Hochschule Wismar
University of Applied Sciences
Technology, Business and Design
Fakultät für Ingenieurwissenschaften, Bereich EuI

Semesterarbeit

Data Mining Semesterprojekt Dokumentation

Eingereicht am: 14. Juni 2021

Christoph Werner
geboren am 16. November 1993
Email: christoph.werner@stud.hs-wismar.de

Josef Prothmann
geboren am 16. Dezember 1998
Email: j.prothmann@stud.hs-wismar.de

Dozent: Prof. Dr. rer. nat. Jürgen Cleve

Inhaltsverzeichnis

1 Grundlagen

Bibtexkey [**?**]

1.1 Datenvorverarbeitung

1.2 Knime Knoten für Entscheidungsbäume

Knime selbst bietet einen 'Decision Tree Learner' Knoten und die Extension von Weka biete eine Vielzahl von neuen Knoten für Entscheidungsbäume, welche im Folgenden kurz angesprochen seien.

1.2.1 Decision Tree Learner (Knime)

Laut Beschreibung auf der Homepage von Knime muss das Zielattribut für diesen Knoten nominal sein. Die anderen zur Entscheidungsfindung verwendeten Attribute können entweder nominal oder numerisch sein. Der Algorithmus stellt zwei Qualitätsmaße für die Split-Berechnung zur Verfügung: den Gini-Index und das Gain-Ratio. Weiterhin gibt es eine Post Pruning-Methode, um die Baumgröße zu reduzieren und die Vorhersagegenauigkeit zu erhöhen. Die Pruning-Methode basiert auf dem Prinzip der minimalen Beschreibungslänge. [**?**]

1.2.2 SimpleCart (Weka 3.7)

Ein bedeutendes Merkmal des CART-Algorithmus ist, dass nur Binärbäume erzeugt werden können. CARTs zeichnen sich dadurch aus, dass sie die Daten in Bezug auf die Klassifikation optimal trennen. Dies wird durch einen Attributschwellwert erreicht. Bei den errechneten Entscheidungsbäumen gilt: Je höher der Informationsgehalt eines Attributs in Bezug auf die Zielgröße, desto weiter oben im Baum findet sich dieses Attribut. [**?**]

1.2.3 J48 (Weka 3.7)

Diesem Knoten liegt der C4.5 Algorithmus von J. Ross Quinlan zu Grunde.

Grundlegend verhalent sich dieser Algorithmus und der CART-Algorithmus ähnlich. Der Hauptunterschied besteht darin, dass bei C4.5 keine binäre Aufteilung erfolgen muss, sondern eine beliebige Anzahl Verzweigungen eingebaut werden können. Der Baum wird dadurch deutlich breiter und ist außerdem meist weniger tief als der korrespondierende CART-Baum. Ein weiterer Unterschied zeigt sich beim sogenannten Pruning, beim Stutzen des Baumes. CART erzeugt einige Subtrees und testet diese mit neuen, vorher noch nicht klassifizierten Daten. C4.5 beschneidet den Baum ohne Beachtung der gegebenen Datenbasis. [**?**]

1.2.4 NBTree (Weka 3.7)

Es wurde bereits gezeigt, dass Naive-Bayes-Induktionsalgorithmen bei vielen Klassifizierungsaufgaben erstaunlich genau sind, selbst wenn die bedingte Unabhängigkeitsannahme, auf der sie basieren, verletzt ist. Die meisten Studien wurden jedoch mit kleinen Datenbanken durchgeführt. NBTree einen Hybridalgorithmus aus Entscheidungsbaum-Klassifikatoren und Naive-Bayes-Klassifikatoren. Die Knoten des Entscheidungsbaums enthalten inivariate Splits wie reguläre Entscheidungsbäume, aber die Blätter enthalten Naive-Bayes'sche Klassifikatoren. Der Ansatz behält die Interpretierbarkeit von Naive-Bayes und Entscheidungsbäume, während er zu Klassifikatoren führt, die häufig beide Konstituenten übertreffen, insbesondere in größeren Datenbanken. [**?**]

1.2.5 REPTree (Weka 3.7)

Dieser Knoten stellt einen schnellen Entscheidungsbaum-Lerner. Erstellt wird hierbei eine Entscheidungs-/Regressionsbaum unter Verwendung von Informationsgewinn/Varianz. Dieser wird anschließend unter Verwendung von fehlerreduziertem Pruning (mit Backfitting) bereinigt. Außerdem werden Werte für numerische Attribute nur einmal sortiert. Fehlende Werte werden behandelt, indem die entsprechenden Instanzen in Stücke aufgeteilt werden (d.h. wie beim C4.5 Algorithmus).

1.2.6 LMT (Weka 3.7)

Baumindukti onsverfahren und lineare Modelle sind beliebte Techniken für überwachte Lernaufgaben, sowohl für die Vorhersage von nominalen Klassen als auch von numerischen Werte. Für die Vorhersage numerischer Größen wurde an der Kombination dieser beiden beiden Verfahren zu „Modellbäumen“ gearbeitet, d.h. zu Bäumen, die lineare Regressionsfunktionen an den Blättern enthalten. LMT ist ein Algorithmus, der diese Idee für Klassifikationsprobleme adaptiert, wobei die logistische Regression anstelle der linearen Regression verwendet wird. Es wird einen stufenweisen Anpassungsprozess verwendet, um die logistischen Regressionsmodelle zu konstruieren, welche die relevante Attribute in den Daten auf natürliche Weise auswählen kann. [**?**]

1.2.7 DecisionStump (Weka 3.7)

Ein Entscheidungsstumpf ist ein maschinelles Lernmodell, das aus einem einstufigen Entscheidungsbaum besteht. Das heißt, es ist ein Entscheidungsbaum mit einem internen Knoten (der Wurzel), der direkt mit den Endknoten (seinen Blättern) verbunden ist. Ein Entscheidungsstumpf macht eine Vorhersage, die auf dem Wert nur eines einzigen Eingabe-Features basiert. Manchmal werden sie auch als „1-rules“ bezeichnet. Bei kontinuierlichen Merkmalen wird in der Regel ein Schwellenwert gewählt, und der Stumpf enthält zwei Blätter - für Werte unterhalb und oberhalb des Schwellenwerts. In seltenen Fällen können jedoch auch mehrere Schwellenwerte gewählt werden, und der Stumpf enthält dann drei oder mehr Blätter. [**?**]

Entscheidungsstümpfe werden oft als Komponenten (als „schwache Lerner“ oder „Basislerner“ bezeichnet) in Ensemble-Techniken des maschinellen Lernens wie Bagging und Boosting verwendet. Ein Algorithmus zur Gesichtserkennung nach Viola-Jones verwendet z.B. AdaBoost mit Entscheidungsstümpfen als schwache Lerner. [**?**]

1.2.8 J48Graft (Weka 3.7)

J48graft basiert auf J48, dem der C4.5 Algortihmus zugrunde liegt. J48graft nutzt den C4.5++ Algortihmus, welcher sich den Ansatz ‘all-tests-but-one-partition’ (ATBOP) zu Nutzen macht um den C4.5 Algortihmus zu verbessern.

Beim Entscheidungsbaum-Grafting werden Knoten zu einem vorhandenen Entscheidungsbaum hinzugefügt, um den Vorhersagefehler zu reduzieren. Der C4.5++ Algortihmus behält die Fehlerreduktion des ursprünglichen Grafting-Algorithmus bei und reduziert gleichzeitig die Rechenzeit und die Komplexität des abgeleiteten Baums drastisch. Bias/Varianz-Analysen zeigen, dass das ursprüngliche Transplantationsverfahren in erster Linie durch Varianzreduktion funktionierte, während das neue Technik sowohl die Verzerrung als auch die Varianz reduziert. [**?**]

1.2.9 BFTree (Weka 3.7)

Entscheidungsbäume sind potenziell leistungsfähige Prädiktoren und stellen die Struktur eines Datensatzes dar. Standard-Entscheidungsbaum-Lerner, wie z.B. C4.5, expandieren Knoten, während bei Best-First-Entscheidungsbaumlernern der „beste“ Knoten zuerst expandiert wird. Der „beste“ Knoten ist der Knoten, dessen Aufteilung zu einer maximalen Reduktion der Unreinheit (z.B. Gini-Index) unter allen für die Aufteilung verfügbaren Knoten führt. Der resultierende Baum ist derselbe, wenn er vollständig gewachsen ist, nur die Reihenfolge, in der er aufgebaut wird, ist unterschiedlich. In der Praxis werden einige Zweige eines vollständig expandierten Baums die zugrundeliegende Information in der Domäne nicht widerspiegeln. Dieses Problem wird als Overfitting bezeichnet und wird hauptsächlich durch verrauschte Daten verursacht. Pruning ist notwendig, um eine Überanpassung der Trainingsdaten zu vermeiden, und verwirft die Teile, die nicht vorhersagend für zukünftige Daten sind. Die Best-First-Knotenerweiterung ermöglicht die Untersuchung neuer Pruning-Techniken, indem die Anzahl der durchgeführten Expansionen auf Basis von Kreuzvalidierung verglichen werden. [**?**]

1.2.10 RandomTree (Weka 3.7)

Dieser Knime-Knoten ist zum Konstruieren eines Baums geeignet, der an jedem seiner Knoten K zufällig ausgewählte Attribute berücksichtigt. Der Algorithmus führt kein Pruning durch. Verfügt jedoch über eine Option, die eine Schätzung der Klassenwahrscheinlichkeiten auf der Basis eines Hold-out-Sets (Backfitting) ermöglicht.

1.2.11 RandomForest (Weka 3.7)

Random Forests sind eine Kombination von Baumprädiktoren, wobei jeder Baum von den Werten eines Zufallsvektors abhängt, der unabhängig und mit der gleichen Verteilung für alle Bäume im 'Wald' genutzt wird. Der Generalisierungsfehler für Wälder konvergiert gegen einen Grenzwert, wenn die Anzahl der Bäume im Wald groß wird. Der Generalisierungsfehler eines Waldes von Baumklassifikatoren hängt von der Stärke der einzelnen Bäume im Wald und der Korrelation zwischen ihnen ab. Die Verwendung einer zufälligen Auswahl von Merkmalen zur Aufteilung jedes Knotens führt zu Fehlerraten, die mit denen von Adaboost (Freund und Schapire) vergleichbar, aber robuster gegenüber Rauschen sind. Interne Schätzungen überwachen Fehler, Stärke und Korrelation, und diese werden verwendet, um die Reaktion auf die Erhöhung der Anzahl der bei der Aufteilung verwendeten Features zu zeigen. Interne Schätzungen werden auch verwendet, um die Wichtigkeit von Variablen zu messen. Diese Ideen sind auch auf Regression anwendbar. [**?**]

1.2.12 FT (Weka 3.7)

Funktionale Bäume erlauben eine enge Integration verschiedener Repräsentationssprachen für verallgemeinernde Beispiele. Es ist hervorzuheben, dass es Standardalgorithmen für topologische Transformationen auf Entscheidungsbäumen gibt, die hier nicht weiter besprochen, jedoch von Gama behandelt werden. [**?**]

1.3 Clusteranalysen

1.3.1 Was ist eine Clusteranalyse

1.3.2 kMeans Cluster

1.3.3 Dichtebasiertes Cluster

1.3.4 Hierachisches Cluster

2 KNIME Implementierung

2.1 Entscheidungsbäume

In Abbildung 2.1 ist zunächst der gesamte Workflow mit Metaknoten dargestellt. Unten befindet sich die Datenvorverarbeitung, oben die in Abbildung 2.2 dargestellte Eingabe der Variablen für Aufteilung der Trainingsmenge und die vorherzusagende Spalte.

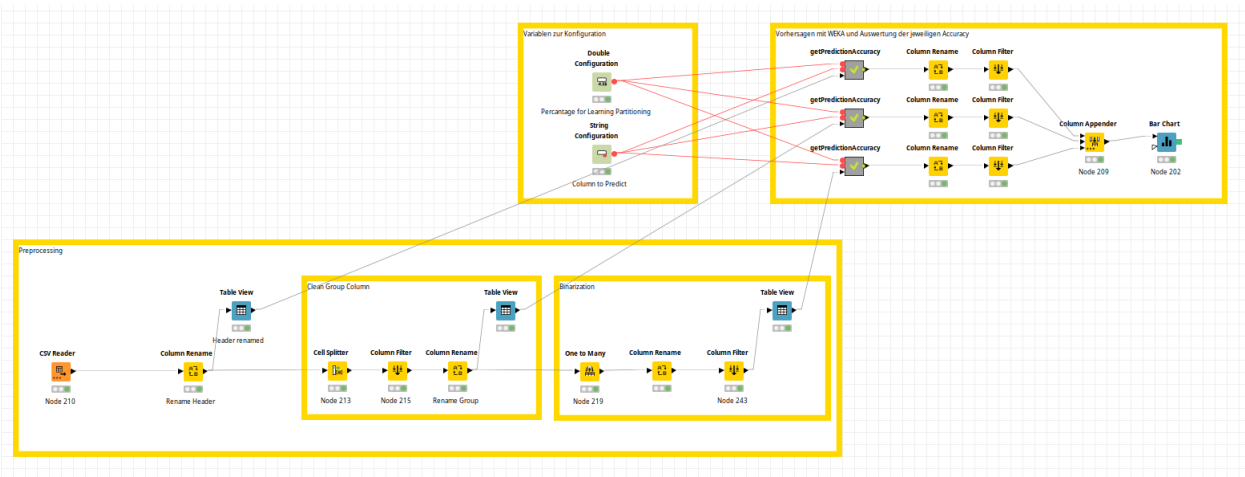

Abbildung 2.1: Aufbau des Workflows

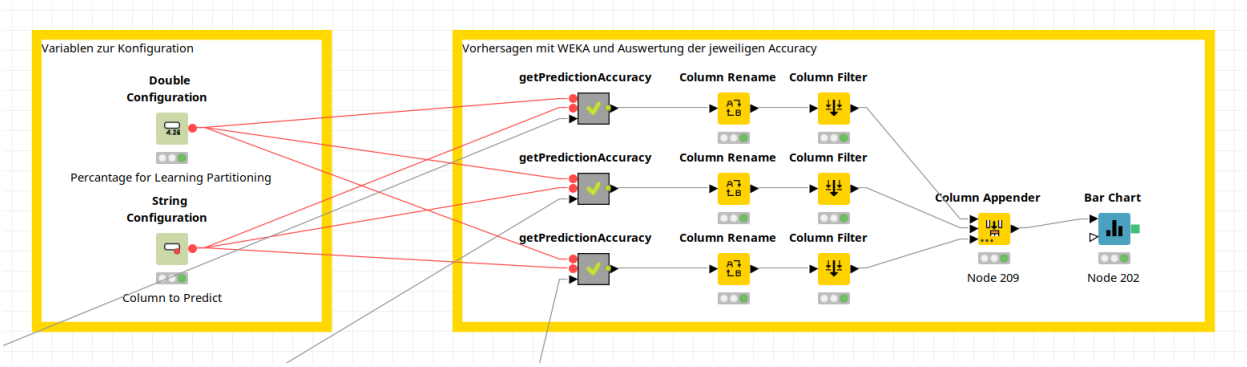

Abbildung 2.2: Ausschnitt des Workflows (Variableneingabe und Vorhersage)

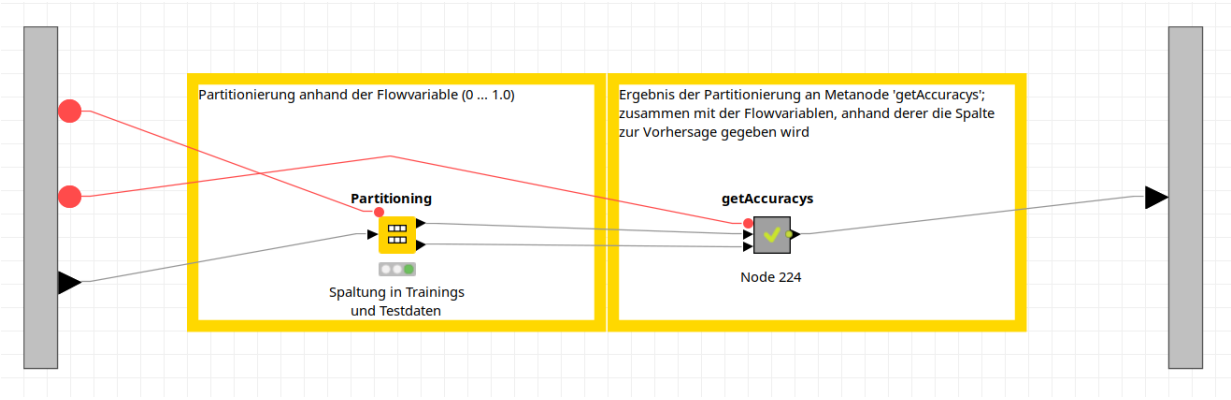

Abbildung 2.3: Inhalt des Metanodes ‘getPredictionAccuracy’

Hierbei wird dem Metaknoten ‘getPredictionAccuracy’ dreimal der verschieden aufbereitete Datensatz übergeben. Von ihm ausgegeben werden, die Werte des ‘Accuracy’ Feldes für jeden in Kapitel 1.2 genannten Entscheidungsbaum-Knoten. Die Trainingsmengen sind hierbei immer gleich (vgl. Abblidung 2.3). Zusammengefügt und visualisiert werden die Ergebnisse in einem Barchart.

Der in Abbildung 2.3 gezeigte Metaknoten ‘getAccuracys’ ist in Abbildung 2.5 genauer dargestellt. Er beinhaltet für jeden in Kapitel 1.2 genannten Entscheidungsbaum einen eigenen Metaknoten dessen Inhalt analog zu dem in Abbildung 2.4 ist. Übergeben werden hierbei die Trainings und die Testmenge, sowie der Name der vorherzusagende Spalte. Der Knoten ‘Scorer’ erzeugt unteranderem den zu vergleichenden Wert ‘Accuracy’, welcher durch einen Spalten- und Reihenfilter extrahiert und umbenannt wird.

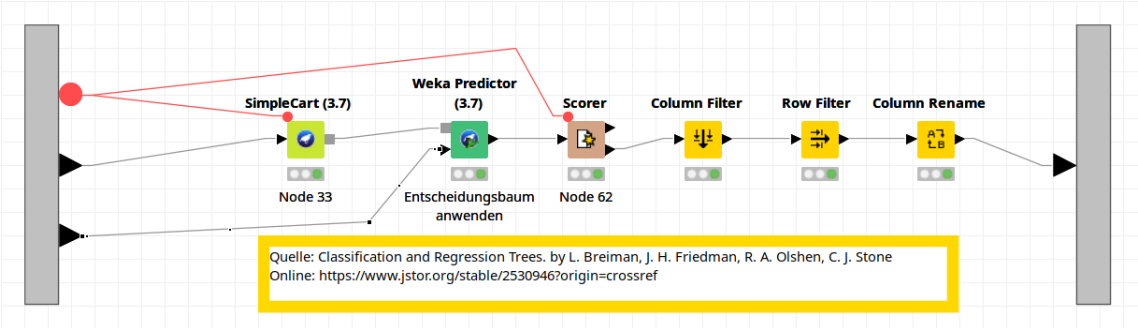

Abbildung 2.4: Inhalt des Metanodes ‘getAccuracyWekaSimpleCart’

2.2 Clusteranalyse

2.2.1 Relevante Knoten

2.2.2 kMeans Cluster Workflow

2.2.3 Dichtebasierters Cluster Workflow

2.2.4 Hierarchisches Cluster Workflow

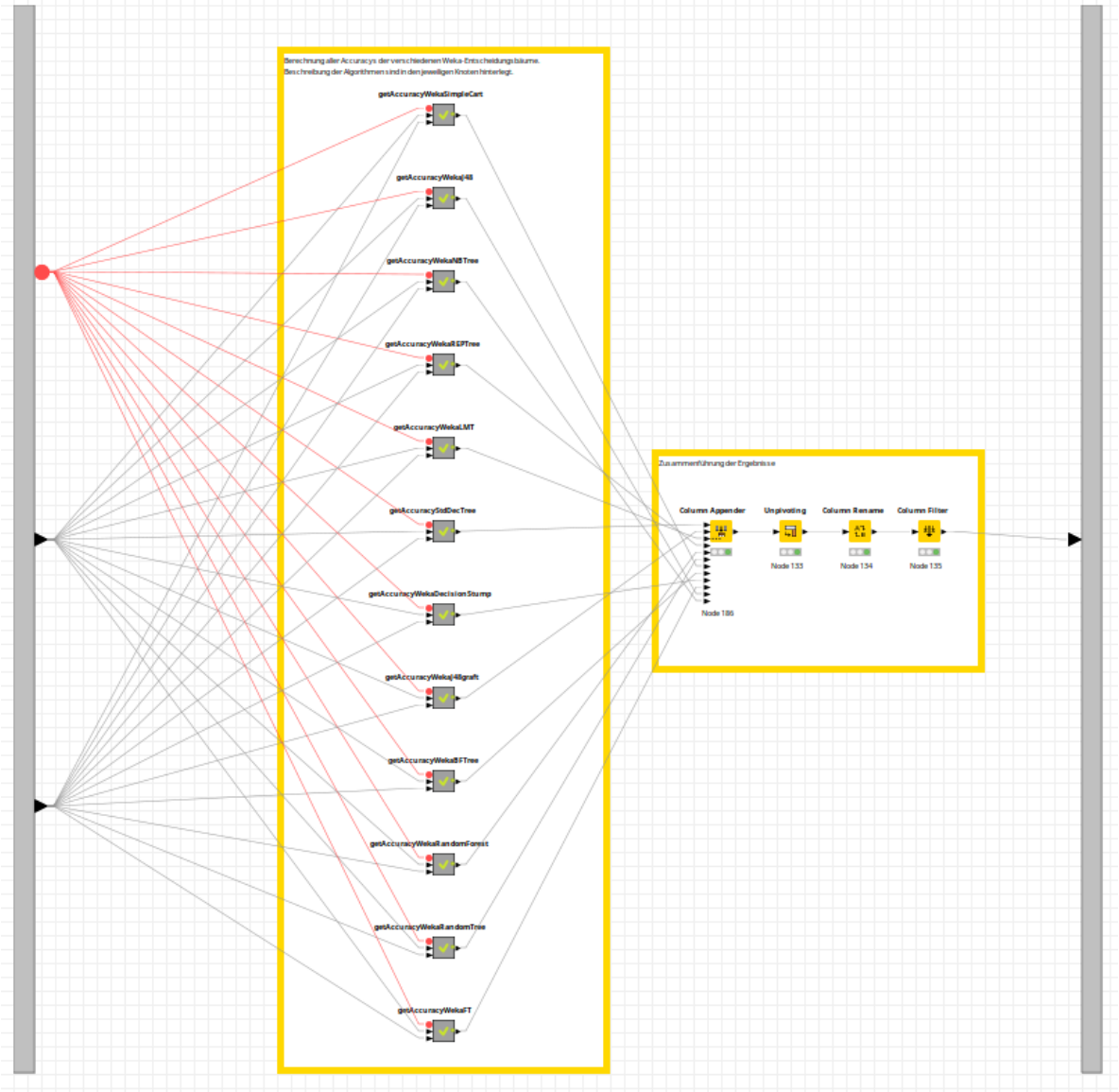

Abbildung 2.5: Inhalt des Metanodes ‘getAccuracys’

Abbildungsverzeichnis