit geprüft. Zuletzt wird überprüft, ob die DM-Ziele mit dieser Auswahl an Date erfüllbar ist.
3. Vorbereitung der Daten
    a. Zunächst werden Inputdaten für die DM-Methoden ausgewählt, die Daten werden auf ihre Bedeutung (in Zusammenhang mit dem zu erreichenden Ziel) geprüft und gesäubert. Des Weiteren können verschiedene Selektions bzw. Extraktionsverfahren in diesem Schritt angewandt werden.
4. Data mining
5. Evaluation der Ergebnisse
    a. Handelt es sich um interessante und sinnvolle Ergebnisse
    b. Welche Schritte könnten rückblickend verändert werden, um die Ergebnisse zu verbessern
    c. Zusammenstellen einer Liste von Fehlern die in diesem Prozess stattgefunden haben
6. Verwendung des neuen Wissens
    a. Planen wie das neue Wissen eingesetzt werden kann
    b. Aufzeichnen der Schritte und deren Auswirkungen die durch das neue Wissen erfolgen
    c. Dokumentation des gesamten Projektes fertigstellen

Auffällig ist bei diesem Modell die hohe Anzahl an Schleifen zwischen den einzelnen Schritten. Diese sorgen für ausgefeiltere Ergebnisse bzw. sinnvollere Ergebnisse bei unzufriedenstellenden Lösungen. Hybridmodelle werden insbesondere in der Medizin und auf dem Softwaregebiet eingesetzt, für bspw. Analysen oder Klassifikationen von Zellen o.ä.. [2]

Folgendes Diagramm zeigt zwar nicht die vorgestellten Methoden, jedoch wird offensichtlich dargestellt, in welche der Schritte die meiste Zeit fließt:

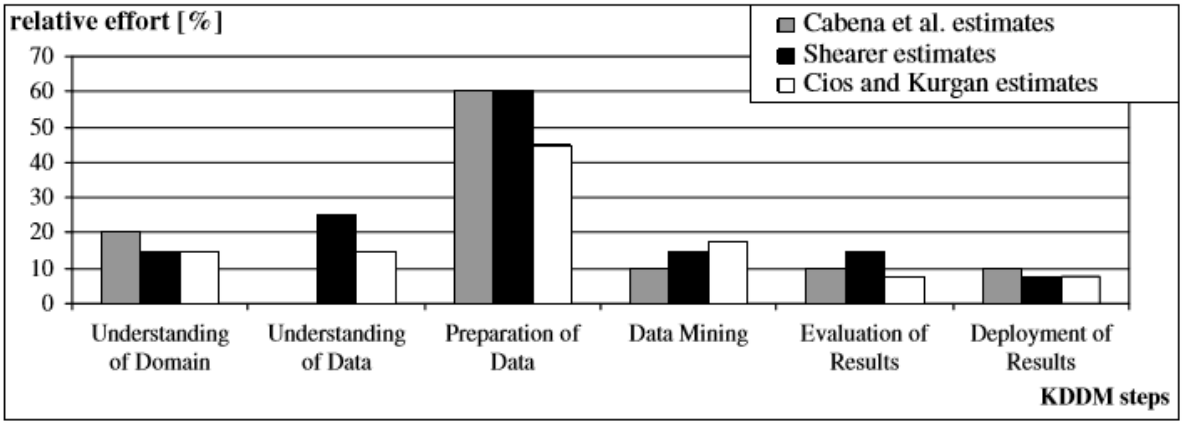

Abbildung 2: Relativer Aufwand pro Schritt [2]

Modelle

Prinzipiell können Modelle in 2 Typen eingeteilt werden: Modelle die kontinuierliche Ergebnisse erzeugen und Modelle die diskrete Ergebnisse erzeugen. Ein kontinuierliches Ergebnis entspricht einer Variable die jeden numerischen Wert annehmen kann. Ein diskretes Ergebnis ist eine Kategorisierung - es gibt eine festgelegte Anzahl an Möglichkeiten die das Ergebnis sein kann. [5]

Genauigkeit

Für jedes Modell ist es notwendig, dass es verlässlich ist und eine gewisse Genauigkeit hat. Hat ein Modell eine schlechte Genaugikeit bzw. Zuverlässigkeit, ist es nicht sinnvoll dieses in den Einsatz zu bringen. Für kontinuierliche Modelle gibt es bspw. das k-fold partitioning. Hier wird das ursprüngliche Datenset in k gleichgroße Partitionen unterteilt und wird dadurch k-Mal gemessen. Zu Beginn wird eine der Partitionen als Testset bestimmt, die übrigen k-1 Partitionen werden als Trainingssets verwendet. Nachdem die Trainingssets abgearbeitet sind, wird das Testset verwendet und die Ungenauigkeit gemessen. Bei der nächsten Iteration wird das 2. Set als Testset verwendet und die übrigen k-1 Partitionen als Trainingsset, etc.. Nach jeder Iteration wird die Abweichung mithilfe des Testsets gemessen und nach der letzten gemittelt, um auf die Ungenauigkeit des Modells zu kommen. Üblicherweise handelt es sich hierbei um ein 10-fold partitioning, jedoch soll das keine Einschränkung darstellen. Der Vorteil dieses Modells ist die Tatsache, dass jegliche Partition sowohl als Test als auch Trainingset verwendet wurde und somit mit allen Daten getestet wurde, somit entspricht diese Ungenauigkeit tatsächlich der aller Aufzeichnungen der gegebenen Daten.[5]

Bei diskreten Modellen kann die Genauigkeit in % Berechnet werden, indem die Anzahl an Fehlschlägen durch die Versuche dividiert wird. Siehe Kapitel Neural networks. [5]

Bei binären Modellen (zählen zu diskreten) gibt es zusätzlich 2 übliche Berechnungen zur Genauigkeit des Modells: sensitivity und specificity: [5]

$$Sensitivity = \frac{TP}{(TP + FN)}$$

$$Specificity = \frac{TN}{(TN + FP)}$$

Sensitivity beschreibt wie gut ein Modell positive Ergebnisse vorhersagt, wobei specificity das Gegenteil erklärt. [5]

Um die Performance eines Modells zu beschreiben, kann man diese mittels Scatterplot leicht visualisieren. Die berechneten Ergebnisse gegenüber den Vorhergesagten Ergebnissen ergeben bei guten Ergebnissen eine gerade Linie, wobei sie bei schlechten Modellen verstreut sind: [5]

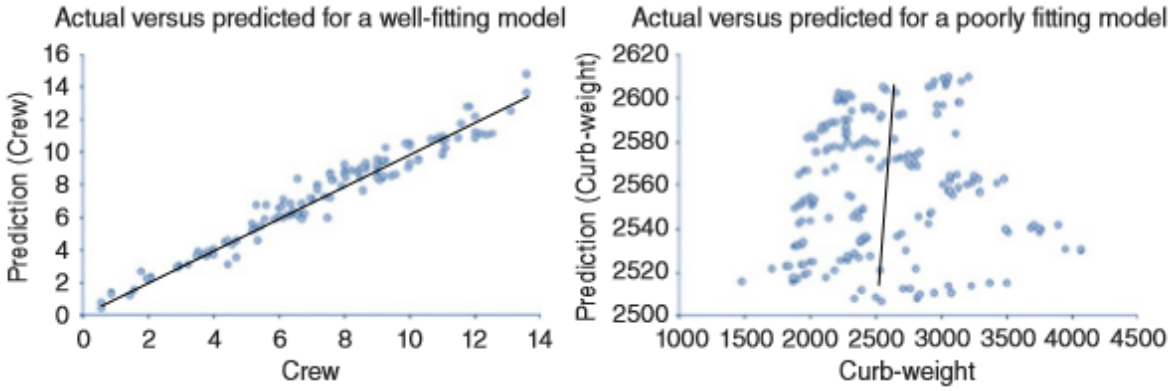

Abbildung 3: Vergleich Scatterplots [5]

Hat der Plot eine nicht-lineare Form, ist zu erkennen, dass entweder das Modell auf ein nicht-lineares geändert werden muss oder nicht-lineare Variablen noch nicht transformiert wurden. [5]

Um Modelle zu vergleichen, kann man die Abweichungen (Residuals) von den tatsächlichen Ergebnissen berechnen, quadrieren (um negative positiv zu machen) und summieren. [5]

Supervised und Unsupervised Methods

Bei supervised Methoden werden in der Testphase Testdaten an das System gefüttert und das Ergebnis wird mit dem eigentlichen verglichen. Hierfür ist es notwendig, sowohl Inputdaten als auch Outputdaten für einen gewissen Zeitraum zu haben. Diese Methode wird bspw. verwendet, um zu überprüfen, ob Kunden auf eine Werbekampagne reagieren oder nicht. Diese Methoden können als Feintuning gesehen werden. [1]

Bei unsupervised Methoden gibt es zwar auch einen Testzeitraum, die Ergebnisse werden jedoch nicht überprüft. Diese Methoden werden verwendet, um natürliche Strukturen zu erkennen. [1]

Multiple lineare Regression bei stetigem Target

Probleme bei Ansätzen der linearen Regression sind häufig Annahmen die zu treffen sind. Prinzipiell wird angenommen, dass keine Messfehler in den Daten vorliegen, also dass die Daten alle korrekt sind. Zudem wird erwartet, dass die Zusammenhänge aller Daten von linearer Natur sind. Aufzeichnungen seien unabhängig und predictor-Variablen sind keine lineare Kombination aus anderen. Außerdem wird angenommen, dass die Varianz der Target-Variable konstant ist. [4]

Lineare Regression nähert sich mithilfe von Punkten in einem 2 dimensionalen Raum an eine Gerade an. Es wird eine Gerade so durch die Punkte gelegt, dass der Abstand auf der Y-Achse im Quadrat am kleinsten ist. Diese Methode reicht jedoch häufig nicht aus, um komplexe Dinge zu beschreiben, da man lediglich eine Linie erzeugt und das Ergebnis möglicherweise kurvig oder weitaus komplexer ist. Hierzu wird multiple lineare Regression verwendet. [1]

Bei diesem Verfahren werden verschiedene sog. Predictor-Variablen zu Hilfe genommen, um eine Target-Variable Y zu errechnen. Diese Predictor-Variablen bewegen sich auf der X-Achse und werden als unabhängig bezeichnet, da sie jeden Wert annehmen können. Es ist jedoch wichtig, dass sowohl Target-, als auch Inputvariablen einen numerischen Wert haben. Jegliche nominale Werte müssen zu Zahlen konvertiert werden. Bspw. kann eine Variable zur Beschreibung der Farbe den Wert 0 oder 1 annehmen, um zu bestimmen, ob das Produkt diese Farbe hat oder nicht. Kategorische Variablen wie

„Größe“ können nur als Nummern angegeben werden, wenn der Abstand konstant ist und sie durch einen einzigen Regressionskoeffizienten beschrieben werden können. Normalerweise müssen diese jedoch in Indicatorvariablen konvertiert werden (Wie bei Farben 0 oder 1). [1]

Ein solches Modell kann viele Dimensionen besitzen, die generelle Formel ist jedoch:

$$Y = a + b1 * X1 + b2 * X2 + b3 * X3 + … + bn * Xn + e$$

Wobei Y die Targetvariable, Xn die Inputvariablen (Predictorvariablen) und bn die Regressionskoeffizienten sind. Bei a handelt es sich um eine Konstante (ähnlich dem d bei einer Gerade). [1] Wobei e dem Fehler entspricht, der die nicht beschriebene Variation erklärt. [5]

Diese Methode bezieht viele wichtige Inputvariablen ein und abhängig vom Problem sind diese äußerst unterschiedlich, jedoch sollte auch in Betracht gezogen werden Variablen einzubeziehen die nicht direkt etwas mit dem Problem zu tun haben, jedoch im großen Ganzen eine bedeutende Rolle spielen, da sie eine Übersicht über die derzeitige Umgebung verschaffen können. [1]

Für gewöhnlich wird mit einem ausgewählten Set an unabhängigen Variablen begonnen. Dieses Set wird aus Variablen zusammengestellt, von denen Experten behaupten, dass sie Einfluss haben. Eine Faustregel ist hierbei: 10 Aufzeichnungen pro Variable. [5]

Zunächst kann man berechnen, ob die gegebenen Inputvariablen einen Einfluss auf das Modell haben. Hierzu nimmt man den Regressionskoeffizienten und dividiert ihn durch den Standardfehler der zugehörigen Größe. Ist dieses Ergebnis nun unter dem Signifikanzniveau ($p < 0.05$ z.B.), bedeutet das, dass der Regressionskoeffizient im Modell eine Rolle spielt. Ist dies nicht der Fall, ist diese Inputvariable irrelevant. [1]

Diese Ergebnisse führen zu 2 Modellen: [1]

- Es werden alle, auch irrelevante, Koeffizienten in die Berechnung miteinbzeogen
- Es werden lediglich relevante Koeffizienten einbezogen

Sind viele Inputvaribalen vorhanden, wird für gewöhnlich das 2. Modell verwendet. Jedesmal wenn eine neue Variable hinzugefügt wird, werden erneut alle Variablen auf deren Relevanz überprüft und je nachdem aussortiert. Das Signifikanzlevel ist selbst zu setzen und auch 15% - 20% sind durchaus üblich.

Gibt es mehrere Modelle die ein ähnliches Ergebnis erzeugen, sind die günstigeren bzw. einfacheren zu verwenden. [1]

Um die Qualität des Modells sicherzustellen, gibt es verschiedene visuelle Möglichkeiten. Die Unteschiede zwischen berechnetem und tatsächlichem Ergebnis, werden hierfür gelplotted und auf Muster überprüft: [1]

- Darstellen der Abweichungen nach Fällen
  Das Modell soll in frühen als auch späten Fällen gleichgute Leistungen erbringen, daher sollten die Abweichungen hier keine Trends erkennen lassen und zufällig verteilt sein
- Ein Histogram der Abweichungen sollte eine Normalverteilung ergeben (symmetrisch, Spitze um 0)
- Scatterplots mit der Targetvariable oder einer Inputvariable auf der x-Achse und der Abweichung auf der y-Achse dürfen keine Trends oder Muster erkennen lassen, da das Modell bei jedem Wert gleichgut arbeiten soll.

Sollten hier Muster oder Trends erkannt werden können, besteht Verbesserungspotential. [1]

Logistische Regression

Diese Methode wird häufig verwendet, wenn das Target bspw. 0 oder 1 oder ein kategorisches mit wenigen Kategorien ist.[1]

Anhand eines erklärten Beispiels soll dieser Vorgang veranschaulicht werden. Folgendes Bsp. Stammt aus dem Buch [5] und ist dort mit weiteren Verweisen erklärt. Ich behalte mir vor an manchen Stellen auf das Buch zu verweisen.

Das Beispiel zeigt die Verwendung einer logistischen Regression zur Bestimmung anhand der SB-Levels ob in 0.5km Nähe eine Goldader vorhanden ist oder nicht. Zur Verfügung stand ein Datenset, aus dem die folgenden 5 Einträge stammen: [5]

Log(Sb level)	Gold Deposit Proximity
−0.251811973	0
−0.22184875	0
−0.15490196	1
−0.096910013	0
−0.040958608	0

Abbildung 4: 5 Bsp.-Datensätze [5]

Welches den log des SB-Levels beinhaltet, und ob ein Goldvorkommen vorhaden ist oder nicht. Diese Datensätze wurden zusammengefasst und in 6 log(Sb-Level) Ranges und die Durchschnittliche Vorhandenheit von Gold in diesen Klassen eingeteilt: [5]

Log(Sb Level) Ranges	Mean Gold Deposit Proximity
−1.1 to −0.7	0
−0.7 to −0.3	0.04
−0.3 to 0.1	0.43
0.1 to 0.5	0.89
0.5 to 0.9	0.9
0.9 to 1.3	1

Abbildung 5: Durchschnittliches Gold in Klassen [5]

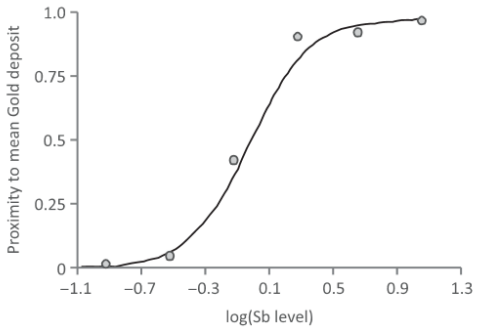

Abbildung 6: Visualisierung der Punkte [5]

Diese Punkte wurden nun geplotted und nach Interpolieren einer Linie ist der S-förmige Graph zu erkennen (Abbildung 6) [5].

Es ist bei diesem Modell von Bedeutung, dass die Werte tatsächlich zwischen 0 und 1 liegen. Bei diesem Beispiel ist das offensichtlich der Fall. Es gibt keine Möglichkeit, dass ein Wert über 1 oder unter 0 liegt. [5]

Mithilfe einer logistischen Formel können nun Punkte entlang der S-Linie berechnet werden:

$$E(Y|x) = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_i x}}$$

Wobei e der eulerschen Zahl entspricht, und β_0 und β_1 konstante Werte sind. Bei diesem Beispiel wurden die Werte mithilfe einer maximum-likelihood-Methode berechnet (siehe [5] S. 164) und man kam auf folgende Gleichung: [5]

$$E(Gold\ deposit\ proximity \mid \log(Sb\ level)) = \frac{e^{-0.0728+5.82*\log(Sb\ level)}}{1+\ e^{-0.0728+5.85*\log(Sb\ level)}}$$

Nun kann man mithilfe dieser Funktion das Sb-Level ersetzen und kommt für das Sb-Level von 0.4 auf das Ergebnis von 0.905, welches nahe 1 liegt und somit vermuten lässt, dass ein Goldvorkommen nahe ist. [5]

Bei diesem Beispiel handelte es sich jedoch um ein simples Regressions Modell mit lediglich einer unabhängigen Variable. Für gewöhnlich ist das nicht der Fall und in Fast jeder Situation verfügt man über mehrere unabhängige Variablen, welche wie folgt in die Formel einfließen können: [5]

$$E(Y|x) = \frac{e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p}}{1+e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\cdots+\beta_p x_p}}$$

Bezieht man nun vorhandene Daten über das Sb- als auch das As-Level ein und berechnet aus gegebenen Daten die Koeffizienten, kommt man zunächst auf die allgemeine Formel: [5]

$$E(Y|x) = \frac{e^{-1.84+5.2\log(As\ level)+3.5\log(Sb\ Level)}}{1+e^{-1.84+5.2\log(As\ \ level)+3.5\log(Sb\ Level)}}$$

Womit man bei einem log(As) von 0.1 und einem log(Sb) von 0.05 auf einen Wert von 0.24 kommt. Da man bei diesem Wert deutlich näher an 0 liegt, ist nicht mit einem nahen Goldvorkommen zu rechnen. [5]

Bei diesem Modell ist es jedoch zusätzlich notwendig, einen Punkt zu fixieren, an dem unterschieden wird. Alle Werte die über diesem Punkt liegen sind somit 1 und alle Werte unter diesem Punkt 0. Diese Variable kann per Hand ausgewählt werden oder automatisch, in dem sie einen Durchschnitt zwischen specificity und sensitivity findet und dementsprechend entscheidet. [5]

Neural networks

Ein neuronales Netzwerk besteht aus 2 Hauptkomponenten, dem Knoten und der Verbindung. Knoten sind in Layer aufgeteilt, wobei der erste Layer Input-Layer, der letzte Output-Layer und die dazwischen Hidden-Layer genannt werden. In den Input-Layer werden alle Daten gespeist und der Output-Layer (häufig bestehend aus 1 Knoten) gibt das Target aus. Die Verbindungen zwischen den einzelnen Neuronen haben Gewichte, welche mit dem Wert des ersten Neurons der Verbindung multipliziert wird und dann als solcher dem zweiten Neuron übergeben wird. [1]

Neuronale Netze werden für gewöhnlich für 2 Arten von Problemen eingesetzt. Entweder zur Bestimmung eines numerischen Wertes, man hat bspw. die Größe, das Gewicht und das Geschlecht gegeben und soll in etwa das Alter der Person bestimmen, in diesem Fall haat man im Outputlyer nur ein Neuron, welches das berechnete Alter ausgibt. Außerdem gibt es das Problem der Klassifizierung. Hier sollen gewisse Daten soweit interpretiert werden, um dann auf eine Klasse sclhießen zu können. Man kann bspw. Pflanzen kategorisieren und hier mit mehreren Neuronen im Outputlayer, wobei jedes Neuron typischerweise einer Klasse entspricht, sagen, zu welcher Wahrscheinlichkeit die gegebenen Daten welcher Klasse entsprechen. [4]

Das Modell zeichnet sich dadurch aus, dass jegliche Art von Input-Variablen verwendet werden kann, sofern sie numerisch ist. Das Netzwerk lernt durch verschiedene Lernalgorithmen, unter Anderem sog. Backpropagation-Algorithmen, welche das Netz (die Gewichtungen) anpassen, um auf ein besseres

Ergebnis zu kommen. Die Ergebnisse von neuronalen Netzen sind bei genügend historischen Daten die zum Lernen verwendet werden können (supervised learning) äußerst präzise und liefern somit eine gute Performance. Ein Problem ist jedoch die Undurchsichtigkeit dieses Modells, außerdem können nur eine eingeschränkte Anzahl an Inputvariablen verwendet werden, da jede Variable mehr das Netz deutlich verkompliziert und die Rechenzeit enorm verlängert. [1]

Neuronale Netze bieten jedoch auch einige Probleme bei der Ausführung. Zunächst muss darauf geachtet werden, dass sinnvolle Startwerte für die Gewichtungen gefunden werden. Gewichtungen nahe 0 lassen das Netz mit einer Art linearer Annäherung starten, mit zunehmenden Durchläufen wird das Modell immer weniger linear und passt sich an den tatsächlichen Verlauf an. Sind die Werte 0 oder zu knapp an 0, ist es dem Netz nicht mehr richtig möglich die Werte sinnvoll anzupassen. Bei zu großen Werten ist es ähnlich schwierig auf sinnvolle Ergebnisse zu kommen, alleine durch die langsame Anpassung des Netzes. Folglich ist es von Bedeutung mit sinnvollen Werten zu beginnen.[4]

Außerdem besteht noch das Problem des Overfitting. Das bedeutet, dass das Netz mit einer gewissen Menge an Testdaten so häufig trainiert wurde, dass es in diesem Bereich äußerst gut performed, in neuen oder anderen jedoch schlecht. Hier ist es wichtig zu beobachten und festzustellen, wann genug gelernt wurde. Hierzu gibt es verschiedene Möglichkeiten, unter anderem mittels eines Trainingssets und eines Validationsets, den Fehler des Ergebnisses zu berechnen. Zunächst werden beide Fehler kleiner, ab einem gewissen Punkt jedoch, wird lediglich der Fehler bezogen auf die Testdaten kleiner, der auf die Validationdaten wächst jedoch wieder. An dieser Stelle ist der Punkt erreicht, an dem das Modell am besten generalisiert ist.[4]

Um sinnvolle Ergebnisse zu erhalten ist es zudem notwendig, dass die Inputvariablen normalisiert werden. Für gewöhnlich sollten sich die Inputwerte im Bereich von 0 +- 1 befinden, um bestmögliche Ergebnisse zu erzielen. Durch verschiedengroße Werte an den Inputdaten werden größere natürlich relevanter und kleinere irrelevanter für die Berechnung, dem schafft man mit normalisierung auf sinnvolle Werte Abhilfe.[4]

Zunächst will ich ein Beispiel aus dem Buch [4] geben, um darzustellen, wie ein solcher Vorgang abläuft. Das Beispiel bezieht sich auf ein tatsächliches, es wurden Daten zu Blumen der Art Iris gesammelt und diese Daten sollen einer Klasse (virginica, versicolor und setosa) zugeordnet werden.

Wählen der Testdaten und Setup

Zu den Blumen wurden 4 Messungen genommen: Blütenblattlänge, -breite, Kelchlänge und die Breite des Kelches. Um nun sinnvolle Testdaten für das Netz zu bilden und overfitting zu vermeiden, werden 70% der Daten als Testdaten und 30% als Validationdaten genommen. Es handelt sich um 4 gesammelte Inputwerte, also gibt es 4 Inputneuronen und 3 Klassen in die die Blumen fallen können, also 3 Outputneuronen. Zudem wurden im hidden Layer 5 Neuronen gewählt (eine übliche Wahl in dieser zusammenstellung). Die Gewichtungen wurden wie üblich zufällig generiert. [4]

Training

Die folgende Tabelle gibt die Interation an, den Fehler gegenüber den Testdaten und den Fehler gegenüber den Validationdaten. Bei Iteration 12 ist zu sehen, dass der Fehler der Testdaten weiterhin kleiner wird, jedoch auf Seiten der Validationdaten wieder wächst. [4]

Iteration	AEF Training	AEF Validation
0	0.443202	0.443281
1	0.231005	0.244472
2	0.149747	0.178680
3	0.131696	0.158875
4	0.063039	0.070936
5	0.054594	0.075916
6	0.046256	0.063721
7	0.026284	0.028231
8	0.021823	0.020531
9	0.020630	0.018305
10	0.016621	0.015093
11	0.013766	0.014756
12	**0.013696**	**0.014804**
13	0.013559	0.014864
14	0.012273	0.014848
15	0.009394	0.014826
16	0.008228	0.014819

Abbildung 7: Tabelle der Fehler [4]

Diesem Ergebnis nach, ist das Netzwerk nach der 11. Iteration jenes, das für die Bestimmung verwendet werden sollte. Im Validationset konnte lediglich eine Iris nicht richtig zugeordnet werden, wie im folgenden zu sehen:

Target	Outcome	Count
SETOSA	SETOSA	15
VERSICOLOR	VERSICOLOR	14
VERSICOLOR	VIRGINICA	1
VIRGINICA	VIRGINICA	15

Abbildung 8: Vergleich der Zuordnung im Validationset [4]

Dieser Fehler entspricht einer Fehlkategorisierung von 2.22%. [4]

Ein weiteres Problem mit neuronalen Netzen ist jedoch die steigende Ungenauigkeit mit zunehmender Komplexität. Mit vielen Hidden Layern und Inputvariabeln, wirkt sich der Fehler bei der Backpropagation nur noch äußerst wenig auf die Gewichtungen aus und wird somit ungenauer. Zudem bewirkt eine Hohe Anzahl an Inputknoten, dass eine enorme Anzahl an Gewichten zwischen den Knoten berechnet werden muss, was zu enormen Performanceeinbußen führen kann. [4]

Vergleich

Lineare und logistische Regression: [1]

- Vorteile:
  - Interpretierbar
  - Schnelle Ausführung
  - Einfache SW

- Nachteile:
  - Viel Arbeit
  - Annahme bei Verteilung oder Linearität
  - Nicht die beste Performance

Neuronale Netze: [1]

- Vorteile:
  - Alle Arten von Daten können analysiert werden
  - Keine Annahmen bzgl. Verteilung oder Linearität
  - Gute Performance
  - Wenig Arbeit
- Nachteile:
  - Schwierig zu interpretieren
  - Langsame Ausführung
  - Spezielle SW

Zusammenschließen verschiedener Methoden

Es ist durchaus üblich mehrere Methoden zusammenzuschließen. Hier gibt es verschiedene Arten der Verwaltung der Ergebnisse. Est ist bspw. möglich, mithilfe von Votes entscheiden zu lassen. Angenommen man hat verschiedene Algorithmen (2 oder mehr) und ein Klassifizierungsproblem, kann man die Stimmen zusammenzählen, die für die einzelnen Klassen sind und demnach entscheiden. Die Klasse mit den meisten Votes gewinnt. Insbesondere bei der Berechnung eines numerischen Wertes ist es jedoch auch üblich den Durchschnitt der Ergebnisse zu bilden und dadurch auf eine Lösung kommen. [4]

WEKA

Bei Weka handelt es sich um eine Sammlung von in Java implementierten Data-Mining-Algorithmen. Diese sind zum Einen sowohl über die Konsole als auch eine GUI zu verwenden und zum Anderen über eine API ansprechbar. Besonders die GUI ist äußerst simpel gehalten und bietet jedem Benutzer ohne Hintergrundwissen im Bereich der IT eine Möglichkeit eine Vielzahl an Algorithmen zu verwenden.

Input-Daten

Von WEKA wird empfohlen deren ARFF (Attribute-Relation File Format) für deren Anwendung zu verwenden (CSV ist auch unterstützt). Bei diesem Dateityp handelt es sich um ein ASCII-Textfile, welches zu verwendende Datensets beschreibt. Ein solches File besteht aus einem Header-Teil und den Daten. [6]

Der Header wird aus 2 Arten von Daten gebildet:

- Relation
  Name der Relation
- Attribute
  Name und Typ des Attributs

Diese werden im File mit @ gekennzeichnet und geben Informationen zu den Daten.

Ein Header eines solchen Files sieht bspw. wie folgt aus: [6]

```
% 1. Title: Iris Plants Database
%
% 2. Sources:
%      (a) Creator: R.A. Fisher
```

```
%      (b) Donor: Michael Marshall (MARSHALL%PLU@io.arc.nasa.gov)
%      (c) Date: July, 1988
%
@RELATION iris

@ATTRIBUTE sepallength  NUMERIC
@ATTRIBUTE sepalwidth   NUMERIC
@ATTRIBUTE petallength  NUMERIC
@ATTRIBUTE petalwidth   NUMERIC
@ATTRIBUTE class        {Iris-setosa,Iris-versicolor,Iris-virginica}
```

Dieser Header beschreibt die Relation iris (wie aus dem Kapitel Neural networks bekannt) mit den 4 numerischen Attributen sepallength, sepalwidth, petallength und petalwidth und dem nominal Attribut class mit den 3 gegebenen Möglichkeiten.

Um nun die gesammelten Daten darzustellen, muss zunächst @DATA angegeben werden, woraufhin alle Daten gelistet werden können. Die folgenden Daten beschreiben alle Daten zu gesammelten Blumen der Klasse „Iris-setosa“: [6]

```
@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
```

Für die Besschreibung der Attribute stehen die folgenden Datentypen zur Verfügung: [6]

- Numeric
- Integer (treated as numeric)
- Real (treated as numeric)
- Nominal specification (wie bei class)
- String
- Date[<date format>]
- Relational (for multi-instance data)

Bei Nominal specification müssen statt dem Datentyp die Optionen mit ``,`` getrennt in ``{}`` eingeschlossen aufgezählt werden (siehe class).

Ein @relational <name> Attribut erwartet nach dem Tag weitere Attribut-Definitionen, die mit einem @end <name> beendet werden: [6]

```
@attribute molecule_name {MUSK-jf78,...,NON-MUSK-199}
@attribute bag relational
  @attribute f1 numeric
  ...
  @attribute f166 numeric
@end bag
@attribute class {0,1}
```

Die Header-section besitzt lediglich eine Deklaration (@DATA). Fehlende Werte werden durch ein einzelnes ``?`` beschrieben. Für weitere Informationen zu den einzelnen Datentypen siehe [6].

Dieses Format kann auch programmatisch in Java verwendet werden. Das Format besitzt die gleichen Möglichkeiten wie als richtiges File, wird lediglich als Objekt abgebildet. Für weitere Information siehe [7].

Verfügbare Tools

Weka stellt eine Vielzahl an Algorithmen zur Verfügung. Alle hier genannten Algorithmen sind unter den folgenden Namen verfügbar:

- Multiple lineare Regression = LinearRegression
- Logistische Regression = Logistic
- Neural networks = Multilayer Perceptron

Alle weiteren Tools sind unter folgendem Link zu finden: [10]

Big Data

Weitere Regressionsmodelle

Bei Big Data liegt der große Unterschied bei den Inputdaten. Durch die riesige Menge an Daten können wesentlich extremere Aufzeichnungen gefunden werden, außerdem bilden sich komplexere Zusammenhänge zwischen den Variablen. Für die Analyse von Big Data wird jedoch äußerst häufig mithilfe von Regressionsmodellen gearbeitet. Diese Modelle sind einfach parallelisierbar und der Output ist für jeglichen Kunden leicht verständlich.[4]

Weka

Weka unterstützt mittlerweile Data Mining mit Big Data. Das Hauptproblem ist der Fakt, dass die GUI von Weka das gesamte Datenset in den RAM lädt, was bei Big Data zu Problemen führt. Die Verwendung der Kommandozeile ermöglicht jedoch schon die Verarbeitung wesentlich größerer Datenmengen, auch solche die nicht komplett in den RAM geladen werden können. Zudem ist es Möglich große Datenmengen programmatisch anzusprechen. Für bessere Performance interstütz Weka nun auch die Verteilung des Dienstes. [8]

Für die Verwendung von Weka mit Big Data ist jedoch auch die Reduzierung der Daten erwünscht. Offensichtlich sollten überflüssige Attribute ausgelassen werden. Für Attribute die an vielen Stellen den Wert „0“ (nicht keinen !!!) besitzen, gibt es das Sparse-Data Format von Weka, welches Speicher sparen kann. Genauere Information zur Verwendung dieses Format befindet sich unter [7].

Einige Algorithmen die von Weka unterstützt werden (siehe [9]) können inkrementell trainiert werden. Ergo muss sich nur jeweils eine Zeile zu jedem Zeitpunkt im Speicher befinden, was rein hypothetisch eine unendliche Menge an Daten ermöglicht.

Quellen

[1] A practical guide to Data Mining for Business and Industry, Andrea Ahlemeyer-Stubbe & Shirley Coleman, publiziert bei Wiley & Sons Ltd.

[2] Data Mining A Knowledge Discovery Approach; K.J. Cios, W. Pedrycz, R.W. Swiniarski, L. Kurgan; publiziert bei Springer

[3] M1: DataWarehousingund DataMining, Business Informatics Group, TU Wien, 2003

[4] Big Data, Data Mining and Machine Learning, Jared Dean, publiziert bei Wiley & Sons Ltd.

[5] Making Sense of Data 1; Glenn J. Myatt, Wayne P. Johnson; publiziert bei Wiley & Sons Ltd.

[6] ARFF (stable version), Weka, http://weka.wikispaces.com/ARFF+%28stable+version%29 , abgerufen am: 31.05.2016

[7] Programmatic Use, Weka, http://weka.wikispaces.com/Programmatic+Use, abgerufen am: 31.05.2016

[8] Mining Big Data Using Weka 3, Weka, http://www.cs.waikato.ac.nz/ml/weka/bigdata.html , zuletzt abgerufen am: 31.05.2016

[9] Handling Large Data Sets with Weka, Weka, http://wiki.pentaho.com/display/DATAMINING/Handling+Large+Data+Sets+with+Weka , zuletzt abgerufen am: 31.05.2016

[10] Data Mining Algorithms and Tools in Weka, Weka, http://wiki.pentaho.com/display/DATAMINING/Data+Mining+Algorithms+and+Tools+in+Weka , zuletzt abgerufen am: 31.05.2016