Hochschule Osnabrück
University of Applied Sciences

Intro

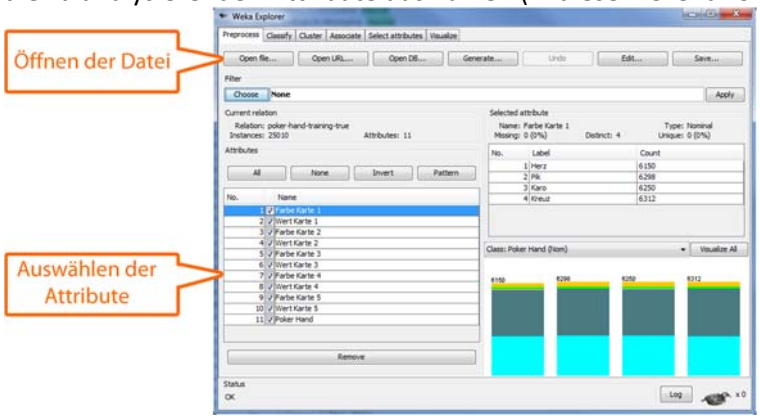
Dieses Übungsblatt besteht aus den drei Teilen des Data Minings, wie sie in der Vorlesung behandelt wurden: Klassifikation, Clustering und Frequent Pattern Mining. Zu jedem dieser Teile gibt es jeweils eine Aufgabe, die mit dem Programm „Weka“ bearbeitet werden müssen. Jeder Teil enthält des Weiteren eine praktische Aufgabe, in der verschiedene Algorithmen „per Hand“ umgesetzt werden sollen, um die Funktionsweise zu verinnerlichen.

Das Programm - Weka

Das Programm „Waikato Environment for Knowledge Analysis“ (Weka) ermöglicht Ihnen verschiedene Data Mining Aufgaben zu lösen, indem es Ihnen neben einer Benutzeroberfläche und verschiedene Algorithmen zur Klassifikation, Clustering oder Assoziation auch die Visualisierung von Data Mining Aufgaben bietet.

Laden Sie Weka herunter und installieren Sie es. Es empfiehlt sich, die Größe des Heaps (den von Java für Weka zugesicherten Speicher) zu vergrößern. Gehen Sie dazu in das Installationsverzeichnis von Weka und erhöhen Sie den Wert von `maxheap` (sinnvoll sind mind. 1024MB) in der `RunWeka.ini`.

Wenn Sie Weka starten, wählen Sie die „Explorer“-Funktion aus. Sie können dann die Datei dort öffnen und unterhalb die zu analysierenden Attribute auswählen (in diesem Szenario sind es alle).



Oberhalb finden Sie die verschiedenen Funktionen, die Weka ermöglicht. Diese werden in den folgenden Aufgaben verwendet, um den gegebenen Datensatz zu analysieren.

Wenn Sie in einem Modus sind, können Sie oben in der Anwendung über „Choose“ den auszuführenden Algorithmus wählen. Einige Algorithmen sind aufgrund der Datenbeschaffenheit eventuell nicht verfügbar. Des Weiteren können Sie rechts neben dem Button „Choose“ auf das Texteingabefeld klicken, indem der Name des aktuellen Algorithmus steht, um weitere Einstellungen für einen Algorithmus machen zu können.

Datenbanken in der Automatisierungstechnik

Aufgabenblatt 5 (Datenanalyse)

Prof. Dr.-Ing. Heiko Tapken

WS 2015/16

Hochschule Osnabrück
University of Applied Sciences

Aufgabe 2.1 (Klassifikation – Weka)

Der Datensatz - Poker

Im Stud.IP finden Sie die Datei `poker-hands.zip`. Diese enthält Daten, die mehrere Hände beim Pokern wiedergeben. Hierzu wird eine Hand mit 5 Spielkarten wie folgt dargestellt: 5 Kombinationen aus *Farbe* und *Wert* (je eine Kombination entspricht einer Spielkarte). Hierbei kann *Farbe* die Werte *Herz*, *Karo*, *Pik* und *Kreuz* annehmen. Wert enthält jeweils eine Zahl von 1 bis 13 mit folgender Zuordnung:

Wert	Karte
1	Ass
12	Dame

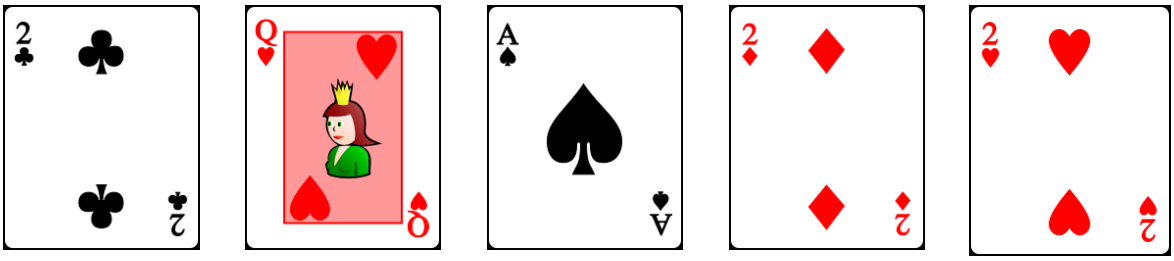
Wert	Karte
2-10	Wert 2-10
13	König

Wert	Karte
11	Bube

Eine elfte Spalte gibt dann die *Pokerhand* an, die – wie im normalen Poker – wie folgt geordnet sind: *Höchste Karte, ein Paar, zwei Paar, Drilling, Straße, Flush, Full House, Vierling, Straight Flush, Royal Flush*. Zum besseren Verständnis folgendes Beispiel:

```
Kreuz, 2, Herz, 12, Pik, 1, Karo, 2, Herz, 2, Drilling
```

Diese Zeile würde dann folgende Hand darstellen, welche dann als höchste Pokerhand einen Drilling liefern würde:



In der ZIP-Datei befinden sich zwei Datensätze. Zum einen gibt es eine `poker-hand-training.arff` und zum anderen eine `poker-hand-testing.arff`. Die Trainingsdatei enthält 25.010 mögliche Farbe/Wert-Kombinationen, die mit einem normalen 52er Kartenset möglich sind. Zu jeder Kombination enthält es die dazugehörige höchste Pokerhand, die mit den jeweiligen 5 Karten möglich ist. Die Testdatei enthält zusätzliche 100.000 mögliche Pokerhände im selben Format, so dass dort auch die 10 Attribute für die Hand und ein Attribut für die Pokerhände existieren.

Die Aufgabe

Das Ziel dieser Aufgabe ist es, dass mit Hilfe der Trainingsdatei ein Klassifikator für den Pokerdatensatz erlernt wird. Dieser Klassifikator soll dann genutzt werden, um im Trainingsdatensatz anhand der Farbe/Wert-Kombinationen die mögliche höchste Pokerhand zu klassifizieren. Anschließend soll geschaut werden, wie gut der Klassifikator ist, indem die klassifizierte Pokerhand mit der tatsächlichen verglichen wird. Gehen Sie dazu wie folgt vor:

a) Öffnen Sie die `poker-hand-training.arff` und wählen Sie alle Attribute zum Analysieren aus. Gehen sie dann in den *Classify*-Modus über. Wählen Sie dort *Supplied test set* als Testmethode aus und wählen die `poker-hand-testing.arff` als Testdatensatz aus. Oberhalb können Sie verschiedene Klassifikationsalgorithmen auswählen. Machen sie mehrere Durchläufe mit folgenden Klassifikationsalgorithmen und geben Sie jeweils die von Weka berechneten Recall und Precision (Durchschnittswerte) an:
   a. *DTNB* Algorithmus
   b. *NaiveBayes* Algorithmus
   c. *DecisionTable*
   d. *LADTree* Algorithmus

b) Schauen Sie sich die Ergebnisse der verschiedenen Algorithmen an. Gehen Sie davon aus, dass Sie einen Pokerspieler mit folgenden Eigenschaften haben:
   a. Wurde eine Pokerhand klassifiziert, dann möchte er sicher sein, dass diese Klassifikation auch korrekt ist. Er möchte also nicht z.B. ein Blatt als Flush klassifiziert bekommen, obwohl es nur ein Drilling ist.
   b. Wurde eine Pokerhand klassifiziert, dann ist es ihm nicht so wichtig, ob z.B. ein Full House auch als Full House erkannt wurde. Ihm würde es dort z.B. auch reichen, wenn Fehler auftreten, da z.B. ein Full House auch ein Drilling sein kann.

   Überlegen Sie nun, welcher der Algorithmen bzw. Klassifikator für den Pokerspieler besser wäre und warum dieser?

Aufgabe 2.2 (Klassifikation - Praktisch)

Gegeben Sei folgende Tabelle als Trainingsdatensatz:

Abteilung	**Status**	**Altersgruppe**	**Gehaltsstufe**
Verkauf	Senior	31…35	A
Verkauf	Junior	26…30	A
Verkauf	Junior	31…35	B
IT	Junior	21…25	A
IT	Junior	31…35	A
IT	Junior	26…30	B
IT	Senior	41…45	B
Marketing	Senior	31…35	B
Marketing	Junior	31…35	A
Marketing	Senior	41…45	B
Marketing	Junior	26…30	B

Erzeugen Sie aus dieser Tabelle einen Entscheidungsbaum. Sei *Status* das zu klassifizierende Attribut. Nutzen Sie dazu den ID3-Algorithmus. Rechnen Sie daher zunächst die Entropie *entropy(T)* aus. Bestimmen Sie dann jeweils den gewichteten *Information Gain*, um den möglichst besten *Split* zu finden. Folgende hilfreiche Hinweise:

- Die entropy für ein Attribut ist 0, wenn eine Ausprägung nicht vorhanden (also ein $p_i = 0$) ist.
- Verwenden Sie den Logarithmus zur Basis zwei.
- Verzichten Sie auf ein Tree-Pruning.

Zeichen Sie den resultierenden Baum.

Aufgabe 3.1 (Clustering – Weka)

Der Datensatz - Zoo

In diesem Datensatz, der unter dem Namen `zoo.arff` im Stud.IP bereitgestellt ist, finden Sie 101 Tiere, die in einem Zoo leben. Zu jedem Tier wurden 18 Attribute erfasst:

- Der Name des Tiers
- 15 Boolean-Werte, die angeben, ob das Tier über die jeweilige Eigenschaft verfügt. Diese sind z.B. ob es Eier legt oder über Haare oder Flossen verfügt.
- 2 numerische Werte. Zum einen gibt es die Anzahl der Beine bzw. Extremitäten (0, 2, 4, 5, 6, 8)1 und zum anderen gibt es eine Zuordnung zu einer Typ-Klasse von 1 bis 7.

Hierzu ein Beispiel:

```
bear, true, false, false, true, false, false, true, true, true, true,
false, false, 4, false, false, true, 1
```

Der Bär hat hiernach Haare und keine Federn, legt keine Eier, gibt Milch etc. Der letzte Wert, das Attribut `type` gibt hier entsprechend die Typklasse 1 an.

Die Aufgabe

Öffnen Sie die `zoo.arff` mit Weka und gehen Sie in anschließend in den *Cluster*-Modus. Wählen Sie dort das Attribut `type` als „Classes to clusters evaluation“ aus.

a) Machen sie mehrere Durchläufe mit folgenden Clusteringalgorithmen. Benutzen Sie jeweils als Einstellung, dass Sie 7 Cluster erzeugen wollen (in Anlehnung an die möglichen Ausprägungen des Typattributs) und geben Sie jeweils den von Weka berechneten Wert an, wie viele Cluster falsch zugeordnet wurden:
   a. *EM* Algorithmus
   b. *SimpleKMeans* Algorithmus
   c. *FarthestFirst* Algorithmus
   d. *MakeDensityBasedClusterer* Algorithmus

1 Wer sich wundert: Der Seestern hat 5

b) Welcher Clustering-Algorithmus hat hier am besten und welcher am schlechtesten abgeschnitten?

Aufgabe 3.2 (Clustering – Praktisch)

Angenommen Sie haben folgenden Datensatz:

ID	X	Y
1	5	3
2	12	14
3	6	5
4	8	11
5	3	9
6	10	16
7	15	1
8	19	14
9	5	12
10	4	3

Führen Sie den K-Means-Algorithmus schrittweise durch, mit dem Ziel 3 Cluster zu erkennen. Verwenden Sie dazu nur die Attribute X und Y, sowie den euklidischen Abstand (2 Stellen hinter dem Komma reichen aus). Nehmen Sie als zufällige Mittelpunkte für die 3 Cluster die ersten drei gegebenen Objekte. Geben Sie sowohl zu jedem Objekt das zugeordnete Cluster an, als auch zu jedem Cluster dessen Centroid.

Aufgabe 4.1 (Frequent Pattern Mining – Weka)

Der Datensatz - Bank

Der Bankdatensatz `bank.arff` liefert ihnen Daten über 600 Bankkunden. In diesem gibt es die Attribute `age, sex, region, income, married, children, car, save_act, current_act, mortage, pep`. Hierbei bedeuten diese: die Altersklasse, das Geschlecht, die Region des Wohnsitzes, die Einkommensklasse, ob verheiratet, Anzahl der Kinder, im Besitz eines Autos, gesichertes Arbeitsverhältnis, ein aktuelles Arbeitsverhältnis, besitzt eine Hypothek und bekommt privilegierte Angebote.

Die Aufgabe

Das Ziel dieser Aufgabe ist es, bestimmte Regeln und Muster in den Datensätzen zu finden, die der Bank helfen können auch ohne vollständige Datensätze (wenn z.B. nur Alter und Geschlecht vorliegt) sinnvolle aussagen zu treffen (z.B. ob diejenige Person privilegierte Angebote bekommen soll).

a) Starten Sie Weka und öffnen Sie die Datei `bank.arff`. Wählen Sie alle Attribute aus und gehen sie in den *Associate*-Modus.

Verwenden Sie den *Apriori*-Algorithmus mit *Confidence* als Metriktyp (*metricType*), einer minimalen Metrik (*minMetric*) von 0.8 und 100 Regeln (*numRules*) lassen Sie sich zusätzlich die *ItemSets* ausgeben.

b) Beantworten Sie anschließend folgende Fragen, indem Sie (ggf. mehrere) der berechneten häufigsten Item-Sets betrachten:

   a. Wie viele Leute haben ein Einkommen von mehr als 43.759 $?

   b. Haben verheiratete Autobesitzer eher eine Hypothek oder eher keine Hypothek?

   c. Wie viele verheiratete Frauen mit einem sicheren und aktuellen Arbeitsverhältnis haben keine Hypothek?

c) Wenn Sie ein/e Bänker/in wären und Ihnen die durch den Apriori-Algorithmus berechneten Regeln vorliegen. Auf welche Eigenschaften können Sie schließen, wenn Sie folgende Kunden sehen würden, jedoch zu 93% oder mehr sicher sein wollen, dass die Aussage auch stimmt:

   a. Der Kunde hat ein Einkommen von 45.000 $

   b. Der 22-jährige Kunde ist verheiratet und wohnt im Stadtzentrum

   c. Eine 33 jährige ohne Kinder, hat keine Hypothek und keine privilegierten Angebote.

d) Wenn Sie in einer Bank arbeiten, müssen sie eventuell beurteilen können, ob ein Kunde Ihnen ggf. Angaben macht, die nicht stimmen, um eventuell ein besseres Angebot zu bekommen. Verwenden Sie ihre erzeugten Regeln und gehen Sie davon aus, dass Ihre Schlussfolgerungen mit einer Wahrscheinlichkeit von mindestens 91% korrekt sein sollen. Welcher Kunde sagt die Wahrheit und welcher Kunde nicht?

   a. „Ich verdiene 65.000 $, aber leider hab ich keinen sicheren Job“

   b. „Ich bin erst 18, bin schon verheiratet, habe aber kein Auto – das ist aber bei einem Gehalt von 20.000 $ auch nicht machbar“

   c. „Ich bin ein 26 Jahre alter Mann und bekomme 50.000 $ im Jahr“

e) Als Bänker/in müssen Sie oft entscheiden, ob jemand ein entsprechendes Angebot bekommt oder nicht. Schauen Sie sich daher folgende Personendaten an und entscheiden Sie anhand Ihrer 100 Regeln, ob Sie dem Kunden ein privilegiertes Angebot machen:

   a. Die Frau ist verheiratet, hat keine Kinder und keine Hypothek

   b. Der Kunde hat ein Kind und einen sicheren Arbeitsplatz

c. Der Mann hat ein Kind und ein Auto, aber kein Hypothek

d. Die Person ist verheiratet, ist kinderlos und hat einen aktuellen und gesicherten Arbeitsplatz

Aufgabe 4.2 (Frequent Pattern Mining - Praktisch)

Gegeben sei folgende Transaktionstabelle eines Kaufhauses, die zum einen eine Rechnungsnummer und zum anderen eine Liste mit gekauften Produkten beinhaltet:

Rechnung	Produkte
1001	Bier, Aspirin, Windeln
1002	Pizza, Bier
1003	Bier, Wein
1004	Pizza, Aspirin, Bier
1005	Aspirin, Wein
1006	Bier, Wein
1007	Wein, Aspirin
1008	Aspirin, Windeln, Bier, Wein
1009	Wein, Bier, Aspirin

Wenden Sie den Apriori-Algorithmus an, um die Frequent Pattern zu finden. Nehmen Sie dazu einen Minimum-Support von 2 an. Geben Sie alle gefunden Frequent Pattern an!