Maschinelles Lernen: Symbolische Ansätze

Projekt

WS 2006/07

Der Fachbereich Informatik misst der Einhaltung der Grundregeln der wissenschaftlichen Ethik großen Wert bei. Zu diesen gehört auch die strikte Verfolgung von Plagiarismus. Mit der Abgabe einer Lösung (Hausaufgabe, Programmierprojekt, Diplomarbeit, etc.) bestätigen Sie, dass (Sie/Ihre Gruppe) (der alleinige Autor/die alleinigen Autoren) des gesamten Materials sind. Falls Ihnen die Verwendung von Fremdmaterial gestattet war, so müssen Sie dessen Quellen deutlich zitiert haben. Bei Unklarheiten zu diesem Thema finden Sie weiterführende Informationen unter `http://www.informatik.tu-darmstadt.de/Plagiarism` *oder sprechen Sie Ihren Betreuer an.*

Auf diesem Blatt finden Sie die Problemstellungen für den Angewandten Teil der Übungen zur Vorlesung Maschinelles Lernen: Symbolische Ansätze. Das Ziel dieser Aufgaben ist es, Ihnen zu ermöglichen, praktische Erfahrungen mit einem Data Mining Werkzeug zu sammeln. Mit einer erfolgreichen Bearbeitung des Projektes können Sie sich in der Klausur um 0,3 Notenpunkte verbessern (allerdings nur bei bestandener Klausur).

Beachten Sie, daß es oft keine eindeutigen Lösungen der Aufgaben gibt. Was zählt ist die praktische Erprobung der Verfahren, die Antworten können sich abängig von den eingesetzten Verfahren bzw. abhängig von den untersuchten Datensätzen unterscheiden.

Es wird erwartet, daß Sie die Aufgaben selbständig bzw. in kleinen Gruppen lösen und die gefundenen Lösungen abgeben. Die Abgabe erfolgt in einem üblichen Textformat an die eMail `mldm@ke.informatik.tu-darmstadt.de`. Die Lösungen werden in der letzten Übungsstunde (Donnerstag, der 8. 2. 2007) kurz durchbesprochen. Eventuell auftretende Probleme können Sie in den laufenden Übungsstunden oder im Vorlesungs-Forum ansprechen.

Sie finden die Homepage der Weka Machine Learning Library unter
`http://www.cs.waikato.ac.nz/ml/weka/`. Dort können Sie sich die Java-Soft- ware, einige Kurzanleitungen, sowie Beispieldatenbanken herunterladen. Wir empfehlen, daß Sie die Experimente mit dem *Explorer* GUI (`weka.gui.explorer.Explorer`) durchführen, Sie können sie aber auch von der Command-line oder einem anderen GUI (`weka.gui.GUIChooser`) durchführen.

Dokumentation zu Weka finden Sie auf der Weka-Homepage. Beachten Sie insbesondere auch die Tipps & Tricks (`http://www.cs.waikato.ac.nz/~ml/weka/tips_and_tricks.html`). Dort wird z.B. erklärt, wie Sie den Speicher der Java VM vergrößern können, falls Probleme auftreten.

Als Lösung einer Aufgabe wird erwartet, daß Sie die wesentlichen Resultate (und die Schritte zu Ihrer Lösung, d.h. z.B. die aufgerufenen Routinen, die verwendeten Parameter,

etc.) schriftlich zusammenfassen bzw. ggf. Grafiken zur Illustration verwenden. Ein Aneinanderhängen der Outputs von Weka ist nicht zielführend.

Lassen Sie sich von Aufgaben, bei denen die Algorithmen noch nicht in der Vorlesung an der Reihe waren, nicht abschrecken. Sie haben genug Zeit für die Bearbeitung des Projektes und es wird im weiteren Verlauf der Vorlesung selbstverständlich auf alle benötigten Algorithmen näher eingegangen.

Als letzte Aufgabe möchten wir einen Wettbewerb anbieten, wo man eine Datenmenge zu Verfügung gestellt bekommt, auf der man mit einem beliebigen Algorithmus eine möglichst hohe Genauigkeit erzielen soll. Die Lösungen werden dann auf einer Testmenge ausgewertet und die beste wird prämiert.

Abgabe der Aufgaben bis **Samstag, den 3. 2. 2007, 23:59 Uhr**

Viel Spaß bei der Durchführung dieser Aufgaben!

1 Aufgabe: Auswahl der Daten

Suchen Sie sich aus den Datensätzen, die mit der Installation von Weka mitkommen und aus denen aus dem `jar`-File auf der Homepage `http://prdownloads.sourceforge.net/weka/datasets-UCI.jar` 10 verschiedene aus, die Sie im weiteren Verlauf des Projektes verwenden. Die Datensätze sollten möglichst unterschiedlich sein, d.h. die Anzahl der Attribute und Klassen sollte variieren und es sollten sowohl nominale als auch numerische Attribute vertreten sein. Auch die Anzahl der Instanzen sollte möglichst unterschiedlich sein.

2 Aufgabe: Evaluierung

Wählen Sie 5 von den 10 Datenmengen aus und testen Sie den Entscheidungsbaumlerner `J48`, eine Implementation von `C4.5`, an diesen Datensets.

- Benutzen Sie die gesamten Daten als Trainingsmenge und bestimmen Sie die Genauigkeit auf dem Trainings-Set.
- Bestimmen Sie die Genauigkeit mittels 2-fold, 5-fold, 10-fold und 20-fold-Crossvalidation.
- Iterieren Sie die Crossvalidation, z.B. mittels 10-facher 10-fold-Crossvalidation.
- Bestimmen Sie den Fehler mittels Leave-one-out cross-validation

Diskutieren Sie die unterschiedlichen Genauigkeitsabschätzungen.

In der Folge wird, so nicht anders angegeben, mit "Genauigkeit" immer die mit 10-facher Cross-validation geschätzte Genauigkeit betrachtet.

3 Aufgabe: Noise und Pruning

Verwenden Sie das Datenset, das in der vorigen Aufgabe die größte Genauigkeit auf dem Trainings-Set erzielt hat.

Stören Sie die Klasseninformation in diesem Datenset durch Hinzufügen von verschiedenen Levels von Noise (z.B., 5%, 10%, 25%, 50%, 75%, 100%; `weka.filters.unsupervised.attribute.AddNoise`).

Beobachten sie die Genauigkeit und Größe der gelernten Bäume auf dem Original bzw. den gestörten Datensets für `J48`

- mit den Default-Parametern
- ohne Pruning (-U und -M 1)

Experimentieren Sie ein wenig mit den Parametern -C und -M und versuchen Sie, die Kombination zu finden, die die höchste Genauigkeit liefert (auf den mit 10% Noise gestörten Daten).

Anmerkung: Ein Noise-Level von x% wird erzeugt, indem bei x% aller Beispiele das Label des Beispiels durch ein zufällig ausgewähltes Label eines der anderen Klassen ersetzt wird. Bei Zwei-Klassen-Problemen werden Sie feststellen, daß die Performanz bei 100% Noise identisch ist mit der Performanz bei 0% Noise (Warum?). Adaptieren Sie in diesem Fall die Schranken in geeigneter Weise (hier entspricht 50% Noise zufälligen Daten).

4 Aufgabe: Regel-Lernen

In Weka finden Sie u.a. `JRip`, eine Nachimplementierung des bekanntesten Regellerners `Ripper`, und `ConjunctiveRule`, ein Lerner, der nur eine einzige Regel lernt (ähnlich dem in der Vorlesung besprochenen `Batch-FindG`).

Vergleichen Sie die Genauigkeit und die Größe (Anzahl der Regeln, Anzahl der Bedingungen) der von `ConjunctiveRule`, `JRip` und `J48` gelernten Konzepte auf den Datensets.

Führen Sie auf den Ergebnissen paarweise einen Signifikanz-Test (Vorzeichen-Test) durch, um festzustellen welche Methode am besten funktioniert.

5 Aufgabe: ROC-Kurven

Vergleichen Sie für einen ausgewählten Datensatz 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.

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.

6 Aufgabe: Pre-Processing

Wählen Sie ein Datenset mit vielen numerischen Attributen aus. Erstellen Sie eine diskretisierte Version (`weka.filters.supervised.attribute.Discretize`).

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

Der Meta-Classifier `FilteredClassifier` erlaubt, eine Kombination einer Pre-processing Methode und eines Classifiers zu einem neuen Classifier zu machen. Erzeugen Sie die Kombination `Discretize` und `J48` und schätzen Sie deren Genauigkeit auf den ursprünglichen Daten ab.

Wie interpretieren Sie den Vergleich der Genauigkeiten und der Größe der gelernten Bäume dieser drei Experimente?

7 Aufgabe: Entdecken von Assoziationsregeln

Das Datenset `adult` (`http://www.ke.informatik.tu-darmstadt.de/lehre/ws0607/mldm/projekt/adult.arff`) enthält Daten von 48842 US Bürgern über Geschlecht, Ausbildung, Familienstand, Beruf, Einkommen (class Variable), etc. Versuchen Sie, mit dem Apriori-Algorithmus aus Weka in diesem Datenset *interessante* Regeln zu finden. Sie können dabei sowohl die Optionen von Weka ausprobieren (z.B. `-T` das Maß, nach dem die Regeln sortiert werden) als auch das Datenset verändern (z.B. durch Entfernen einzelner Attribute). Beachten Sie, daß in der Version zum Download zwei numerische Attribute enthalten sind, die Sie diskretisieren oder einfach entfernen können. Falls die Laufzeiten zu lange werden (mehrere Minuten), können Sie auch auf einer Teilmenge der Daten arbeiten.

8 Aufgabe: Ensemble-Lernen

Vergleichen Sie die Genauigkeit von J48, Bagging mit J48, AdaBoost mit J48 und RandomForest auf fünf Datensets. Erhöhen Sie die Anzahl der Iterationen bei den drei Ensemble-Verfahren und beobachten Sie die Entwicklung der erzielten Genauigkeiten.

9 Wettbewerb

Versuchen Sie, auf dem Dataset `competition.arff` (zu finden unter `http://www.ke.informatik.tu-darmstadt.de/lehre/ws0607/mldm/projekt/competition.arff`) eine möglichst hohe Vorhersage-Genauigkeit (abgeschätzt mit 10-fold Cross-Validation) zu erzielen. Sie können jedes beliebige Verfahren, bzw. jede Kombination von Verfahren (inkl. Pre-Processing) verwenden. Alle Parameter der verschiedenen Verfahren dürfen beliebig justiert werden. Eigene Implementierungen sind selbstverständlich auch zulässig.

Welche Genauigkeit erzielen Sie mit welchem Verfahren? Wie haben Sie den Vergleich durchgeführt, bzw. aufgrund welcher Kriterien haben Sie sich für Ihr favorisiertes Verfahren entschieden?

Wir werden in der letzten Übungsstunde die eingereichten Lösungen (als Lösung gilt die verwendete Kommandozeile, bzw. der Source-Code) vergleichen und den "Sieger" ermitteln.