Data Mining und Maschinelles Lernen

TECHNISCHE UNIVERSITÄT DARMSTADT

Wintersemester 2015/2016
Projekt: Anmeldung: 6.12.
Abgabe 1-3: 13.12. / Abgabe 4-6: 17.01. / Abgabe 7, 8, 9, Implementation: 7.02.

Ziel des Projektes ist es, praktische Erfahrungen im Maschinellem Lernen zu sammeln. Hierzu sollen mehrere Projektaufgaben mit Hilfe des Machine Learning Frameworks *Weka* (`http://www.cs.waikato.ac.nz/ml/weka/`) gelöst werden. Des Weiteren folgt im späterem Verlauf eine Implementierungsaufgabe. Das Projekt kann allein bzw. in einer Kleingruppe (maximal 3 Studierende) bearbeitet werden. Die Registrierung, Anmeldung und Bildung der Gruppen findet online unter dem Upload-System `https://www.ke.tu-darmstadt.de/webmining-upload/mldm15` bis zum jeweiligen Stichtag statt.

Die Abgaben erfolgen über das gleiche System bis zu den jeweiligen Stichtagen. Folgendes ist zu den Abgaben zu beachten:

- **Format:** Die Abgabe erfolgt als **ZIP-Datei**, die das Lösungsdokument enthalten soll. Das Lösungsdokument soll eine **Präsentation** im PDF-Format sein (z.B. 4:3 Querformat). Der Umfang einer Abgabe sollte 15 Folien nicht überschreiten.

- **Inhalt** Die Präsentation soll **selbsterklärend** sein. Das bedeutet, dass die Lösung ohne mündliche Erklärung nachvollziehbar sein soll. Das Dokument muss genügend Erläuterungen und Ausführungen enthalten. Eine reine Ansammlung von Graphiken und Tabellen ohne jegliche Begleittexte ist hierfür nicht ausreichend. Die **Bewertung** findet allein anhand der PDF-Datei statt (abgesehen von der Implementationsaufgabe).

- **Abbildungen** Tabellen, Diagramme, etc. müssen **vollständig beschriftet** sein, d.h. sie müssen zumindest direkt an der Abbildung eine kurze Beschreibung enthalten und ausreichend kommentiert sein. Insbesondere sollte bei Abbildungen von Resultaten der verwendete Datensatz angegeben werden.

- **Begleitmaterial** Zusätzlich abgegebene Dateien sollten nur zum Nachweis oder zum Beantworten von Detailfragen dienen. Diese werden im Normalfall jedoch nicht berücksichtigt.

Für die Programmieraufgabe ist der Java-Quellcode als *einzelne* NearestNeighbor.java einzureichen (keine GIT/SVN Links). Die Beantwortung der Implementationsfragen erfolgt wie üblich als PDF-Dokument.

Für einen Klausurbonus ist es nicht zwingend nötig alle Aufgaben zu bearbeiten. Bei Teilabgaben kann noch ein entsprechender Teilbonus erreicht werden. Das Implementationsprojekt umfasst etwa 1/3 der zu erreichenden Punkte und sollte ca 10 Stunden/Person dauern.

In den jeweiligen Aufgaben verwenden Sie eine vorgegebene Anzahl von Datensätze, die Sie der Sammlung von Klassifikations- und Regressionsdatensätzen unter `http://www.ke.informatik.tu-darmstadt.de/lehre/ws-15-16/mldm/projekt/datasets.zip` entnehmen, falls die Datensätze nicht in der Aufgabenstellung festgelegt werden. Achten Sie hierbei bitte darauf, möglichst unterschiedliche Datensätze zu wählen und diese in den einzelnen Aufgaben zu variieren. Bei der Auswahl der Datensätze ist weiterhin zu beachten, dass bestimmte Lernverfahren nicht mit allen Datensätze umgehen können. Optional können Sie dieses Problem beheben, indem Sie die Daten vorverarbeiten, z.B. mittels `FilteredClassifier` und einem entsprechendem Preprocessing-Filter. In diesem Fall bzw. falls Sie die Standardeinstellungen in Weka modifizieren, geben Sie dies bitte in ihrer Lösung an.

Die Informationen zum Programmierprojekt finden Sie in einem getrennten Dokument, siehe Webseite. Nach Abschluss der Vorlesung *Instanzbasiertes Lernen*, bzw. Projektaufgabe 6, sollten alle relevanten Vorkenntnisse für die Implementierung vorliegen.

Aufgabe 1 Regellernen: Anwendung und Interpretation (3 Punkte)

In dieser einführenden Aufgabe sollen Sie die Verwendung von WEKA erlernen und dabei die Ergebnisse dreier Regellerner auf drei unterschiedlichen Datensätzen vergleichen. Wenden Sie hierzu die Regellerner `ConjunctiveRule`, `JRip` und `Prism` auf 3 Klassifikationsdatensätze an. Benutzen Sie hierfür keine Cross-validation, sondern *training set* als Test-Option.

a) Vergleichen Sie die Anzahl der Regeln, der Bedingungen und der vorhergesagten Klassen der resultierenden Regelmengen jeweils in Bezug auf

   - die einzelnen Datensätze
   - die jeweiligen Regellerner

b) Existiert bei allen Algorithmen eine Default-Rule? Wenn ja:

   - Welche Klasse wird üblicherweise als Default-Rule ausgewählt?
   - Wie interpretieren Sie die Güte dieser Default-Rule?

c) Läßt sich anhand der vorherigen Teilaufgaben eine Aussage treffen, welche der drei Datenmengen am leichtesten bzw. am besten zu lernen ist?

d) Vergleichen Sie die Regelmengen der Algorithmen `JRip` und `Prism` für den Datensatz `contact-lenses.arff`. Wie schätzen Sie die Allgemeinheit der von `JRip` bzw. `Prism` gefundenen Regeln ein? Beachten Sie hierbei, daß `JRip` als Heuristik Information Gain und `Prism` Precision verwendet.

Aufgabe 2 Evaluation von Regellernern (4 Punkte)

In dieser Aufgabe sollen unterschiedliche Evaluierungsmethoden unter Verwendung von WEKA eingesetzt und deren Ergebnisse diskutiert werden. Wenden Sie den Regellerner `JRip` auf 5 Datensätze an. Teilen Sie hierzu jeden Datensatz zunächst in 2 gleich große, stratifizierte Teile, einer Trainingsmenge und Validierungsmenge, auf. Eine stratifizierte Aufteilung kann mit dem Filter `StratifiedRemoveFolds` erreicht werden.

a) Trainieren Sie nun `JRip` auf jeder dieser Trainingsmengen (ggf. auf Teilen dieser Mengen, siehe Cross-Validation usw.) und evaluieren Sie die Genauigkeit (prozentualer Anteil der korrekt klassifizierten Beispiele) der resultierenden Klassifizierer jeweils mittels:

   - 1x5 Cross-Validation
   - 1x10 Cross-Validation
   - 1x20 Cross-Validation
   - Leave-One-Out
   - bzw. auf der Trainingsmenge

   Wie schätzen Sie die Qualität der erhaltenen Genauigkeitsabschätzungen ein?

   Anmerkung: *In dieser Teilaufgabe sollen vorerst keine Veränderungen an weiterführenden Einstellungen, wie z.B. andere Random-Seeds, vorgenommen werden.*

b) Wiederholen Sie Aufgabe a) mit dem Unterschied, daß Sie nun eine 10x10 Cross-Validation zur Evaluation verwenden sollen. Wenden Sie hierzu zehnmal eine 1x10 Cross-Validation mit 10 unterschiedlichen Random-Seeds an und mitteln die erzielten Genauigkeiten.

   Vergleichen Sie die so erzielte Genauigkeitsabschätzung mit den Abschätzungen aus der Aufgabe a).

   Führt ihrer Meinung nach eine geschickte Auswahl von Random-Seeds zu einer besseren Abschätzung?

c) Bestimmen Sie die Genauigkeit auf der Validierungsmenge (d.h. verwenden Sie diese als Testmenge). Wie schätzen Sie nun unter der Annahme, daß es sich bei der Validerungsmenge um einen realen Anwendungsfall handelt, die Abschätzungen der Evaluierungsmethoden aus den Aufgaben a) und b) ein?

Aufgabe 3 ROC-Kurven (2 Punkte)

a) Vergleichen Sie für einen ausgewählten Klassifikationsdatensatz die ROC-Kurven bzw. die Fläche unter diesen Kurven für die Klassifizierer `J48` und `NaiveBayes`. Sie können die ROC-Kurven betrachten, indem Sie mit der rechten Maustaste im Fenster “Result List” den Menü-Punkt “Threshold List” auswählen.

b) Interpretieren Sie die Resultate. Sie können die Werte, die zum Zeichnen der Kurve verwendet wurden, auch mit “Save” in ein ARFF-File exportieren, und dieses (nach Löschen des Headers) in Grafik-Programme importieren. So können Sie z.B. beide Kurven (für `J48` und `NaiveBayes`) übereinander legen.

Aufgabe 4 Entscheidungsbäume (3 Punkte)

a) Wählen Sie 2 Klassifikationsdatensätze aus. Vergleichen Sie für diese Datensätze die ROC-Kurven bzw. die Fläche unter diesen Kurven für die Klassifizierer `J48` einmal mit und einmal ohne Pruning (Option 'unpruned') und `ID3`. Bei `J48` verwenden Sie für die anderen Optionen die Default-Werte.

b) Vergleichen Sie die Klassifizierer ebenfalls mit den Accuracy-Werten der Cross-Validation.

c) Betrachten Sie auch die Größe der entstandenen Bäume (Anzahl Knoten und/oder Blätter im Baum) und setzen Sie diese in Zusammenhang mit der Güte der Klassifizierer.

Aufgabe 5 Nearest Neighbour (2 Punkte)

a) Verwenden Sie für diese Aufgabe die gleichen Datensets wie in der vorherigen Aufgabe. Finden Sie heraus, für welches $k \in \{1, 3, 5, 7, 9, 11\}$ der Algorithmus `k-NN` (in `weka` heisst der Algorithmus `IBk`; verwenden Sie auch hier die Default-Optionen) die höchste Cross-Validation-Accuracy bekommt.

Aufgabe 6 Regressionsbäume (3 Punkte)

Benutzen Sie die 5 Regressionsdatensätze für diese Aufgabe (außer dem Datensatz `regression`). Für nominale Attribute beachten Sie bitte, dass der Lerner `M5P` eine Binarisierung der Daten vornimmt ($A = a <= 0.5$ bedeutet also: alle Instanzen, für die A *nicht* den Wert a hat). Die Gesamtanzahl der Instanzen ist n, der tatsächliche Wert einer Instanz j ist y_j und der vorhergesagte Wert einer Instanz j ist r_j (genau wie im Skript).

a) Vergleichen Sie den *Mean Absolute Error* ($\frac{1}{n} \cdot \sum_j |y_j - r_j|$) und den *Root Mean Squared Error* ($\sqrt{Mean\ Squared\ Error}$) (10 CV), sowie die Modelle (Interpretierbarkeit/Größe) jeweils für den Regressionsbaumlerner `M5P`, einmal mit angeschaltetem Pruning und einmal ohne Pruning (Benutzen Sie Regressionsbäume, also setzen Sie die Option 'buildRegressionTree' auf 'True'). Bringt Pruning bei Regressionstasks eine Verbesserung?

b) Verwenden Sie nun Model Trees (Option 'buildRegressionTree' auf 'False' setzen, ansonsten Default Optionen). Vergleichen Sie die Model Trees mit den Regressionsbäumen.

Aufgabe 7 Ensemble-Lernen (3 Punkte)

In dieser Aufgabe sollen unterschiedliche Ensemble-Methoden eingesetzt und deren Ergebnisse verglichen werden. Der Entscheidungsbaumlerner J48 soll als Basislerner verwendet werden. Nutzen Sie für diese Aufgabe die Datensätze `labor`, `yeast` und `car`.

a) Bestimmen Sie die Genauigkeit des regulären J48.

b) Verwenden Sie nun Bagging mit J48 und AdaBoost mit J48. Benutzen Sie außerdem noch Random Forests. Bestimmen Sie für die so erhaltenen Klassifizierer die Genauigkeiten für eine stetig wachsende Anzahl von Iterationen (bei den Random Forest verändern Sie bitte die Anzahl der Bäume). Wie interpretieren Sie die Entwicklung der erzielten Genauigkeiten?

Aufgabe 8 Pre-Processing (2 Punkte)

Wählen Sie drei Klassifikationsdatensätze aus. Erstellen Sie für jeden Datenset eine diskretisierte Version unter Verwendung des Filters `weka.filters.unsupervised.attribute.Discretize`.

a) Schätzen Sie die Genauigkeit von `J48` mittels Cross-validation auf den ursprünglichen Daten und auf den diskretisierten Daten ab.

b) Wie interpretieren Sie den Vergleich der Genauigkeiten und der Größe der gelernten Bäume dieser drei Experimente (die Ergebnisse können über die drei Datensets gemittelt werden)?

Aufgabe 9 Klassifikationswettbewerb (3+1 Punkte)

In dieser Übung werden Sie gegen die anderen Kleingruppen in einem Klassifikationswettbewerb antreten. Laden Sie hierfür den Datensatz *purchase* unter `http://www.ke.informatik.tu-darmstadt.de/lehre/ws-15-16/mldm/projekt/purchase.zip` herunter. Der Datensatz besteht aus zwei Teilmengen: *train*, in dem die Instanzen klassifiziert sind, und *test*, in dem jeweils das Klassenattribut unbekannt ist. Ziel ist es, auf der Trainingsmenge einen Klassifizierer zu lernen, der die Testinstanzen korrekt ihren Klassen zuordnet.

Sie können Ihre Vorhersage über das Upload-System hochladen und evaluieren lassen (Tab "Competition"). Es werden alle Ihre Abgaben in einer Tabelle angezeigt, also auch Ihre beste Abgabe im Vergleich zu den besten Abgaben der anderen Gruppen. Pro Kalendertag ist nur **eine Evaluierung** möglich. Verglichen wird die Klassifikationsgenauigkeit.

Sie dürfen für die Berechnung der Vorhersagen das gesamte (legale!) Repertoire, das Sie in den Übungen und in der Vorlesung kennengelernt haben, oder über Weka zur Verfügung steht oder Ihnen sonst wie einfällt, ausschöpfen. Sie sollten jedoch die Maßnahmen verstehen, die Sie anwenden. Geben Sie also bei Verfahren, die nicht in Vorlesung oder Übung behandelt wurden, wenigstens eine kurze Beschreibung in eigenen Worten an.

a) Dokumentieren Sie Ihre Vorgehensweise und Ihren Fortschritt. Berichten Sie z.B. welchen Unterschied eine bestimmte Maßnahme bewirkt hat, und wieso Sie diese ausprobiert haben. Wurden Ihre Erwartungen jeweils erfüllt?

b) Da Sie jeweils nur ein begrenztes Kontingent an Auswertungen auf den Testdaten haben, führen Sie jeweils vorher eigene Abschätzungen aus (siehe Aufgabe 2) und geben Sie, falls Sie doch ein Upload wagen wollen, sowohl die reale Auswertung als auch Ihre Abschätzung an. Entsprechen sich beide Werte? Falls nicht, warum?

Format der Vorhersagen:

- Schreiben Sie Ihre Vorhersagen in eine Datei namens `GXX_predictions` mit `XX` als Gruppennummer. Schreiben Sie dabei für jedes Testdokument eine Zeile im Format *InstanzID,Klassename*, also z.B. beginnend:

  ```
  33334,NonPurchase
  33335,Purchase
  33336,NonPurchase
  33337,Purchase
  ...
  ```

- Sie können Ihre Abgabe teilweise validieren, indem Sie folgende Befehle unter Linux bzw. unter Windows mit den GNU Utilities (`http://unxutils.sourceforge.net/`) auf Ihrer Abgabedatei `file` ausführen:
  - `grep -c "" file`
  - `sort file | uniq | grep -c ""`
  - `sort file | uniq | grep -P -c "^[0-9]{5,5},(Purchase|NonPurchase)$"`

    sollten alle `16667` ergeben.
  - `sort file | diff - file`

    sollte sich höchstens über ein Zeilenumbruch am Ende beschweren.

Weitere Hinweise:

- Man erhält die Vorhersagen wenn man in `Test options` → `More options` bei `Output predictions` ein Ausgabeformat auswählt. Die Vorhersagen werden ins Ausgabefenster geschrieben. Man kann sich auch im KnowledgeFlow die Vorhersagen in eine Datei schreiben lassen.
- Fügen Sie Ihre beste Vorhersagedatei der Abgabe bei.
- Als kleinen Anreiz gibt es für den Erst-Platzierten einen **Bonuspunkt**, für den 2. bis 3. 0,75 Bonuspunkte, für den 4. und 5. 0,5 Bonuspunkt und für den 6. und 7. 0,25 Punkte.