HOCHSCHULE FÜR TECHNIK ZÜRICH

UBS

Diplomarbeit

Ein Prototyp für die Segmentierung und Klassifizierung von Börsenaufträgen

Student	Fabian Merki
Betreuer	Klaus Wolfertz
Experte	Christoph Burkhardt

Inhaltsverzeichnis

Zusammenfassung

In meiner Diplomarbeit habe ich Möglichkeiten untersucht, die Ausführungsqualität von Wertschriftenaufträgen vorherzusagen.

Um dieses Ziel zu erreichen, bereitete ich Wertschriftenaufträge mit einer selbst entwickelten Software auf. Mittels Data Mining teilte ich die Wertschriftenaufträge in Ausführungsqualitätsgruppen ein und entwickelte mehrere Vorhersagemodelle. Dazu verwendete ich zwei Data Mining-Tools und verschiedene Algorithmen. Um das optimale Vorhersagemodell zu bestimmen, validierte ich die verschiedenen Modelle und verglich die Resultate miteinander. Mit Hilfe des entwickelten Software-Prototyps lässt sich schliesslich die Ausführungsqualität von neuen Wertschriftenaufträgen anhand eines Modells theoretisch vorhersagen.

Allerdings sind die Vorhersagen der entwickelten Modelle nicht genau genug, um im produktiven Umfeld die Verarbeitungsentscheidungen lediglich anhand der Vorhersagen zu treffen. Zwei businessrelevante Trends konnten jedoch erkannt werden: Ausführungen in Mailand via einen bestimmten Broker und Ausführungen in Athen erhalten eine etwas grössere Preisabweichung. Des Weiteren konnte ich eine interessante Feststellung betreffend Verteilung der Auftragsgrössen in Franken machen: Die Form der Verteilung entspricht auf einer logarithmischen Skala der einer Gauss’schen Normalverteilung.

Meiner Meinung nach liessen sich mit einer verbesserten Datenaufbereitung und einer grösseren Lernmenge für die Data Mining-Algorithmen zuverlässigere Vorhersagemodelle entwickeln und weitere interessante Erkenntnisse gewinnen. Allenfalls könnte eine andere Definition der Ausführungsqualität zu einer verbesserten, aussagekräftigeren und produktiv einsetzbaren Vorhersage führen. Das Potenzial einer genauen Vorhersage ist sehr gross, da damit Kostenersparnisse und Risikominimierungen möglich sind.

Abstract

In my diploma I studied the possibilities of predicting the execution quality of securities orders.

To reach this goal I preprocessed securities orders with a specially developed tool. Using data mining approaches, I separated the orders into groups of different execution quality and then developed prediction models. As part of this process two data mining tools and several algorithms were used. I validated the different models and compared the results to build the optimal prediction model. The execution quality of securities orders can be theoretically predicted using the software prototype according to a model.

The accuracy of the prediction is not good enough to be used solely in making processing decisions within a productive environment. However, two business relevant trends were recognized: executions at the exchange in Milano via a specific broker and in general executions at the exchange in Athens, both receive a slightly larger price difference. Another interesting discovery is the distribution of turnover: the form of the distribution on a logarithmic scale matches the Gaussian distribution function.

In my opinion it would be possible to create better prediction models and gain valuable knowledge if the preprocessing of the data were optimised and a larger set of learning data were used. Additionally, a preassigned definition of the execution quality could lead to an improved and more significant prediction which in turn could be used in a productive environment. More precise predictions have the potential for large cost savings and risk optimizations.

1 Einleitung

1.1 Ausgangslage

Die UBS Investment Bank verarbeitet pro Tag hunderttausende von Börsenaufträgen. Bei der Ausführung dieser Aufträge steht das Interesse des Kunden im Vordergrund. Es gibt zahlreiche Gründe, weshalb nicht alle Aufträge automatisch an die Börse geschickt werden. Die heutigen (Stand Februar 2007) Regeln für das manuelle Verarbeiten der Aufträge berücksichtigen im Wesentlichen nur:

- Sehr grosse Aufträge (zum Beispiel Marktwert > CHF 500’000) müssen in Tranchen geschickt werden, da sie sonst den Markt zu stark beeinflussen würden.
- Aufträge mit falschen Limiten (zum Beispiel Limite und Quantität offensichtlich vertauscht) sollen manuell behandelt werden.
- Aufträge mit speziellen Kundenwünschen, welche bei der Ausführung berücksichtigt werden müssen.

Zurzeit werden diese Aufträge anhand von Regeln automatisch verarbeitet oder angehalten, um manuell verarbeitet zu werden. Aufträge werden jedoch mit ungenügender Genauigkeit oder gar nicht klassifiziert. Dies führt zu folgenden Problemen:

- Wegen zu vereinfachten Regeln werden zum Teil Aufträge unnötigerweise als „manuell“ klassifiziert und behandelt. Manuell klassifizierte Aufträge sind solche, die durch einen qualifizierten Händler geprüft, angepasst sowie verarbeitet werden. Bei dem Transaktionsvolumen, welches über die UBS Investment Bank abgewickelt wird, müssen manuelle Interventionen möglichst gering gehalten werden, um die Kosten tief zu halten.
- Da gewisse Informationen in den Regeln nicht berücksichtigt werden, werden manchmal Aufträge automatisch an die Börse geschickt, dabei hätten sie vielleicht durch manuelle Bearbeitung optimierter ausgeführt werden können.
- Fehlende Regeln bei der Unterteilung in Gruppen führen zu einer gröberen Analyse. Eine feinere Unterteilung würde die Analyse vereinfachen.

Die bisherige Lösung soll verfeinert werden, eine optimierte Verarbeitung ermöglichen und dank einer Klassifizierung die Überwachung vereinfachen.

Aus diesem Grund soll eine neue Lösung für das frühzeitige Erkennen von möglichen Problemfällen entwickelt werden.

1.2 Ziel der Arbeit

In dieser Arbeit soll geprüft werden, ob es möglich ist, Wertschriftenaufträge mit Data Mining-Methoden[1] anhand einer dafür getroffenen Definition von „Best Execution“ [2] zu segmentieren und automatisch zu klassifizieren, um mögliche Probleme vorzeitig erkennen und entsprechend handeln zu können.

Die Segmentierung soll anhand verarbeiteter Aufträge und zusätzlicher Daten (Preisinformationen etc.) entwickelt werden. Es ist zu prüfen, welche Attribute sich für eine robuste Segmentierung eignen.

Den verarbeiteten Aufträgen sollen mittels dieser Segmentierung Qualitätsklassen zugeordnet werden. Es soll ein Klassifizierungsmodell entwickelt werden, welches die zu erwartende Qualität für neue Aufträge (ohne Abschlussinformationen) vorhersagt.

Dieses Vorhersagemodell soll einer Überwachungskomponente (Metrics Service) und der graphischen Benutzerapplikation (GUI) zur Verfügung stehen, um die Datenauswahl im Überwachungs-GUI zu vereinfachen. Dazu ist ein Klassifizierer-Prototyp zu erstellen, welcher neue Aufträge (unbekannte Instanzen) automatisch klassifiziert.

Es soll geprüft werden, wie gut sich diese Klassifizierung zur Überprüfung der Zieldefinition von Best Execution eignet. Zusätzlich soll eine Aussage gemacht werden, ob und wie die in der Arbeit verwendeten Verfahren in Zukunft für die Auftragsverarbeitung, Kostenverrechnung, mögliche Kostenersparnisse und weitere Anwendungen verwendet werden könnten.

1.3 Abgrenzung

Der Fokus der Arbeit liegt in der Analyse der Daten und im Design von Segmentierungs- und Klassifikationsmodellen. Erweiterungen und Verbesserungen des existierenden Data Warehouse sowie an der Benutzerapplikation (TOPAZ GUI) sind nicht Inhalt dieser Arbeit.

Die Berechnung der Ausführungszeit eines Auftrages ist in Realität sehr komplex respektive nicht vollständig ohne sehr grossen Aufwand möglich. In dieser Arbeit werden Aufträge an der Hongkonger Börse[3], telefonische Aufträge, Aufträge vor März 2007, nicht in TOPAZ erstellte oder limitierte (mit Auftragslimiten) nicht behandelt. Es wird eine vereinfachte Berechnung der Ausführungszeit ohne Berücksichtigung der Auktionen[4] verwendet.

[1] Die Diplomarbeit habe ich nach den Regeln der neuen deutschen Rechtschreibung verfasst (Rechtschreib-Duden, 24. Auflage). Bei der Wortzusammensetzung englischer Fachausdrücke habe ich auf den Kopplungsbindestrich verzichtet (Data Mining); gängige Komposita habe ich durchgekoppelt (Cluster-Analyse).
[2] Der Begriff „Best Execution“ wird in Kapitel 2.1 für diese Arbeit definiert. Die UBS Investment Bank muss sicherstellen, dass Wertschriftenaufträge bestmöglich ausgeführt werden.
[3] In Hong Kong wird die Börse über den Mittag geschlossen.
[4] Die verschiedenen Börsen haben zum Teil sehr unterschiedliche Markphasenmodelle (siehe Anhang C).

1.4 Vorgehen

Die Abbildung 1 zeigt eine schematische Übersicht des Vorgehens, welches für diese Arbeit gewählt wird.

Kapitel 2 befasst sich mit der Analyse der Data Mining-Aktivitäten in der UBS Investment Bank, dem System TOPAZ und dessen Datenquellen.

In Kapitel 3 werden das Vorgehen fürs Data Mining und die Datenaufbereitung beschrieben.

In Kapitel 4 wird die Segmentierung („1. Segmentierung“) der Wertschriftenaufträge anhand der Abschlussinformationen (*x* und *y*) behandelt. Dabei werden Segmentierungsmodelle entwickelt, welche neue Klassen (*k*) für bekannte Instanzen bestimmen.

In Kapitel 5 werden Klassifizierungsmodelle („2. Klassifizierung“) entwickelt, welche auf den Auftragsdaten (*a* und *b*) und der Klasse (*k*) aus Kapitel 4 basieren.

In Kapitel 6 werden Data Mining-Experimente durchgeführt und die Vorhersagemodelle validiert („3. Validierung“). Dabei werden für abgeschlossene Wertschriftenaufträge mittels Segmentierungsmodell die effektiven Klassen bestimmt. Die nun bekannten Instanzen (Klasse bekannt durch Segmentierung) werden neu als unbekannte Instanzen bei der Klassifizierung betrachtet und die Klasse mittels Klassifizierungsmodell vorhergesagt. Die beiden Klassenzuordnungen werden nun miteinander verglichen, um zu prüfen, wie gut die Vorhersage funktioniert.

In Kapitel 7 wird ein Klassifizierer-Prototyp („4. Klassifizierer“) entwickelt, welcher für neue Instanzen anhand des zuvor erstellten Modells die Klasse vorhersagt.

In Kapitel 8 werden die Erkenntnisse reflektiert.

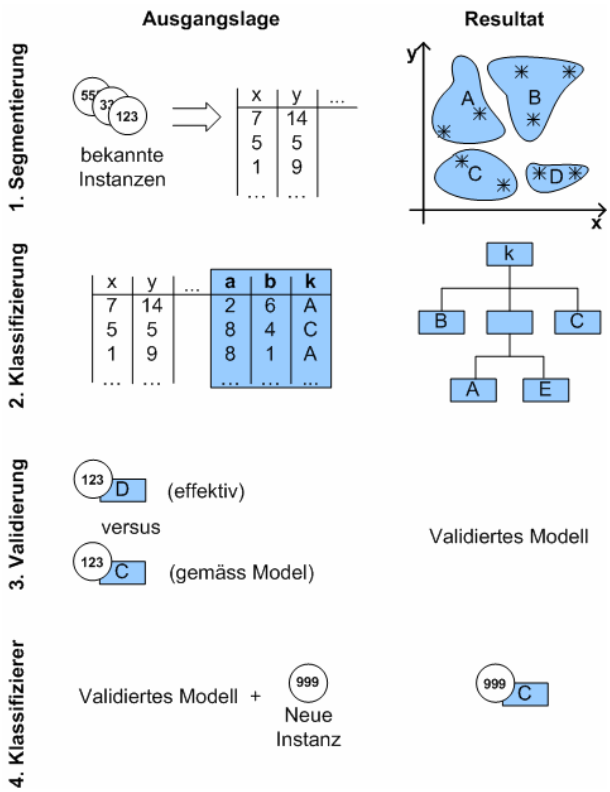


Abbildung 1 – Vorgehen

2 Analyse

2.1 Data Mining in der UBS Investment Bank (Order Monitoring)

Bereits heute (Stand Februar 2007) werden zahlreiche Reports und Statistiken durch Business-User erstellt, um den Auftragsfluss zu überwachen. Zum Beispiel werden die Anzahl der Aufträge nach Börsenplatz, Währung, Broker usw. ausgewertet sowie graphisch aufbereitet, um die Auftragsverarbeitung zu überwachen und um das Management bei Verhandlungen mit Partnern zu unterstützen.

Das Extrahieren des Wissens aus diesem grossen Datenbestand, wie Data Mining in [HanKamber2001, S. 5] definiert wird, geschieht zurzeit manuell. Es handelt sich nur um triviales Wissen, das direkt aus OLAP Reports herauszulesen ist. Es ist nicht klar, ob und wie gut die Erkennung von interessanten Mustern im Sinne von Data Mining zurzeit mit dem aktuellen Vorgehen erreicht wird. Da der Mensch aber über eine sehr gute Mustererkennung verfügt, ist davon auszugehen, dass dieses Ziel teilweise erreicht wird. Eine maschinelle Mustererkennung findet zurzeit im Bereich der Wertschriftenauftragsabwicklung nicht statt.

2.2 Begriffe des Order Routings

Die Begriffe des Order Routings sind im Glossar in Anhang D kurz erklärt. Nachfolgend werden die für diese Arbeit wichtigsten Begriffe analysiert und definiert.

2.2.1 Ausführungslatenz

Die Zeit, bis ein Wertschriftenauftrag vollständig ausgeführt ist, wird als Ausführungslatenz bezeichnet. Die Zeit wird jedoch nicht einfach vom Erfassungszeitpunkt bis zum Zeitpunkt der letzten Ausführung gemessen, sondern es wird nur die Zeit gemessen, in welcher der Auftrag unter Berücksichtigung von Feiertagen, Wochenenden und Börsenöffnungszeiten ausgeführt werden konnte. Die Ausführungslatenz wird je nach Auftragsart (unlimitiert, limitiert usw.) unterschiedlich berechnet. Bei Aufträgen mit Limiten muss der Kursverlauf in die Berechnung miteinbezogen werden. Abbildung 2 zeigt die verschiedenen Komponenten der Berechnung in einem Zeitdiagramm. Die zu messende Zeit entspricht der Schnittmenge der einzelnen Komponenten.

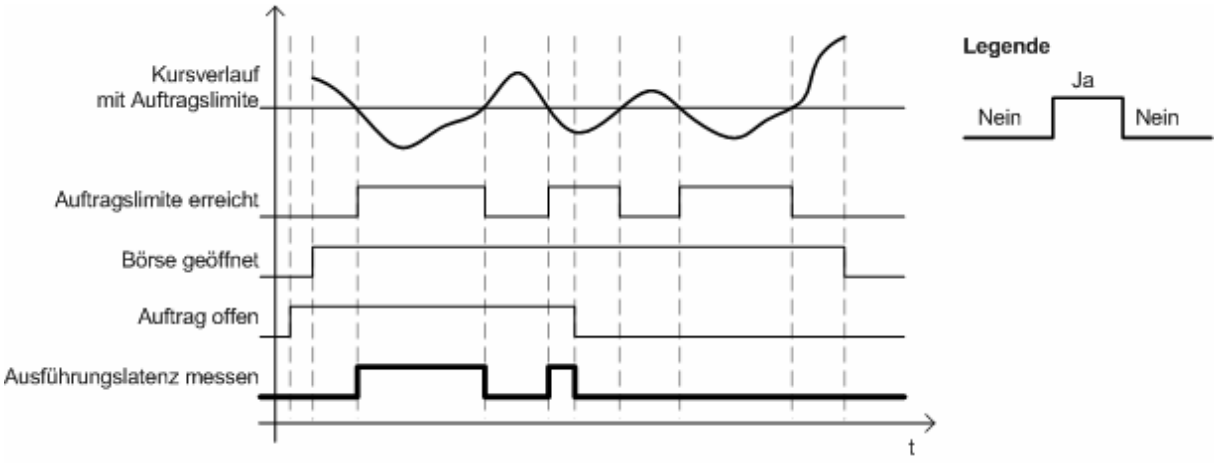


Abbildung 2 – Ausführungslatenz eines limitierten Börsenauftrages

Die Marktmodelle der verschiedenen Börsen und nach Art der Wertschriften (siehe Anhang C) unterscheiden sich stark voneinander. In der Diplomarbeit wird ein vereinfachtes, allgemeines Marktmodell angenommen: Aufträge können von der Börsenöffnung bis zum Börsenschluss ausgeführt werden.

2.2.2 Preisverbesserung (Price improvement)

Die Preisverbesserung ist die Differenz zwischen dem Durchschnittspreis der Ausführung und dem Durchschnittspreis am Markt während der Laufzeit des Auftrages.

2.2.3 Best Execution

Ein wichtiges Ziel der UBS Investment Bank ist, Wertschriftenaufträge bestmöglich auszuführen. Dies wird als „Best Execution“ bezeichnet und in [BestExecutionWikipedia2007] folgendermassen definiert:

> ***Best Execution*** *refers to the obligation of an investment services firm (such as a stock broker) executing orders on behalf of customers to ensure that the prices those orders receive reflect the optimal mix of price improvement, speed and likelihood of execution. Brokers with customer orders are obligated to send orders to venues with the optimal "best execution stats."*

“Best Execution” ist ein Begriff für sehr vieles, weshalb er im Rahmen dieser Arbeit folgendermassen eingeschränkt und klar definiert wird:

„Best Execution“ ist der Mix aus Preisverbesserung und Ausführungszeit (siehe Abbildung 3).

Der linke untere Bereich in Abbildung 3 stellt ein optimales Verhältnis zwischen Preisverbesserung und Ausführungszeit dar. Grundsätzlich gilt: Je grösser respektive höher die Preisverbesserung ist, desto besser oder schlechter ist der Preis für den Kunden; je kürzer die Ausführungszeit ist, desto besser wurde der Auftrag ausgeführt. Verlängert sich die Ausführungszeit jedoch massiv, erhöht sich auch das Risiko, da grössere Preisschwankungen zu erwarten sind, was am Schluss zu einem sehr guten oder sehr schlechten Preis führen kann. Das Wissen des Händlers spielt hier eine grosse Rolle, da er das Risiko managen muss.

Dieser Mix kann folgendermassen für einen Wertschriftenauftrag an verschiedenen Märkten und zu verschiedenen Zeiten (Varianten = blaue Punkte) dargestellt werden:

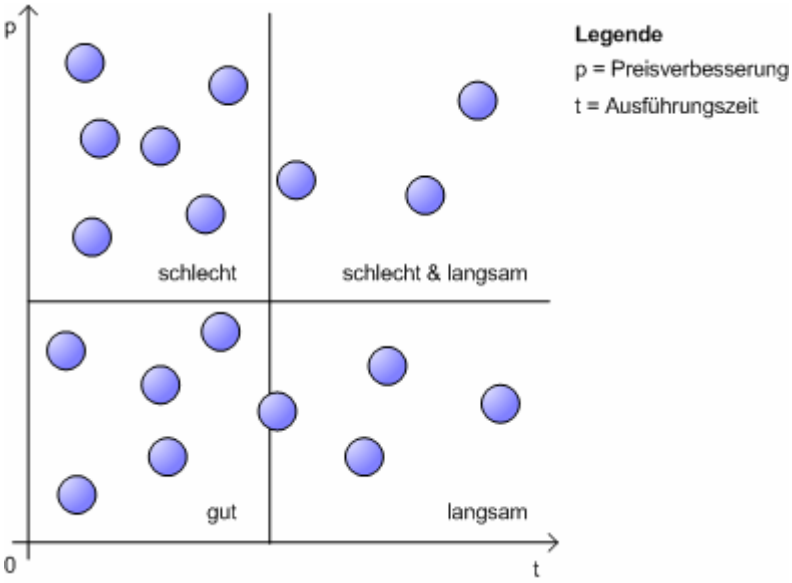


Abbildung 3 – Mix aus Preisverbesserung und Ausführungszeit

Die beiden Achsen lassen sich folgendermassen normalisieren, wodurch Abbildung 3 allgemeiner dargestellt werden kann. Für jeden Auftrag werden zwei Werte berechnet:

- **Preisverbesserungsratio**
  Preisverbesserung in Prozent während der Laufzeit des Auftrages gegenüber VWAP.
- **Ausführungszeitratio**
  Ausführungszeit im Verhältnis zur Ausführungszeit des dreifachen Volumens.
  Dies entspricht der Strategie „33% Partizipation des Volumens“, welche in der Bank häufig angewendet wird.

Abbildung 4 zeigt verschiedene ausgeführte Aufträge mit normalisierten Ausführungs- und Preisverbesserungsachsen. Grösse und Form der Figuren stellen die einzelnen Aufträge mit unterschiedlichen Auftragsgrössen, Erfassungszeitpunkten, Wertschriften usw. dar.

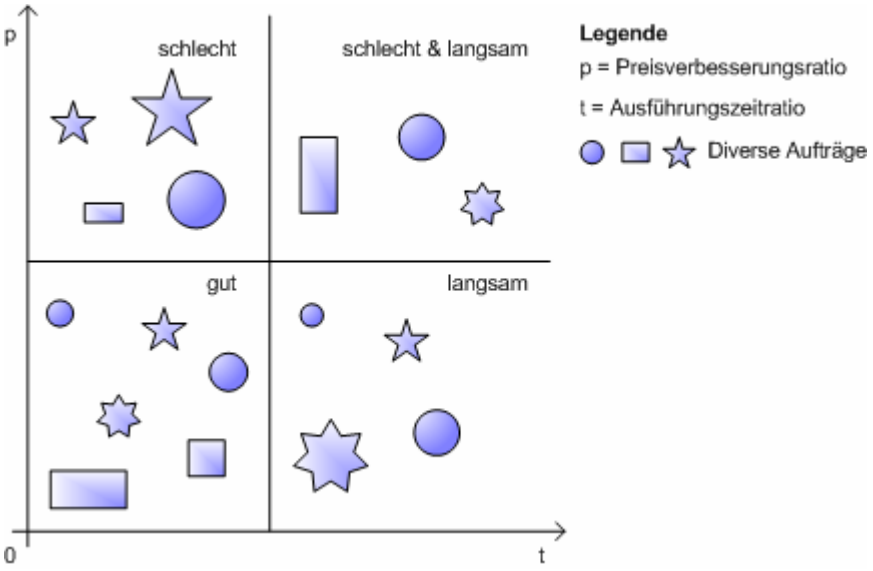


Abbildung 4 – Mix aus Preisverbesserung und Ausführungszeit (normalisiert)

Dank der Normalisierung von Preisverbesserung und Ausführungszeit als Preisverbesserungs- und Ausführungszeitratio ist nun ein qualitativer Vergleich verschiedener Auftragsabschlüsse möglich. Die triviale Aufteilung in gut, schlecht und langsam dient nur dem Verständnis der Abbildung 4 und ist etwas willkürlich gewählt. Falls Gruppen von ähnlichen Abschlüssen gemäss den beiden Achsen existieren, könnten diese Gruppen vielleicht vorhergesagt werden. Ob und wie gut diese Vorhersage funktioniert, soll in dieser Arbeit untersucht werden.

2.3 TOPAZ – Order Routing System der UBS Investment Bank

Das System TOPAZ ist für das Routing von Wertschriftenaufträgen zuständig. Wie in Abbildung 5 ersichtlich, können Börsenaufträge von Kunden elektronisch oder telefonisch an die Börse platziert werden. Diese werden in gewissen Situationen durch das System angehalten und müssen von Executors manuell oder elektronisch an eine Börse platziert werden. Nicht angehaltene Aufträge werden automatisch an die Börse geschickt. Wenn ein Handel an der Börse stattfindet, wird die Ausführung an den Kunden über den gleichen Weg zurückgeschickt. Analog verhalten sich Annullierungen der Aufträge.

Die Benutzeroberfläche (GUI) von TOPAZ ermöglicht eine graphische Überwachung der Wertschriftenaufträge. Diese werden in Diagrammen nach bestimmten Kriterien dargestellt. Erfahrene Benutzer sind damit in der Lage, Probleme zu erkennen, da sie zum Beispiel Ausreisser auf einen Blick oder Anomalien im Vergleich zu anderen Tagen erkennen können.

Abbildung 5 zeigt ein vereinfachtes Modell für Wertschriftenaufträge und deren Ausführung vom Kunden bis zur Börse und zurück.

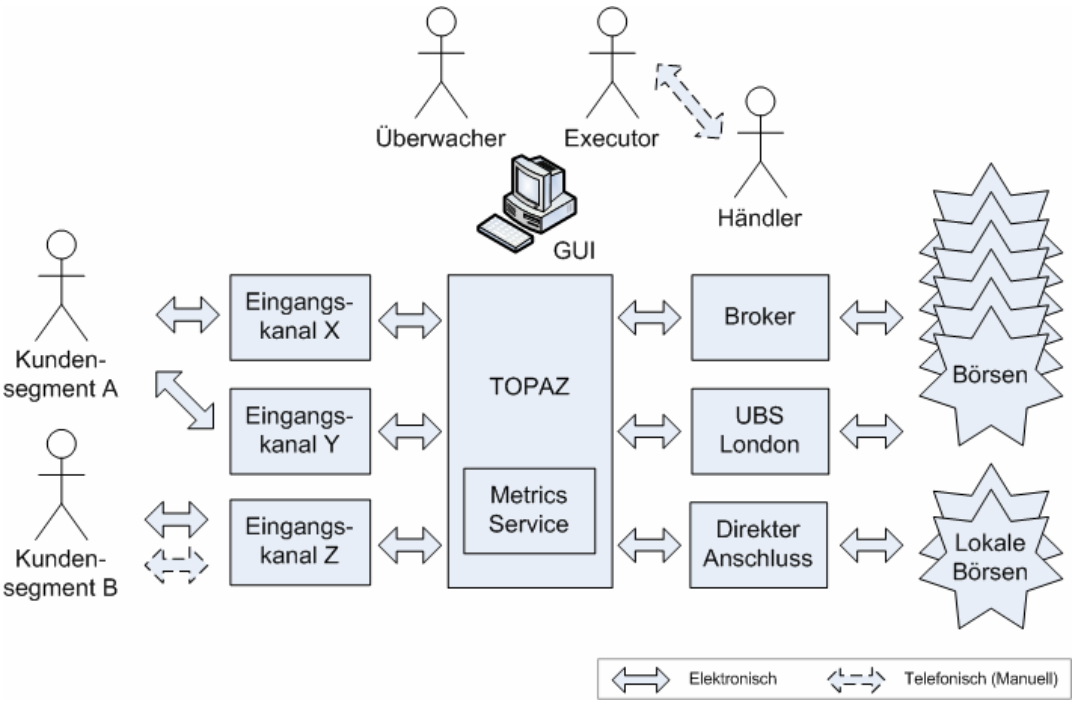


Abbildung 5 – TOPAZ-Systemübersicht

TOPAZ besteht aus einem Order Management-System (Auftragsverwaltungssystem) sowie dem Trade Support-System (Abrechnungssystem), an welchen Ein- und Ausgangskanäle (diverse Eingangskanäle und Broker, UBS London, direkter Börsenanschluss) angeschlossen sind. Der Metrics-Service ist ein Teil von TOPAZ, welcher in near-time Statistiken für die Überwachung der Wertschriftenaufträge berechnet.

Das Ziel dieser Arbeit ist in Zukunft neue Aufträge bei Erhalt durch den Metrics Service zu klassifizieren. Somit kann die Vorhersage der Qualität der Ausführung den Überwachern und Executoren angezeigt werden, so dass aktiv mögliche Probleme vorzeitig erkannt und gelöst werden können. Zum Beispiel könnte ein Auftrag an eine andere Börse geschickt oder in kleine Aufträge aufgeteilt werden. Die benötigten Anpassungen im TOPAZ GUI sind nicht Teil dieser Diplomarbeit.

2.4 Daten und Datenquellen

Nachfolgend werden die Daten und Datenquellen, welche für das Data Mining zur Verfügung stehen, beschrieben.

2.4.1 Auftragsdaten

Die Auftragsdaten werden in TOPAZ in einer Datenbank („OMS Daily“) dauerhaft gespeichert. Fünf Tage nach Verarbeitung eines Auftrages gelangen diese in die Archivdatenbank „OMS History“, welche ein fast identisches Datenmodell wie „OMS Daily“ enthält. In der Archivdatenbank existiert in den Auftrags- und Ausführungstabellen ein zusätzliches Attribut für die Session (vergleichbar mit Datum der Archivierung), wodurch pro Objekt mehrere Versionen des Auftrages gespeichert sind. Das Abrechnen und Verbuchen der Aufträge erfolgt über das System Trade Support, welches die Auftrags- und Abrechnungsdaten in die Datenbank „TS Daily“ mit einem adaptierten Datenmodell speichert und in „TS History“ archiviert.

Die Datenbank „TS Daily“ eignet sich sehr gut als Datenbasis, da dort alle Aufträge als Snapshot (nur die letzte Version) gespeichert sind. Dadurch vereinfachen sich Datenbankabfragen. Diese Quelle wird in dieser Arbeit als Auftragsdatenbank bezeichnet.

Die Datenbasis enthält einige fehlerhafte Attribute, welche zum Beispiel durch Softwareänderungen ohne Migration alter Daten entstanden sind. Deshalb ist ein Säubern dieser Daten nötig, welches in Kapitel 3.3 beschrieben wird.

2.4.2 Metriken der Auftragsdaten

Informationen, wie zum Beispiel die Ausführungslatenz des Brokers, werden als Metriken bezeichnet.

Diese Werte werden durch den Metrics Service berechnet und in der Auftragsdatenbank gespeichert. Da der Service erst seit Juni 2007 als Pilot in dem produktiven System installiert ist, stehen die Metriken nur für Aufträge ab diesem Zeitpunkt zur Verfügung. Dadurch reduziert sich die Menge der Testdaten. Sobald der Metrics Service um eine In-/Out-of-limit Minding Funktionalität erweitert wird und genügend Testdaten zur Verfügung stehen, können limitierte Aufträge in die Analyse mit einbezogen werden. In dieser Arbeit können deshalb nur alte Aufträge ohne Limit („bestens“) berücksichtigt werden, da nur bei diesen die Berechnung der Latenz zum heutigen Zeitpunkt noch möglich ist.

2.4.3 Referenzdaten

Informationen, wie zum Beispiel der Firmenname einer Aktie (UBS), die Mindeststückelung (kleinste handelbare Menge: 1 Stück), das Segment (Finanzunternehmen), Börsenöffnungszeiten und Feiertage, werden als Referenzdaten bezeichnet.

Diese Daten können in TOPAZ über Webservices abgefragt werden.

Die Art des Marktanschlusses (Direktanschluss, elektronisch, manuell) kann auf der UBS-Intranetseite „Trading Place Information“ (siehe Abbildung 6) abgerufen werden.

Trading Place	Trading Hours Winter	Trading Hours Summer	Market Access
SWX (SWX)	09:00 to 17:30	09:00 to 17:30	Direct Link
VTX (SMI only) (VXi)	09:00 to 17:30	09:00 to 17:30	Direct Link
Vienna (VIE)	09:20 to 17:30	09:20 to 17:30	Electronic

Abbildung 6 – Auszug aus „Trading Place Information“

2.4.4 Marktdaten

Informationen über den Bid-/Ask-Preis, das Bid-/Ask-Volumen einer Aktie (respektive einer andern Wertschriftenart) an einem Markt zu einem bestimmten Zeitpunkt werden als Marktdaten bezeichnet.

Marktdaten können historisch von einem MDS (zum Beispiel von Vhayu[5] mit Datenquelle von Reuters) bezogen werden.

Eine andere Bibliotheksfunktion („Trade analytics“) liefert historisch statistische Informationen zu einer Aktie an einem Markt zu einem bestimmten Zeitpunkt respektive Zeitraum, wie zum Beispiel das durchschnittliche Volumen über 30 Tage.

2.4.5 Data Warehouse

Die Auftragsdaten mit einigen Referenzdaten, aber ohne Marktdaten und Metriken werden fortlaufend ins Data Warehouse übertragen. Dort stehen diese für Auswertungen durch die Business Users zur Verfügung. Datenextrakte können vom Data Warehouse-Team bei Bedarf bezogen werden.

2.4.6 Wahl der Quellen

Ein Data Warehouse wäre die ideale Quelle für das Data Mining. Da aber die Auftragsdaten direkt ins Data Warehouse übernommen werden, ohne dass diese gesäubert werden, müssten diese (wie in Kapitel 2.4.1 beschrieben) aufbereitet werden. Zudem sind nicht alle fürs Data Mining benötigen Attribute vorhanden, weshalb andere Quellen integriert werden müssten.

In dieser Arbeit werden die Auftragsdaten mit den Markt- und Referenzdaten integriert, um eine Datengrundlage für die Segmentierung zu erhalten. Die benötigten Metriken werden nachträglich berechnet, da noch nicht genügend Daten des Metrics Service verfügbar sind. Diese Datenquellen genügen den Anforderungen der Segmentierung und sind einfach zu integrieren.

[5] http://www.vhayu.com/

3 Data Mining

In diesem Kapitel werden wichtige Data Mining-Begriffe beschrieben und ein Vorgehen für Data Mining entwickelt. Für die nachfolgende Arbeit werden die Daten aufbereitet und in einem Mini Data Warehouse gespeichert.

3.1 Wichtige Begriffe

3.1.1 Instanz

Instanzen (siehe [InstanzWikipedia2007]) stellen abstrakte oder physische Dinge respektive Objekte in Data Mining dar.

Ein Wertschriftenauftrag wird in dieser Arbeit als eine Instanz betrachtet.

3.1.2 Attribut

Die Eigenschaften der Instanzen werden Attribute (siehe [AttributWikipedia2007]) genannt.

Die Auftragsgrösse ist zum Beispiel ein Attribut eines Wertschriftenauftrages und 1000 Stück ein Attributwert.

3.1.3 Cluster-Analyse

Als Cluster-Analyse oder Segmentierung bezeichnet [HanKamber2001, S. 25] den Prozess, Instanzen in Klassen von ähnlichen Instanzen einzuteilen. Das Ziel ist: Instanzen in einem Cluster sind sich ähnlich, aber sie sind möglichst unterschiedlich zu Instanzen in einem andern Cluster. Der Ähnlichkeitsgrad kann zum Beispiel als Euklidischer Abstand (Pythagoras im n-dimensionalen Raum) der Attributswerte zwischen zwei Instanzen gemessen werden.

3.1.4 Robuste Segmentierung

Robust bedeutet, dass das Clustering wegen kleiner Datenfehler in den Instanzen nicht überproportional beeinflusst wird und dass das Segmentierungsmodell für längere Zeit (mind. 1 Monat) für zukünftige Instanzen gültig bleibt. Ein erneutes Herleiten des Segmentierungsmodells mit zukünftigen Instanzen soll ein möglichst identisches Modell liefern.

3.1.5 Klassifikation

Als Klassifikation bezeichnet [HanKamber2001, S. 24] das Suchen nach Vorhersagemodellen für unbekannte Instanzen. Bei der Klassifikation werden den Instanzen anhand der Attributswerte und einem Modell entsprechende Klassen zugeordnet.

3.2 Vorgehen

Data Mining ist ein dynamischer, iterativer Prozess (siehe [HanKamber2001, S. 5-7] und [HocheKorgelWrobel2007, S. 10]). Für die Segmentierung wurde folgendes Vorgehen (siehe Abbildung 7) entwickelt und angewendet:

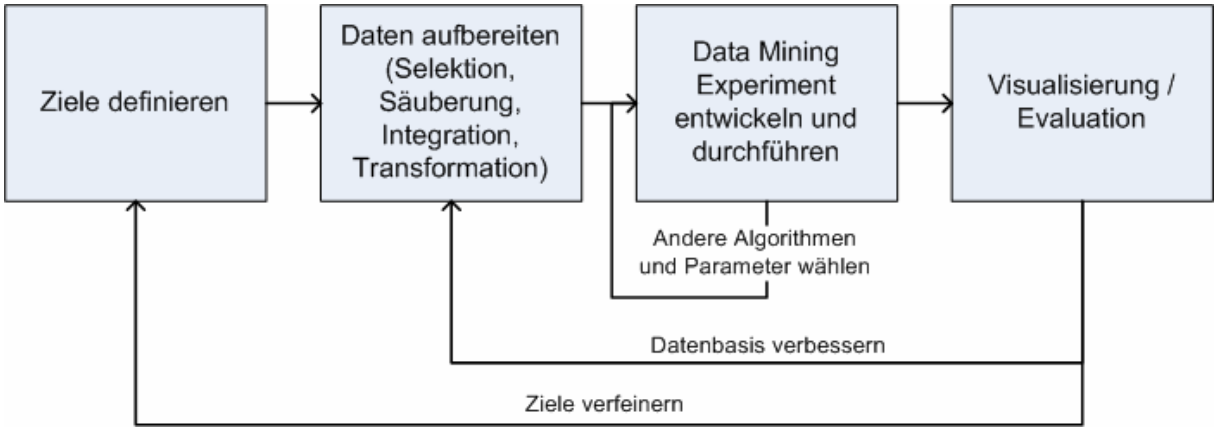


Abbildung 7 – Verwendeter Segmentierungsprozess

Mit Hilfe der Software Weka (siehe Anhang A.1) und RapidMiner (siehe Anhang A.2) lassen sich in kurzer Zeit einige Iterationen dieses Prozesses ausführen. In den ersten Iterationen wird ein exploratives Integrieren und Analysieren der Daten angestrebt, um Erfahrungen über die Datenaufbereitung, Analysemethoden, Segmentierungs- und Klassifikationsmodelle zu sammeln. Somit werden zu Beginn die Ergebnisse und Erkenntnisse einer Iteration fast ausschliesslich für die nächsten Iterationen verwendet; der betriebliche Nutzen steht noch im Hintergrund. Erst durch das Verfeinern der Datenbasis, Analyseverfahren, Modelle und Visualisierung werden die Ergebnisse betrieblich nutzbar. Am Ende soll eine gute Segmentierung der Wertschriftenaufträge mit einem robusten Vorhersagemodell gefunden werden.

Eine Iteration des Segmentierungsprozesses dauert wenige Minuten (andere Parameter für die Segmentierung) bis einige Tage (Ziele anpassen, Datenaufbereitung).

Nachfolgend werden die Bereiche dieses Prozesses fallspezifisch beschrieben.

3.3 Datenaufbereitung

Bevor die Segmentierung mittels Cluster-Algorithmen durchgeführt werden kann, müssen die Rohdaten für das Data Mining aufbereitet werden. Dies erfordert viel Fachwissen und nimmt sehr viel Zeit in Anspruch. [HanKamber2001, S. 7] unterteilt dies in die Säuberung, Integration, Selektion und Transformation der Daten.

Für die Segmentierung habe ich ein Programm in Java (siehe Anhang D) entwickelt, welches Aufträge aus der Auftragsdatenbank liest, weitere Daten integriert, Werte korrigiert und diese Aufträge in eine neue Tabelle speichert. Diese Daten werden dann mittels Datenbankview in einem weiteren Schritt aufbereitet. Obwohl für die Segmentierung nur wenige Attribute erforderlich sind, werden möglichst viele relevante Attribute für die spätere Klassifizierung aufbereitet. Nachfolgend werden die in Java und mit der Datenbankview realisierten Aufbereitungsschritte beschrieben.

3.3.1 Selektion

Für die Segmentierung der Wertschriftenaufträge stehen verschiedene Datenquellen zur Verfügung, welche in einem Mini Data Warehouse zusammengefasst werden (siehe Kapitel 3.4.3). Um bessere Ergebnisse bei der Segmentierung zu erzielen, werden diese Daten folgendermassen reduziert:

- Offene Aufträge enthalten die Ausführungsinformationen noch nicht, weshalb diese ignoriert werden müssen.
- Die Latenz („Latency“) lässt sich nur bei abgeschlossenen Aufträgen vollständig berechnen, da bei offenen Aufträgen die Abschlusszeit noch unbekannt ist.
- Das Berechnen der Latenz telefonischer Aufträge ist wegen inkorrekter Zeitinformation nicht möglich.
- Das Berechnen der Latenz von Aufträgen an der Hongkonger Börse ist sehr komplex, da diese über Mittag (Lokalzeit) geschlossen wird. Aus diesem Grund werden solche Aufträge (weniger als 5% der Aufträge in TOPAZ) nicht selektiert.
- Unvollständige Instanzen (zum Beispiel mit fehlenden Preisinformationen) müssen ignoriert werden.
- Daten vor März 2007 müssen durch die gewonnen Erkenntnisse in Kapitel 4.3.2 ignoriert werden.

3.3.2 Säuberung

Daten, welche inkonsistent sind, müssen in eine konsistente Form gebracht werden. So müssten zum Beispiel 0 Werte des Attributs *effectiveTime* (gültig ab) in „null“ umgewandelt werden, damit diese einheitlich sind und die Resultate nicht verfälscht würden. Da dies nur Aufträge vor Dezember 2006 (welche ignoriert werden) betrifft, ist dies nicht im Ladeprogramm integriert.

Das Attribut *destination* enthält unter anderem die beiden Werte UBSA4LD2 und UBSA4LDN, wobei diese semantisch äquivalent sind. In gewissen Fällen enthält dieses Attribut den Wert „IOM“, welcher durch den Wert des Attributes *execBroker* ersetzt werden muss, damit die Brokerinformation korrekt ist.

3.3.3 Integration

Die Preise verschiedener Wertschriften unterscheiden sich oft um Faktoren. Um aussagekräftige Vergleiche der Ausführungspreise über verschiedene Wertschriften zu erhalten, müssen diese in Relation zum Value Weighted Average Price gesetzt werden.

Aufträge, bei welchen die Auftragsgrösse verglichen mit dem Tagesvolumen einen grossen Teil ausmacht, beeinflussen den Markt stärker als „kleine“ Aufträge. Der theoretische Markteinfluss (siehe [MarketImpactWiki2007]) wird anhand des durchschnittlichen Tagesvolumens und der Auftragsgrösse berechnet, so dass dieses Attribut für das Mining zur Verfügung steht. Der VWAP und das durchschnittlichen Tagesvolumen sind von System Vhayu zu beziehen.

Die Öffnungszeiten der Börsen und Aufträge mit Auftragslimiten beeinflussen die Latenz. Somit müssen Feiertage und Öffnungszeiten integriert werden, um korrekte theoretische Latenzen zu berechnen. Das zusätzliche Integrieren der Kursverläufe für limitierte Aufträge und die Berechnung der effektiven Latenz dieser Aufträge sind nicht Teil dieser Arbeit.

3.3.4 Transformation

Bei allen Segmentierungsalgorithmen muss ein Mass (Distanz oder Ähnlichkeitsmass) zwischen zwei Instanzen oder Attributen berechnet werden. Bei dem Börsenplatz wäre das Berechnen der Distanz wörtlich möglich: die Distanz in Kilometer zwischen der Virt-X (Schweizer Börse) und New York Stock Exchange beträgt ca. 5’000 bis 6’000 km. Eine Generalisierung der Börsenplätze zu Zugangsart (direkter Marktanschluss, elektronisch, telephonisch = manuell) ist jedoch aussagekräftiger als die geographische Distanz.

Die auf die Sekunde genauen Ausführungszeiten sind für die Segmentierung zu präzis. Mittels „Binning“ oder Konzeptbeschreibungen (Öffnungsphase, Mittagszeit, „New York Opening“, etc.) könnten die Zeiten in Bereiche eingeteilt werden, welche weniger Rauschen verursachen respektive aussagekräftiger sind. Versuche haben leider keinen Erfolg ergeben, weshalb darauf verzichtet wird.

Das Attribut *facevalue* (Gegenwert in CHF) hat einen sehr grossen Wertebereich (Rappen bis mehrere Mio. Franken), wobei es sehr viele kleine Aufträge gibt (siehe Histogramm links in Abbildung 8). Es ist offensichtlich, dass ein Clustering, welches auf dem Messen von Distanzen basiert, unnütze Clusters liefern würde. Wird die Skala auf minimal CHF 1’000 und maximal CHF 300’000 limitiert (zweites Historgramm *facevalueLimited*), so werden Ausreisser in einer Gruppe zusammengefasst. Wird eine logarithmische Skala (drittes Histogramm *facevalueLog10*) verwendet, so entsteht eine ausgewogene Verteilung.

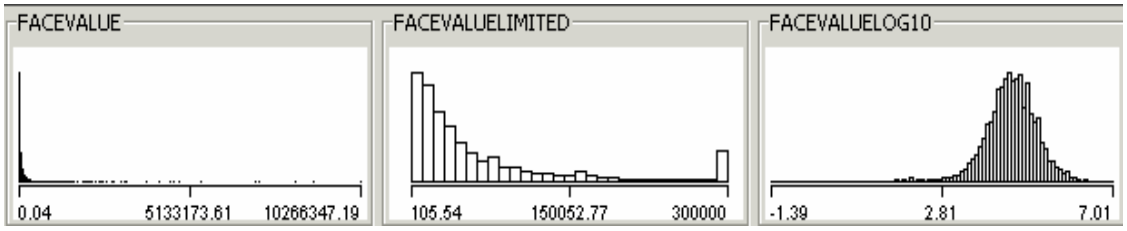


Abbildung 8 – Facevalue-Transformation

In RapidMiner können die Attribute mittels verschiedener Algorithmen (zum Beispiel mit StandardDeviationWeighting) gewichtet und normalisiert werden.

3.4 Mini Data Warehouse

Für das Data Mining-Experiment wird ein Mini Data Warehouse entwickelt, um eine stabile, korrekte Datenbasis zu erhalten. Die Fakten und Domänendaten werden in eine Tabelle (*tbtopAnalyticsOrder*) geladen, um das Mapping und die nachfolgende Datenanalyse möglichst einfach zu halten. Das ETL-Programm (Extract Transform Load) wird in Java geschrieben, da die Datenquellen auf verschiedenen Technologien basieren und dafür geeignete Java-APIs existieren. Dazu wird die Art des Marktanschlusses manuell in eine zusätzliche Tabelle *tbtopAnalyticsMarket* geladen. Die beiden Tabellen werden über eine Datenbankview gejoint und stehen so für das Data Mining zur Verfügung.

3.4.1 Architektur

Die Architektur des Mini Data Warehouse ist in Abbildung 9 ersichtlich. Die Quellsysteme werden mittels eines selbstentwickelten ETL-Programms in eine Zieldatenbank geladen. Dabei werden die verschiedenen Datenquellen direkt integriert und in der Analysetabelle gespeichert.

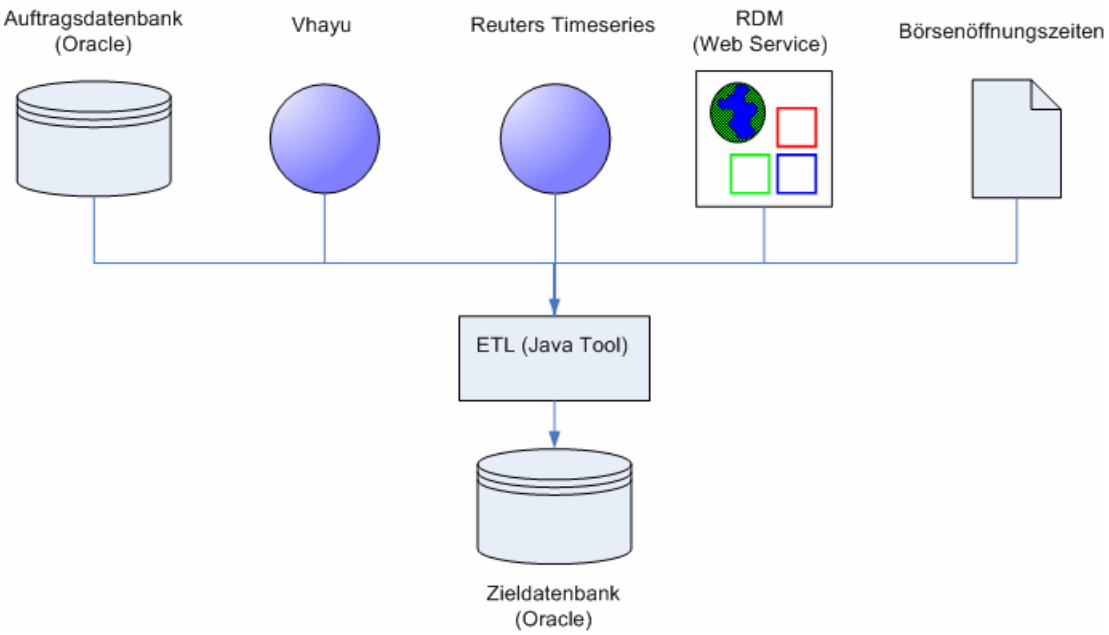


Abbildung 9 – Architektur

3.4.2 Manuell importierte Daten

Für die Art des Marktanschlusses werden die Daten manuell in eine Tabelle geladen, da diese lediglich einmal importiert werden müssen. Dazu werden die Daten der Intranetseite kopiert, mit Regular Expressions in SQL-Befehle transformiert und so in die Datenbank geladen.

3.4.3 Mappingtabelle

Tabelle 1 enthält die Mapping-Definition der wichtigsten Attribute des ETL-Prozesses.

Quellsystem	Quellfeld / Transformation	Zielfeld	Bemerkung
Auftrags-datenbank	orderid	orderid	Auftragsnummer
	createtime - openingtime	afteropening	Der Wert *openingtime* stammt aus der Datei mit den Börsenöffnungszeiten
	sum(shares)	cumulativequantity	Ausgeführte Stückzahl
	currency	currency	Währung
	destination oder execbroker	destination	Broker oder internes Zielsystem (falls *destination*=IOM dann *execbroker*)
	sum(price*shares)	executedfacevalue	Gegenwert (ausgeführt)
	getLatency(…)	latencybroker	Die Methode *getLatency(Startzeit, Endzeit, Börsenplatz)* berücksichtigt Öffnungszeiten und Feiertage
	market	market	Börsenplatz
	side	side	Kauf / Verkauf
	transacttime	transacttime	Erfassungszeit
Vhayu	30dayadv	a30dayadv	Durchschnittliches Tagesvolumen über 30 Tage
	ask - bid	spread	Differenz zwischen Kaufs- und Verkaufskurs
	timereached - starttime	timeforvwap33	Zeit für das dreifache Volumen der Strategie PV33%
	vwap	vwap	VWAP während der Laufzeit des Auftrages
RDM	ric	riccode	Der RIC (Reuters Instrument Code) wird für Vhayu benötigt.
	scaledratechf	fxratechf	Wechselkurs
	ubstitelcode	ubstitelcode	Art des Wertpapiers (Aktie, …)
ETL-Prozess	-	error	Enthält Fehlertexte (zum Beispiel “no Ric found“), die den Fehler beim Laden der Daten beschreiben

Tabelle 1 – ETL-Mappingtabelle

3.4.4 Datenbankview

Mit SQL lassen sich einfach – ohne Programme kompilieren zu müssen – Attribute für die Analyse in Datenbankviews kombinieren und transformieren. Die nachfolgende Tabelle listet die Attribute der View *tbtopAnalyticsView* auf, welche für das Data Mining verwendet wird.

Tabelle	Quellfeld / Transformation	Zielfeld	Bemerkung
Order	log(10, vwap*a30dayadv*fxratechf)	adv30_log	Vereinfachte Umsatzberechnung
	afteropening/3600000	afteropening	In Stunden anstatt Millisekunden
	currency	currency	
	destination	destination	
	least(5, latencybroker / timeforvwap33)	execratio	Max 5 (optimal ist ein Wert von maximal 1/3; der genaue Wert über 5 interessiert nicht)
	log(10, executedfacevalue*fxratechf)	facevaluelog10	Gegenwert des Auftrages in Auftragswährung
	market	market	
	orderid	orderid	
	executedfacevalue/(vwap*a30dayadv)	percentof30daysvol	Anteil des durchschnittlichen Volumens
	least(5, abs((executedfacevalue/ cumulativequantity)/vwap*100-100))	priceimprovement	Max 5 % (für unlimitierte Aufträge ist dieser Wert fair)
	side	side	
	least(100, abs(spread / vwap * 100))	spreadratio	Durchschnittlicher Spread in Prozent der VWAP (maximal 100 % wegen einzelner Ausreisser)
	ubstitelcode	ubstitelcode	
Market	markettype	markettype	

Tabelle 2 – Attributdefinition von tbtopAnalyticsView

3.5 Fazit

Dank der aufbereiteten Daten im Mini Data Warehouse und der Datenbankview stehen nun die Wertschriftenaufträge fürs Data Mining optimal zur Verfügung.

4 Segmentierung

Nachdem die Daten ins Mini Data Warehouse geladen sind, können sie nun mit einem Data Mining-Programm segmentiert werden. Die verwendeten Attribute, Programme, Algorithmen und Parametern werden nachfolgend beschrieben und die erstellten Modelle visuell dargestellt.

Mit dem Weka-Explorer (siehe Anhang A.1) können die Daten aus der Datenbank geladen und mit Hilfe von Histogrammen grob analysiert werden. Bei der Segmentierung müssen alle Attribute ignoriert werden, welche nur für die nachfolgende Klassifizierung benötigt werden. Nachdem der entsprechende Algorithmus gewählt und konfiguriert ist, kann der Data Mining-Prozess gestartet werden. Nachdem die Segmentierung abgeschlossen ist, können die Instanzen 2D-farbig visualisiert werden. Die Cluster-Zuordnung kann nun als ARFF[6]-Datei gespeichert werden und steht somit für die anschliessende Klassifizierung zur Verfügung.

Da RapidMiner (siehe Anhang A.2) nicht über einen Explorermodus verfügt, muss von Beginn an ein Data Mining-Experiment (siehe Kapitel 6) entwickelt werden.

4.1 Wahl der Attribute

Das Ausführungszeit- und Preisverbesserungsratio sind aus meiner Definition von „Best Execution“ offensichtlich die entscheidenden Attribute für die Segmentierung, da diese die Qualität der Ausführung bestimmen. Weitere qualitätsbestimmende Attribute konnten nicht evaluiert werden.

4.2 Algorithmen

Die Clustering-Algorithmen werden in fünf Gruppen unterteilt (siehe [HanKamber2001, S. 346-348]): partionierend, hierarchisch, dichte-, raster- und modellbasiert. Für das Clustering stehen einige Algorithmen in Weka und viele in RapidMiner zur Verfügung. Nachfolgend werden die in dieser Arbeit verwendeten Clustering-Algorithmen beschrieben.

4.2.1 Partionierend

Zu Beginn werden bei SimpleKMeans (siehe [HanKamber2001, S. 349f]) k[7] zufällig gewählte Instanzen als Cluster-Mittelpunkte bestimmt.
Nun werden die Distanzen (zum Beispiel als euklidische Distanz) der Instanzen zu den Cluster-Mittelpunkten gemesssen und jede Instanz wird dem nächsten Cluster-Mittelpunkt zugeordnet. Dann werden die Cluster-Mittelpunkte aus den Instanzen des Clusters neu berechnet. Dieser Prozess wird solange durchgeführt, bis die Abbruchsbedinung (zum Beispiel quadratischer Fehler < x) erfüllt ist.

Beim EM-Algorithmus (Expectation Maximisation, [HanKamber2001, S. 351]) wird für jede Instanz die Zugehörigkeitswahrscheinlichkeit zu den Clustern bestimmt, danach werden die Verteilungsparameter neu geschätzt, sodass die Zugehörigkeitswahrscheinlichkeit für die Instanzen iterativ maximiert wird. Die Anzahl der Cluster wird von EM selbst optimal bestimmt, dafür benötigt er gemäss Tests mit den vorliegenden Beispieldaten ein etwa 100-faches an Zeit, als wenn die Anzahl Cluster a priori gegeben ist.

Tests mit Wertschriftenaufträgen haben gezeigt, dass EM die Daten besser clustert als der einfachere SimpleKMeans, vor allem dann, wenn zwei Gruppen von Instanzen sehr nahe zusammen liegen.

[6] Spezifikation: http://www.cs.waikato.ac.nz/~ml/weka/arff.html
[7] k = Anzahl Cluster, welche a priori bestimmt werden.

4.2.2 Dichtebasiert

DBSCAN (Density-Based Spatial Clustering of Applications with Noise [HanKamber2001, S. 363f]) lässt Cluster wachsen, solange mindestens *min_pts* Instanzen dichtebasiert erreichbar sind. Die restlichen Instanzen werden als Rauschen („Noise“) klassifiziert. Somit entstehen *n* Cluster plus ein Ausreissercluster.

Bei der Density Based Outlier Analysis (siehe [RapidMinerTutorial2007, S. 346-348]) werden Ausreisser anhand einer Verteilfunktion gesucht. Befindet sich eine Instanz nicht nahe genug bei andern Instanzen, wird diese als Ausreisser markiert. Neben der dichtebasierten existiert auch eine distanzbasierte Variante[8]. Dabei wird die Distanz zu den *k* nächsten Instanzen gemessen.

LOFOutlierDetection (siehe [RapidMinerTutorial2007, S. 381-383]) versucht lokale Ausreisser zu finden. Tests haben gezeigt, dass das Verfahren mit meinen Daten nicht anwendbar ist, da nur ein paar wenige (< 0.2 %) Ausreisser gefunden werden.

4.2.3 Hierarchisch

Beim AgglomerativeFlatClustering (siehe [RapidMinerTutorial2007, S. 116]) werden die Cluster Buttom-Up gebildet. Da die Dichte der Instanzen nicht berücksichtigt wird, werden Gruppen von Ausreissern nahe einer grossen Ansammlung von Instanzen nicht als eigener Cluster erkannt (Roter Kreis in Abbildung 10). Die Implementierung in RapidMiner behält einen der beiden Cluster-Namen beim Zusammenfügen zweier Cluster (id 0, id 93, id 34 usw.).

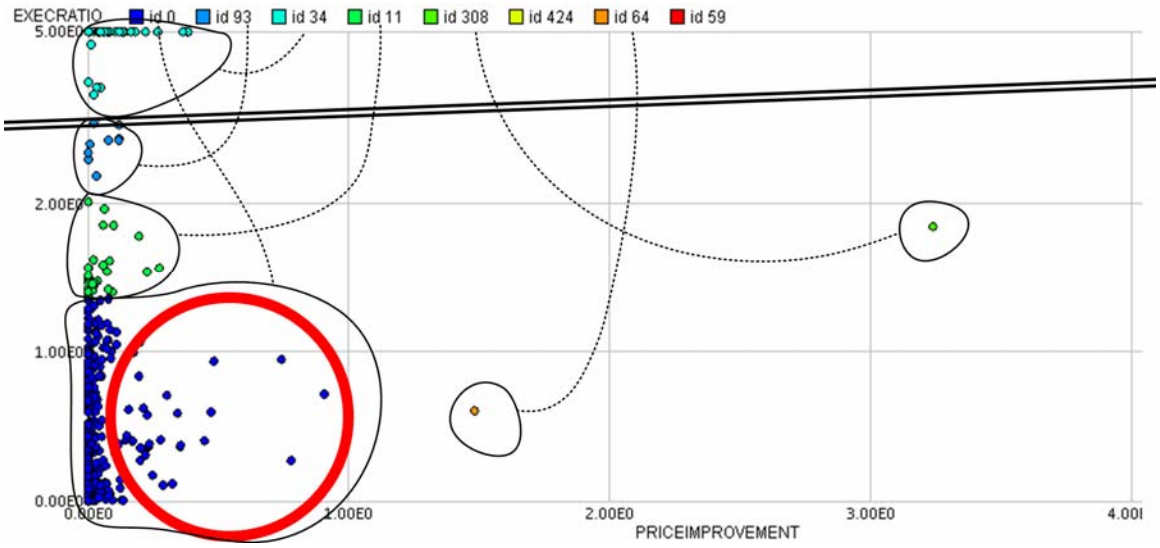


Abbildung 10 – AgglomerativeFlatClustering von Wertschriftenaufträgen

[8] Die distanzbasierte Variante konnte ich in RapidMiner nie testen, da bereits bei 500 Instanzen der Arbeitsspeicherverbrauch über 1 GB liegt und der Algorithmus abstürzt.

4.3 Visualisierung

4.3.1 Visualisierung in 2D

Um die Qualität und Vielfalt der Daten visuell zu prüfen, können zum Beispiel die Instanzen nach zwei Attributen geclustert und mit zwei andern Attributen dargestellt werden. Diese manuelle Mustererkennung erlaubt ein Erkennen von allfälligen Fehlern oder interessanten Fällen in den Basisdaten. Die Abbildung 11 zeigt die Cluster-Zuordnung farblich auf den Achsen Tageszeit (*x*) und Börsenplatz (*y*). Eigentlich liegen die Instanzen pro Börsenplatz auf einer Linie; dank des Jitters kann Rauschen hinzugefügt werden, sodass „versteckte“ Instanzen ersichtlich werden.

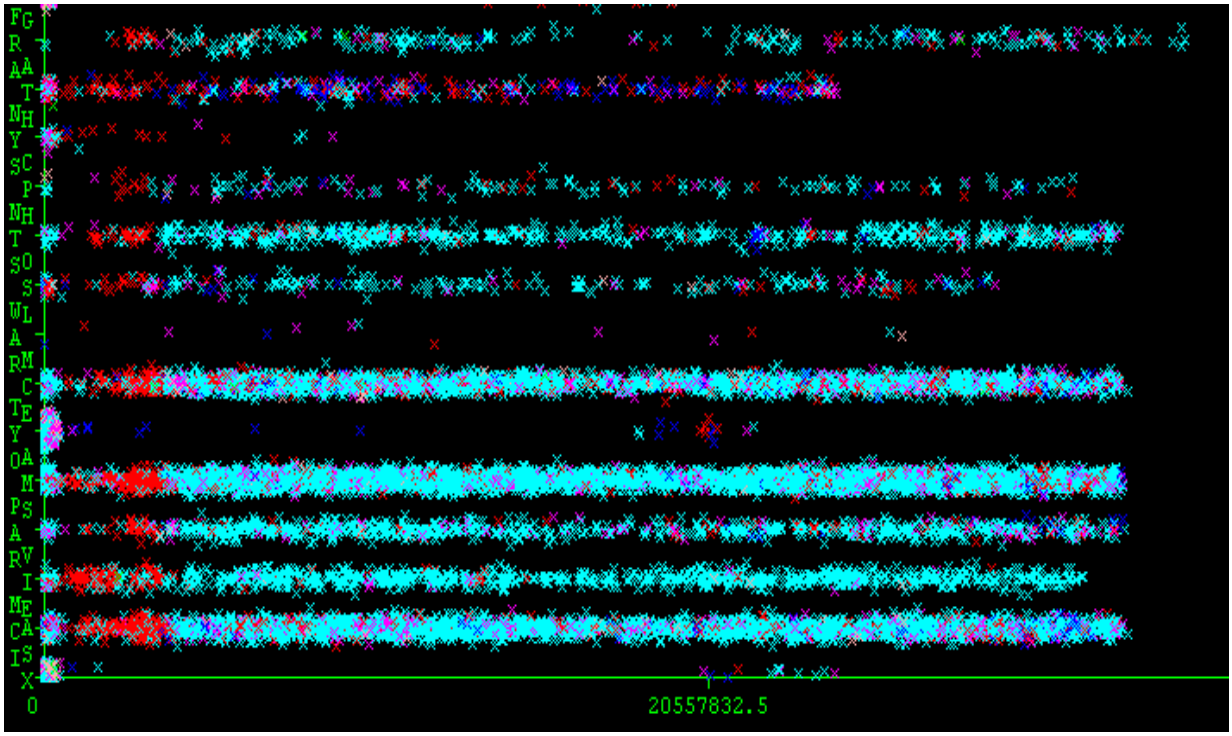


Abbildung 11 – Tageszeit-Börsenplatz-Diagramm

Legende:
x-Achse = „Tageszeit“ (Zeit nach Börsenöffnung)
y-Achse = Börsenplatz
Farbe = Cluster (türkis + violet + rot = 94 %)
Jitter = ca. 1/8

4.3.2 Visualisierung in 3D

Da Weka keine Visualisierung der Daten in 3D anbietet und in RapidMiner die Anzahl dargestellter Punkte limitiert ist, habe ich in kurzer Zeit einen Konverter in Java geschrieben (siehe Anhang D), welcher das Weka-Format (ARFF) in eine 3D-Beschreibungssprache (IV) übersetzt. Dank dieser Visualisierung der Daten nach dem Clustern in 3D konnten weitere Muster erkannt werden, wie zum Beispiel die fehlerhaften Timestamps[9] im Januar und Februar (die ersten fünf Wochen des Jahres 2007, unten links in Abbildung 12). Um das Clustering zu verbessern wird dieser Zeitraum in weiteren Iterationen ignoriert.

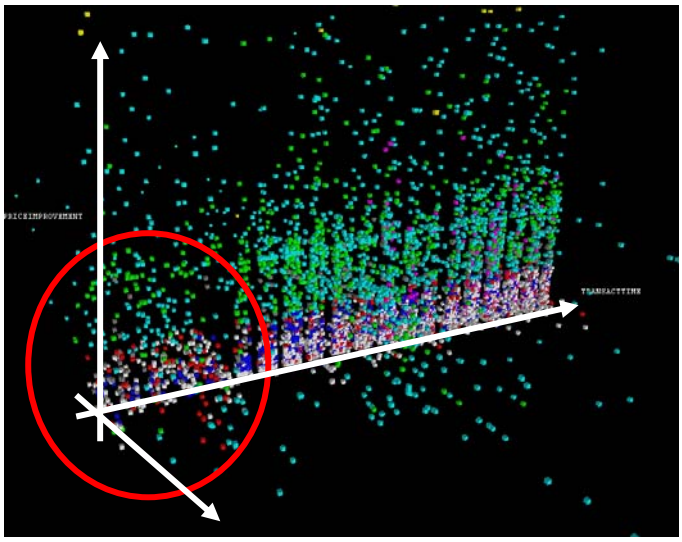


Abbildung 12 – Fehlerhafte Daten Anfang 2007

Ein weiteres interessantes Muster ist auf der z-Achse in Abbildung 13 zu erkennen. Die Berechnung der Preisverbesserung scheint an Freitagen (jeweils letzter Tag vor „Lücke“) nicht immer zu klappen. Eine andere Erklärung wäre die vermutlich grössere Preisabweichung für Aufträge, welche von Freitag bis Montag am Markt platziert sind. Denn die Wahrscheinlichkeit von kursbewegenden Ereignissen ist in dieser Zeit als während der Nacht höher. Nach genauerer Analyse hat sich jedoch herausgestellt, dass diese Preisabweichungen während der Börsenöffnungsphasen entstanden sind und es sich somit um eine eher zufällige Ansammlung von Ausreissern vor Wochenenden handelt. Zudem konnte kein ähnliches Muster in den Folgemonaten erkannt werden.

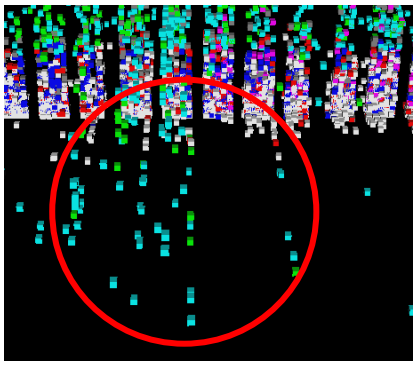


Abbildung 13 – Falsche Berechnung der Preise vor Wochenenden?

[9] Timestamps (Datums- / Zeitfelder) waren zum Beispiel vor dem Softwarerelease Mitte Februar teilweise inkorrekt.

4.4 Parameterwahl

Nachfolgend werden die wichtigsten Parameter der Segmentierungsalgorithmen beschrieben.

4.4.1 EM

Die Anzahl der Cluster *N* kann a priori angeben werden, wodurch die Performance des Algorithmus stark verbessert wird. Um den optimalen Wert für *N* zu bestimmen, kann zum Beispiel nur ein Teil der Daten, ohne *N* a priori zu bestimmen, geclustert werden. Die daraus resultierende Anzahl Cluster kann dann für das Clustern aller Instanzen verwendet werden.

Für die nachfolgenden Experimente wird *N* auf 6 festgelegt. Dies ist ein Kompromiss. Die Anzahl Cluster sollten meiner Meinung nach nicht zu hoch sein, damit die verschiedenen Cluster überhaupt verständlich sein können. Zudem wird aus meiner Erfahrung die Vorhersage für jeden weiteren Cluster schwieriger und ungenauer. *N* darf aber auch nicht zu tief sein, da sonst keine Information mehr in den Clustern steckt.

4.4.2 Density Based Outlier Analysis

Der Parameter *proportion* ist der minimale Anteil aller Instanzen, welche mindestens *distance* weit weg liegen.

Für den Parameter *distance_function* wird der euklidische Abstand in den nachfolgenden Experimenten verwendet.

4.4.3 DBScan

Der Parameter *min_pts* definiert die minimale Anzahl Punkte, welche *max_distance* (Parameter für die maximale Distanz) entfernt liegen, um den Cluster zu erweitern.

Für den Parameter *mesasure* wird der euklidische Abstand in den nachfolgenden Experimenten verwendet.

4.5 Validierung

Die verwendeten Clustering-Modelle werden nachfolgend interpretiert.

4.5.1 EM

Das Interpretieren der einzelnen Cluster ist nicht einfach, da einige sehr nahe bei einander liegen und die Semantik der Cluster nicht offensichtlich ist. Cluster 0 ist der grösste Cluster mit ca. 2/3 aller Aufträge. Im Sinne der Qualität sind Cluster 2 und 3 am interessantesten und entsprechen sogar meinen ungenau definierten Cluster „langsam“ und „schlecht“ in Kapitel 2.2.3. Cluster 1 enthält vermutlich hauptsächlich kleine Aufträge nach der Öffnungsphase für liquide Wertschriften, da die Ausführungszeit sehr kurz ist.

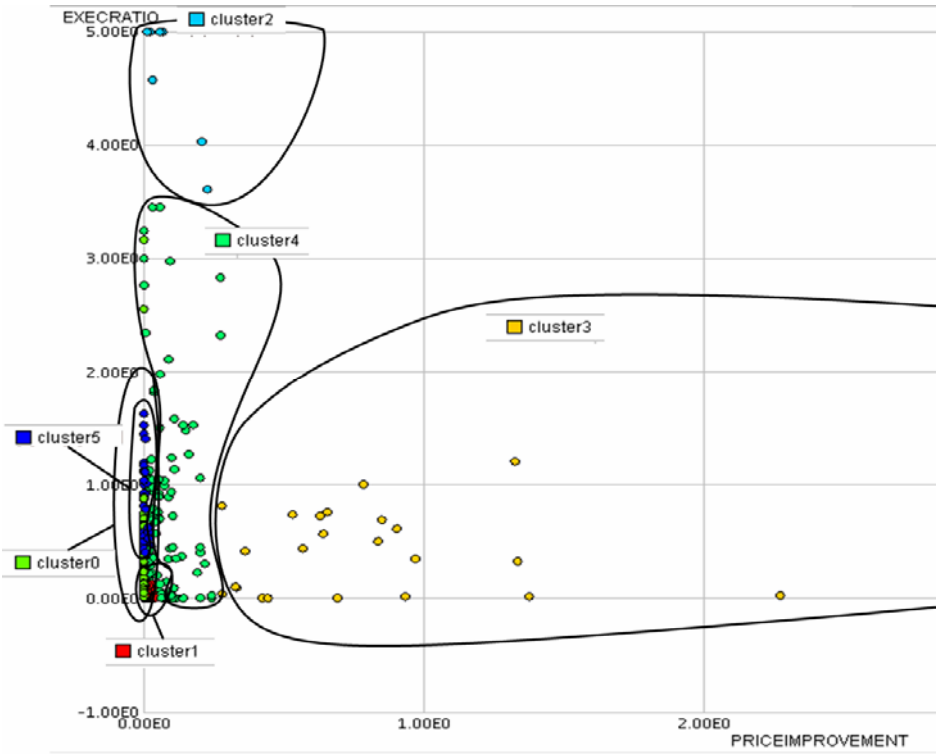


Abbildung 14 – Clustering mit EM von Wertschriftenaufträgen

4.5.2 Density Based Outlier

Die Ausreisseranalyse zeigt mit den Standardparametern (*proportion* = 0.5, *distance* = 0.5) bereits ein sehr einfaches und klares Bild der Instanzen. Die Nicht-Ausreisser bilden den ungenau definierten Cluster „gut“ in Kapitel 2.2.3. Die Instanzen im eingekreisten Bereich wurden nicht als Ausreisser klassifiziert, obwohl sie rein visuell eher Ausreisser sind.

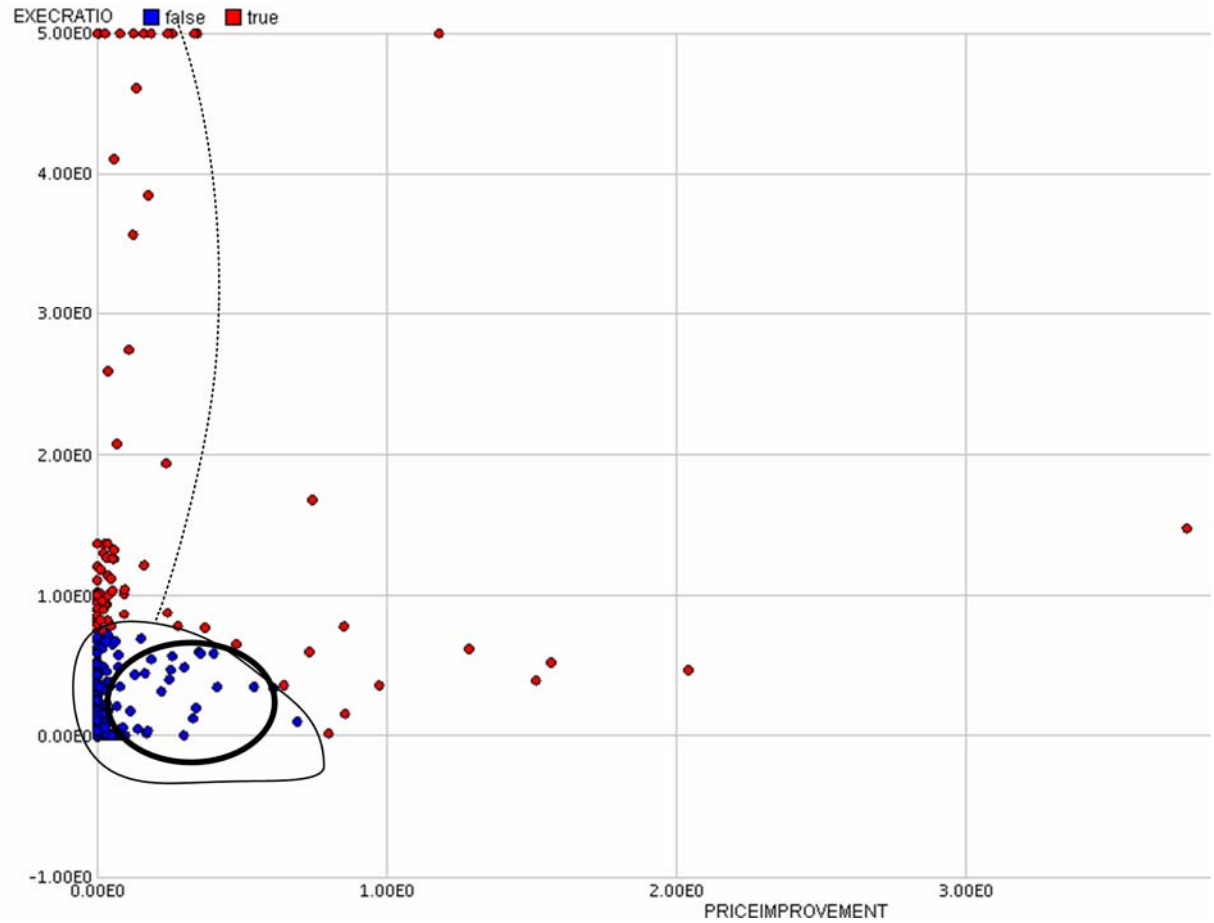


Abbildung 15 – Clustering mit DB Outlier Analyse von Wertschriftenaufträgen

4.5.3 DBScan

Das Clustering mit DBScan ist sehr interessant, da Cluster 0 immer der Cluster mit den Ausreissern ist. Im Gegensatz zu reiner Ausreisseranalyse werden die „Nicht-Ausreisser“ feiner geclustert (siehe eingekreiste Bereiche in Abbildung 16). Dieser Algorithmus liefert je nach Parameterwahl sehr unterschiedliche Resultate. Für das Clustering der Aufträge, welche nicht an UBS London geschickt werden, liefern die Parameter *min_pts* = 30 und *distance* = 0.05 interessante Cluster.

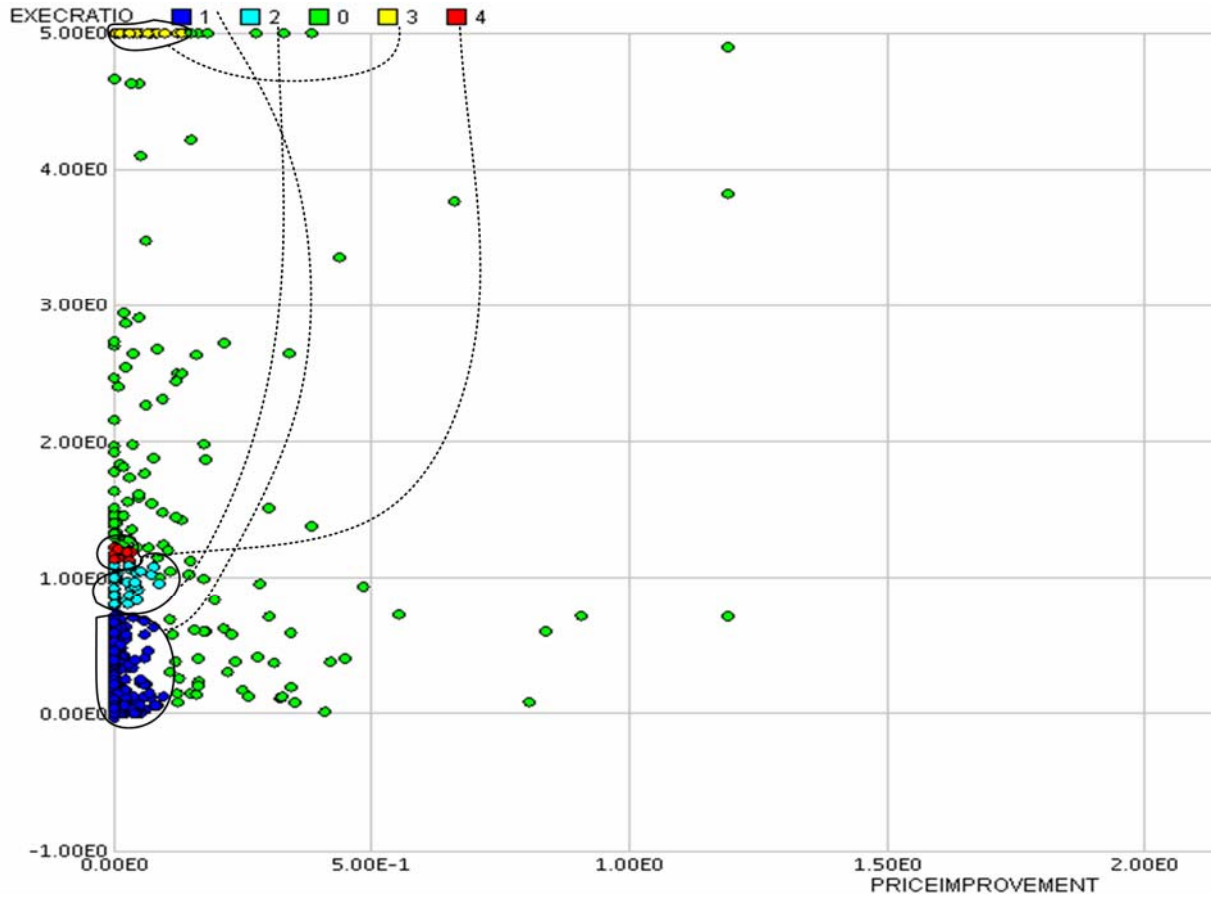


Abbildung 16 – Clustering mit DBScan von Wertschriftenaufträgen

4.6 Fazit

Zahlreiche Tests haben gezeigt, dass sich für die Segmentierung der Abschlussinformationen vor allem die Algorithmen EM, Density Based Outlier Analysis und DBScan eignen. Die Validierung der Datenbasis und der Clustering-Modelle wurde durch mich visuell vorgenommen. Eine objektive Validierung des gesamten Data Mining-Experiments folgt in Kapitel 6.

5 Klassifizierung

Wie für das Clustering stehen für die Klassifizierung zahlreiche Algorithmen zur Verfügung. In diesem Kapitel werden einige Klassifizierungsalgorithmen mit dessen Parametern beschrieben.

Die Klassifizierung erfolgt jeweils auf einem Teilset von Daten und wird anhand der Restmenge validiert, was Kreuzvalidieren (siehe [HanKamber2001, S. 323]) genannt wird. Wenn die Klassen beim Aufteilen mitberücksichtigt werden, spricht man von stratifiziertem Kreuzvalidieren. Das 10-fache stratifizierte Kreuzvalidieren liefert gute Resultate und ist in RapidMiner verfügbar, weshalb dieses Verfahren angewandt wird.

5.1 Algorithmen

Ein vollständiges Vergleichen aller verfügbaren Algorithmen würde den Umfang dieser Arbeit bei Weitem sprengen, weshalb ich mich auf ein paar Algorithmen beschränkt habe. Nachfolgend werden die in der Arbeit verwendeten Algorithmen vorgestellt.

5.1.1 Entscheidungsbäume

5.1.1.1 ID3 und ID3Numerical

Der ID3-Algorithmus (siehe [HanKamber2001, S. 285f]) teilt die Instanzen über das Attribut mit dem höchsten Informationsgewinn in mehrere Teilmengen (eine pro Attributswert) auf. Solange sich verschiedene Klassen in den Teilmengen befinden und noch nicht alle Attribute verwendet wurden, werden diese Mengen wie zuvor als weitere Stufe im Baum aufgeteilt. Der Algorithmus selbst führt kein automatisches Pruning[10] durch. Die Implementation von ID3 in RapidMiner ignoriert numerische Attribute. Der ID3Numerical Algorithmus berücksichtigt alle Attribute und ist deshalb zu bevorzugen.

5.1.1.2 J48 (C4.5)

J48 ist der Name des C4.5 Algorithmus (siehe [HanKamber2001, S. 291f]) in Weka, welcher auch in RapidMiner verfügbar ist. C4.5 ist eine Erweiterung des ID3 Algorithmus, wobei jede Bedingung die Instanzen in jeweils zwei Teilmengen unterteilt. Anhand einer Testmenge werden die Regeln geprüft und falls nötig geprunt und vereinfacht.

5.1.2 Dichtebasiert (Naive Bayes)

Der Naive Bayes-Algorithmus (siehe [HanKamber2001, S. 297-299]) basiert auf dem Bayesschen Theorem. Die Parameter von Wahrscheinlichkeitsverteilungsfunktionen werden geschätzt und optimiert, sodass für jede Instanz die Klassenzugehörigkeitswahrscheinlichkeit bestimmt werden kann. Dabei wird angenommen, dass jedes Attribut nur vom Klassenattribut abhängt.

Für die Berechnung der Zugehörigkeitswahrscheinlichkeit nominaler[11] Attribute wird die Häufigkeit der Vorkommnisse bestimmt. Für numerische Attribute wird die Gausssche Normalverteilungsfunktion verwendet.

[10] Pruning = Beschneiden des (Entscheidungs-) Baums; schlechte Äste abschneiden
[11] Nominale Attribute besitzen keine Rangfolge der Werte (Beispiele Auftragsart, Börsenplatz, Währung).

5.1.3 Regelbasiert

5.1.3.1 Prism

Der Prism-Algorithmus (siehe [RapidMinerTutorial2007, S. 154f]) liefert perfekte Klassifikationsregeln. Er benötigt mehr Zeit als andere Algorithmen. Die Regeln müssen danach geprunt werden oder es wird von Beginn an der erweiterte Ripper-Algorithmus verwendet (siehe nächster Abschnitt).

5.1.3.2 Ripper (Repeated Incremental Pruning to Produce Error Reduction)

Ripper (siehe [RapidMinerTutorial2007, S. 161]) beginnt iterativ Regeln zu erstellen und beschneidet diese gleich. Das Erstellen und Beschneiden geschieht anhand zweier Sets: Trainings- und Validierungsset. Der Prozess wird abgebrochen, sobald die Fehlerrate über 50% steigt. Nur Bedingungen mit dem grössten Informationgewinn werden behalten. Für die nächste Iteration werden die bereits durch die neuen Regeln klassifizierten Instanzen weggelassen.

Dieser Algorithmus ist sehr schnell und liefert oft nur eine Handvoll Regeln, wodurch die Regeln für Menschen (eher) nachvollziehbar werden.

5.2 Parameterwahl

5.2.1 J48

Der Parameter *M*, minimale Anzahl Instanzen in einem Blatt, muss je nach Anzahl zu analysierenden Instanzen variiert werden. Wird ein zu kleiner Wert gewählt, wird der Baum zu genau, zu gross (zu viele Blätter) und die Statistik wird erst noch schlechter beim Kreuzvalidieren. $M = 2 – 5$ ‰ der Instanzen hat sich als brauchbar erwiesen.

5.2.2 NaiveBayes

Dieser Algorithmus benötigt keine vordefinierten Parameter.

5.2.3 Ripper

Der Parameter *grow_part* (Werte 0..1) definiert, welcher Bruchteil an Instanzen für das Trainingsset verwendet wird. Der Rest wird für das Testset verwendet.

5.3 Fazit

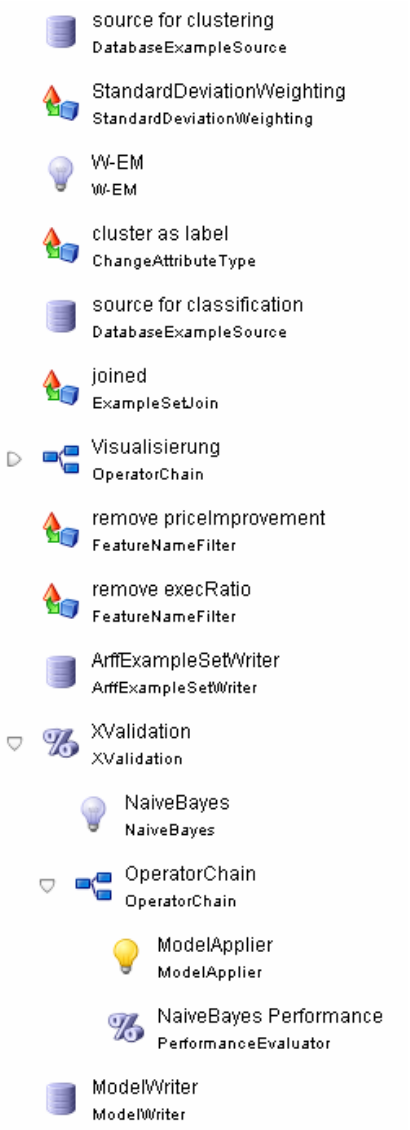
Zahlreiche Tests mit den Wertschriftaufträgen haben gezeigt, dass sich für die Klassifizierung J48, NaiveBayes und Ripper eignen. Diese werden in Kapitel 6 objektiv validiert und miteinander verglichen.

6 Data Mining-Experiment

In diesem Kapitel werden ein Data Mining-Experiment und die Validierung der Vorhersagemodelle beschrieben. Die Definition der Experimente lässt sich speichern und kann so in Zukunft mit neuen Daten wiederholt werden.

Ein Data Mining-Experiment beinhaltet das Lesen der Daten, das Erstellen des Segmentierungs- und Klassifikationsmodells und das Messen der Vorhersagegenauigkeit. Das gelernte Vorhersagemodell kann gespeichert und somit in eigenen Programmen verwendet werden.

Nachfolgend wird ein verwendetes Data Mining-Experiment (Abbildung 17) exemplarisch beschrieben (RapidMinder-Modelle siehe Anhang D).



Einlesen der Instanzen aus der Datenbank

Normalisieren der Attribute

Clustering

Ändern des Typs des Klassenattributs

Einlesen der restlichen Attribute

Speichern der Daten für die Visualisierung

Entfernen der Attribute fürs Clustering

Attributdefinition in ARFF-Datei speichern für die Verwendung im Klassifizierer

10-fache stratifizierte Kreuzvalidierung mit NaiveBayes

Lernen des Klassifikationsmodells

Genauigkeit der Vorhersage berechnen

Schreiben des Vorhersagemodells für die Verwendung im Klassifizierer

Abbildung 17 – Experiment in RapidMiner

6.1 Validierung des Modells

6.1.1 Kriterien

Um die verschiedenen Data Mining-Experimente zu vergleichen sowie das beste Klassifikationsmodell auszuwählen, müssen die Bewertungskriterien klar definiert sein. [HanKamber2001, S. 283] definiert folgende Kriterien für die Validierung des Vorhersagemodells: Genauigkeit der Vorhersage, Robustheit, Geschwindigkeit, Skalierbarkeit, Verständlichkeit (Interpretierbarkeit).

Die Geschwindigkeit für das Erstellen des Modells ist meiner Meinung eher eine Frage der Skalierbarkeit, denn bei wenigen Instanzen (< 100) sind alle Algorithmen sehr schnell. Für mich ist die Reproduzierbarkeit des Modells ein weiterer wichtiger Punkt. Das Modell muss mit wenig Aufwand neu erstellt werden können, um es den technischen sowie betrieblichen Änderungen in TOPAZ anpassen zu können. Mein Ziel ist es, innerhalb von 2 Tagen das Modell neu erstellen zu können. Das Laden der neusten Daten nimmt die meiste Zeit in Anspruch (ca. 1 Auftrag pro Sekunde). Das Analysieren und Produzieren des neuen Modells in RapidMiner ist je nach Algorithmus in weniger Minuten bis Stunden möglich. Das Installieren, Deployen des neuen Modells ist je nach Lösung innert wenigen Minuten, falls nichts programmiert werden muss, möglich. Bei der Entwicklung des Prototyps soll eine Lösung angestrebt werden, welche zukünftig keinen Programmieraufwand benötigt.

Genauigkeit der Vorhersage (Accuracy, Kappa, Precision, Recall)

Die Accuracy (siehe [HanKamber2001, S. 326]) ist die Genauigkeit der Vorhersage (der Anteil der korrekt vorhergesagten Instanzen im Verhältnis zu allen Instanzen).

Die Kappa-Statistik (siehe [KappaWikipedia2007]) gibt die Korrelation der vorhergesagten mit der effektiven Klasse an (Werte über 70 % werden als gut bis ausgezeichnet betrachtet).

Die Precision (siehe [HanKamber2001, S. 325]) ist die Wahrscheinlichkeit, dass die vorhergesagte mit der effektiven Klasse übereinstimmt. Der Recall ist die Wahrscheinlichkeit, dass eine effektive Klasse korrekt vorhergesagt wird. Da diese Werte pro Klasse berechnet werden, müssen sie aggregiert werden, um verschiedene Modelle miteinander vergleichen zu können. Für die Auswertung wird der Durchschnitt der Werte verwendet.

Robustheit (Varianz)

Am besten würde die Robustheit mit Instanzen geprüft werden, welche nicht das Vorhersagemodell bestimmt respektive beeinflusst haben. Je nach Segmentierungsmodell (zum Beispiel bei der Ausreisseranalyse) ist das aber fast unmöglich. Eine einfachere Möglichkeit ist die Analyse der Varianz bei der Kreuzvalidierung. Je kleiner die Varianz von den Vorhersagegenauigkeitswerten ist, desto robuster ist das Modell.

Geschwindigkeit

Die Geschwindigkeit für das Anwenden des Modells an neuen Instanzen ist nicht relevant, da jeweils nicht extrem viele, sondern nur einzelne Instanzen klassifiziert werden müssen. Alle geprüften Modelle sind in der Lage, dies in kurzer Zeit (unter 1 Sekunde) zu tun.

Skalierbarkeit (Zeit)

Bei über 50’000 analysierten Instanzen werden die Laufzeitunterschiede bereits deutlich sichtbar (wenige Minuten sind sehr gut, unter einer Stunde ist akzeptabel).

Reproduzierbarkeit (Zeit)

Dieses Kriterium bezieht sich auf den gesamten Data Mining-Prozess, aber auch auf die Laufzeit der einzelnen Algortihmen und den manuellen Aufwand (zum Beispiel das Interpretieren der Cluster).

6.1.2 Konfusionsmatrix

Die Konfusionsmatrix (siehe [ConfusionMatrixWikipedia2007]) gibt Auskunft, welche Klassen wie vorhergesagt werden. Um verschiedene Vorhersagemodelle und somit verschiedene Konfusionsmatrixen vergleichen zu können, lassen sich diverse Metriken berechnen. Die Spalten der Konfusionsmatrix in RapidMiner enthalten die effektiven Cluster und die Zeilen die vorhergesagten Cluster.

Die beiden Werte Accuracy und Kappa können direkt über die ganze Konfusionsmatrix berechnet werden. Der Recall und die Precision werden pro Klasse berechnet. Um je einen Wert für den Recall und die Precision pro Konfusionsmatrix zu erhalten, kann zum Beispiel jeweils der Durchschnitt (Avg.) berechnet werden. Alternativ kann nur die „Hauptklasse“ betrachtet werden, wobei diese zuerst zu bestimmen ist.

In der Tabelle 3 ist eine Konfusionsmatrix (mit frei erfundenen Werten) ersichtlich, welche die Zusammenhänge der einzelnen Werte veranschaulicht.

		Effektiver Cluster			Precision (Avg. 79 %)
		A	B	C	
Vorher-gesagter Cluster	A	50	3	2	91 %
	B	5	33	1	85 %
	C	0	2	15	88 %
Recall (Avg. 78 %)		91 %	87 %	83 %	

Accuracy = 88 % (= (50+33+15)/111)

Kappa = 81 %

Tabelle 3 – Konfusionsmatrix

6.1.3 Auswertung

Die Experimente wurden mit 56'828 Aufträgen ausgeführt. Das Erstellen der Segmentierungsmodelle dauerte mit EM 5 und mit DBOutlier 20 Minuten.

DBScan skaliert in meinem Fall sehr schlecht. In [HanKamber2001, S. 364] wird eine Laufzeitkomplexität von $O(n^2)$ angegeben. Durch Messungen und ein Kurvenfitting mit Excel konnte eine $n^{2.3}$ Laufzeitfunktion berechnet werden. Nach 17 Stunden habe ich das Experiment mit DBScan, fast 60'000 Instanzen zu klassifizieren, abgebrochen.

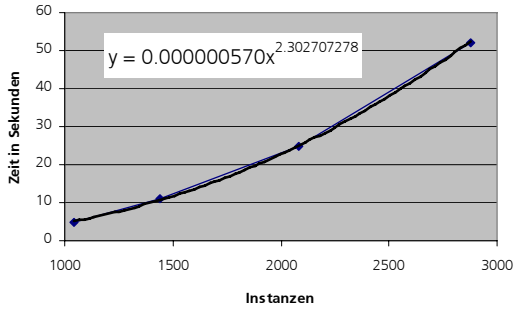


Abbildung 18 – Laufzeit von DBScan

Die Auswertung der Experimente erfolgt gemäss den oben definierten Kriterien. Die Resultate sind in Tabelle 4 ersichtlich.

Die Accuracy ist je Clustering ähnlich. Wenn der grösste Cluster weggelassen wird, so werden die Unterschiede deutlicher. Denn würden bei den Experimenten mit EM einfach alle Aufträge in den grössten der sechs Cluster eingeteilt, so ergibt dies bereits eine Accuracy von 62.85 % und einen durchschnittlichen Recall von 16.67 %, da 36'288 von 56'826 Aufträge sich im grössten Cluster befinden. Beim DBOutlier gibt es 6'722 Ausreisser (11.82 %) und der Rest (88.18 %) bildet den zweiten Cluster. Das Weglassen des grössten Clusters entspricht einer einfachen Kostenmatrix, bei welcher die Kosten für den grössten Cluster 0 und für die restlichen 1 betragen. Ripper liefert zum Beispiel so wesentlich schlechtere Werte als die andern Algorithmen. Die Accuracy bei den Experimenten mit DBScan ist höher, da hier nur zwei Klassen vorhergesagt werden müssen. Ein Vergleich der Werte aller Experimente ist nicht direkt möglich, sondern muss in Relation der Cluster gebracht werden.

Die Kappa-Statistik und der durchschnittliche Recall ergeben ein ähnliches Bild, ohne dass der grosse Cluster weggelassen wird. Der höchste Wert von lediglich 30.64 % zeigt, dass die Korrelation zwischen effektiver und vorgesagter Klasse klein ist. Die Precision ist beim Experiment mit Ripper viel grösser als bei den andern. Dies hat vermutlich damit zu tun, dass Ripper nur wenige genaue Regeln verwendet und so viele Instanzen in den grössten Cluster einteilt, wodurch die Precision der restlichen Cluster gross ist.

Die Varianz von Accuracy ist bei allen Experimenten unter einem Prozent und bei J48 jeweils am tiefsten. Die Varianz von Kappa unterscheidet sich je Experiment fast nicht. Die Modelle scheinen also recht robust zu sein, da genügend Instanzen fürs Erstellen zur Verfügung stehen.

Bei J48 ist zu achten, dass *M* (Anzahl Cluster) der Anzahl Instanzen angepasst wird, so dass der Baum nicht zu viele oder zu wenige Blätter (leaves) enthält. Im Direktvergleich der drei Klassifizierungsalgorithmen ist NaiveBayes extrem schnell und Ripper[12] extrem langsam.

[12] Die Implementation von Ripper in RapidMiner scheint zahlreiche Objekte zu verwenden, welche sehr kurzlebig sind. Dies ist durch eine hohe Garbage Collector Aktivität ersichtlich. Ein Java-optimierter Algorithmus würde vermutlich um ein Vielfaches schneller funktionieren.

Clustering	Klassifizierung	Accuracy [%]	Kappa [%]	Precision [%] **	Recall [%] **	Zeit [h]	Bemerkung
EM (*M*=6)	J48 (*M*=5)	68.83 ± 0.27 21.46*	**30.64** ± 0.7	43.59 48.45*	31.17 18.51*	0:55	**Leaves: 911**
	J48 (*M*=25)	68.59 ± 0.30 20.56*	**29.36** ± 1.0	47.96 43.10*	29.30 16.22*	0:16	Leaves: 237
	J48 (*M*=100)	68.10 ± 0.40 17.80*	26.19 ± 0.7	46.61 41.72*	27.61 7.23*	0:07	Leaves: 117
	Ripper (*grow_part*=0.5)	65.63 ± 0.69 **5.12***	7.87 ± 0.7	**72.36** 74.13*	19.89 3.89*	**9:40**	Regeln: 38
	Ripper (*grow_part*=0.9)	66.24 ± 0.84 **7.23***	11.08 ± 0.9	**61.30** 60.37*	20.95 5.22*	**8:10**	**Regeln: 22**
	NaiveBayes	66.57 ± 0.64 22.98*	**29.35** ± 1.1	35.35 27.67*	32.59 20.87*	0:01	
DBOutlier (*distance*=0.5, *proportion*=0.5)	J48 (*M*=5)	88.51 ± 0.16	14.40 ± 1.7	58.67 ± 4.75	10.24 ± 1.39	0:25	Leaves: 790
	J48 (*M*=20)	88.26 ± 0.07	3.70 ± 1.4	62.14 ± 8.05	2.43 ± 0.94	0:23	**Leaves: 23**
	NaiveBayes	87.04 ± 0.25	19.20 ± 1.9	39.76 ± 2.43	18.56 ± 1.60	0:01	

Interessante Werte sind **fett** markiert.
* der Wert des grössten Clusters wurde ignoriert
** Durchschnittswerte von Precision bzw. Recall
± Varianz

Tabelle 4 – Auswertung verschiedener Experimente

6.1.4 Auswertung (ohne Aufträge „UBS London“)

Da die Vorhersage nicht optimal ist, habe ich versucht, nur eine bestimmte Teilmenge der Instanzen vorherzusagen. Eine interessante Teilmenge sind die 2’739 Aufträge, welche nicht über UBS London geroutet, sondern direkt an einen Broker geschickt worden sind. Die Laufzeit ist nun bei allen Experimenten (inkl. DBScan) unter 2 Minuten.

Mit EM befinden sich 60 %, mit DBOutlier 80.32% und mit DBScan 88.52 % im grössten Cluster. DBScan hat die Instanzen in 4 + 1 (Ausreisser) Cluster eingeteilt.

Die Vorhersage der Cluster gemäss EM ist mit J48 nun deutlich besser als mit NaiveBayes. Der Maximalwert von Kappa ist nun mit 36.04 % höher als die Vorhersage mit allen Aufträgen (30.64 %). Die Varianzen sind auch stark gestiegen, was zwar nicht wünschenswert, aber durch die kleinere Menge an Daten begründbar ist.

Clustering	Klassifizierung	Accuracy [%]	Kappa [%]	Precision [%]	Recall [%]	Bemerkung
EM (*M*=6)	J48 (*M*=5)	68.53 ± 1.64	**36.04** ± 3.3	47.19	38.00	Leaves: 83
	J48 (*M*=25)	67.58 ± 1.64	33.64 **± 4.4**	46.39	37.01	Leaves: 40
	Ripper (*grow_part*=0.5)	65.21 ± 0.80	15.69 ± 2.6	60.33	21.49	Regeln: 3
	NaiveBayes	62.40 ± 1.40	29.37 ± 2.2	33.62	29.41	
DBOutlier (*distance*=0.5, *proportion*=0.5)	J48 (*M*=25)	80.54 ± 1.11	14.80 ± 7.6	52.03 ± 3.54	14.29 ± 7.47	Leaves: 38
	Ripper (*grow_part*=0.5)	**82.48** ± 1.20	21.00 **± 8.6**	79.33 ± 12.83	16.14 ± 6.56	Regeln: 11
	NaiveBayes	79.85 ± 2.31	30.60 ± 5.8	49.37 ± 7.54	37.85 ± 4.18	
DBScan (min_pts=30, max_distance=0.05)	J48 (*M*=20)	**79.77** ± 1.61	32.83 ± 7.6	39.65	28.36	Leaves: 30
	Ripper (*grow_part*=0.5)	78.16 ± 0.91	10.68 ± 7.6	50.11	22.72	Regeln: 4
	NaiveBayes	70.06 ± 9.6	24.44 ± 5.5	33.15	35.59	

Tabelle 5 – Auswertung verschiedener Experimente (ohne Aufträge „UBS London“)

6.2 Fazit

Für die Segmentierung eignet sich EM gut, da die Anzahl Cluster a priori bestimmt werden kann und die Cluster am besten vorhergesagt werden. J48 und NaiveBayes liefern die besten Resultate, wobei sich die Genauigkeit der Vorhersage von NaiveBayes mit einer kleinern Lernmenge verschlechtert. Die mit J48 entwickelten Entscheidungsbäume sind für Menschen einfach zu verstehen, sofern der Baum nicht zu viele Blätter (leaves < 50) enthält. Zudem kann der grösste Cluster aus dem Baum entfernt und mit einem globalen „sonst“ ersetzt werden, um den Entscheidungsbaum zu vereinfachen.

Leider sind die Vorhersagemodelle meiner Meinung nach zu ungenau, um produktiv eingesetzt werden zu können. Der angestrebte Kappa-Wert von 70 % konnte nicht annähernd erreicht werden. Mögliche Verbesserungen der Modelle werden in Kapitel 8.4 beschrieben.

7 Klassifizierer-Prototyp

Für die Vorhersage der Klasse soll nun ein allgemeiner Klassifizierer-Prototyp für den Metrics Service entwickelt werden. Dieser wird dann die Klasse von zukünftigen Wertschriftenaufträgen, also von unbekannten Instanzen, mittels eines Modells vorhersagen. Mit diesem Prototyp soll untersucht werden, ob und wie das RapidMiner-API[13] und die in Zukunft erstellten Modelle im Metrics Service verwendet werden könnten.

7.1 Use Case

Für die Entwicklung dieses Prototyps wird nachfolgend die funktionale Anforderung in Form eines Use Cases beschrieben. Anhand dieses Use Case kann die Software programmiert und getestet werden.

7.1.1 Klassifikation eines Wertschriftenauftrages

	Klassifikation eines Wertschriftenauftrages
Beschreibung	Ein Wertschriftenauftrag wird durch den Klassifizierer anhand eines Klassifikationsmodells klassifiziert.
Vorbedingung	Ein Wertschriftenauftrag wird im System erfasst. Alle für die Klassifikation benötigten Attributswerte sind korrekt erfasst. Ein Klassifikationsmodell ist vorhanden.
Ablauf	Anhand der Attribute des Wertschriftenauftrages sowie des Klassifikationsmodells wird die Klasse bestimmt.
Nachbedingung	Der Wertschriftenauftrag ist klassifiziert
Akteur	Klassifizierer
Ausnahmen	Falls Attributswerte fehlen, wird der Auftrag als „nicht klassifizierbar“ markiert.

Tabelle 6 – Use Case: Klassifikation eines Wertschriftenauftrages

[13] API = Application Programming Interface = Programmierschnittstelle

7.2 Implementierung

In RapidMiner können die Klassifikationsmodelle gespeichert und mittels dokumentierter Java-API in einem selbst entwickelten Java-Programm (siehe Anhang D) geladen werden. Dies hat den Vorteil, dass Änderungen am Modell keine Programmänderung bedeuten.

Für die Vorhersage habe ich eine Klasse entwickelt, welche ein Vorhersagemodell und die Definition der Attribute inklusive Übersetzungstabelle bei der Konstruktion des Objektes einliest. Der in Kapitel 7.1.1 beschriebene Use Case wird in der Methode *predict* realisiert. Diese erwartet die zu vorhersagende Instanz als *java.util.Map* und liefert die vorhergesagte Instanz als *java.lang.String* (siehe Abbildung 19) zurück. Dabei müssen die Namen der Schlüssel in der Map mit den Attributsnamen in der ARFF-Datei übereinstimmen.

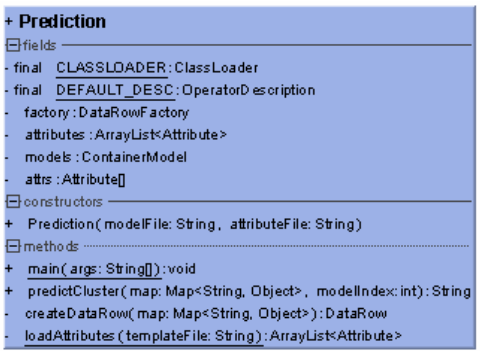


Abbildung 19 – UML-Klassendiagramm

7.3 Integration in TOPAZ

Sofern in Zukunft ein produktiv verwendbares Vorhersagemodell gefunden wird (siehe auch Kapitel 8.4), könnte die Klasse Prediction im Metric Service verwendet werden. Dazu müssten das Modell geladen, die Auftragsdaten in eine *java.util.Map* konvertiert und die Methode *predictCluster* aufgerufen werden.

8 Reflexion

8.1 Businessrelevante Erkenntnisse

Ein sehr überraschendes und interessantes Resultat entsteht durch die Verwendung des Logarithmus beim Auftragsgegenwert. Das Histogramm (siehe Abbildung 8) „facevalueLog10“ zeigt eine fast perfekte Gaussche-Normalverteilung. Der daraus berechnete durchschnittliche Auftragswert entspricht dem einer früheren Analyse der UBS Investment Bank.

Beim Interpretieren des Tageszeit-Marktplatz-Diagramms (siehe Abbildung 20) stelle ich Folgendes fest: Das Muster in Athen (blaues Oval horizontal gestreckt) unterscheidet sich von den Mustern der andern Börsen. Bei der Börsenöffnung aller Börsen ist ein gemeinsames Muster (rotes Oval vertikal gestreckt) erkennbar. Die Problematik der Berechnung der Ausführungszeit ist in Hong Kong wegen der Mittagspause (grünes gestricheltes Oval) sowie der geringen Anzahl von Kundenaufträgen während der Nacht (aus Schweizer Sicht) gut erkennbar.

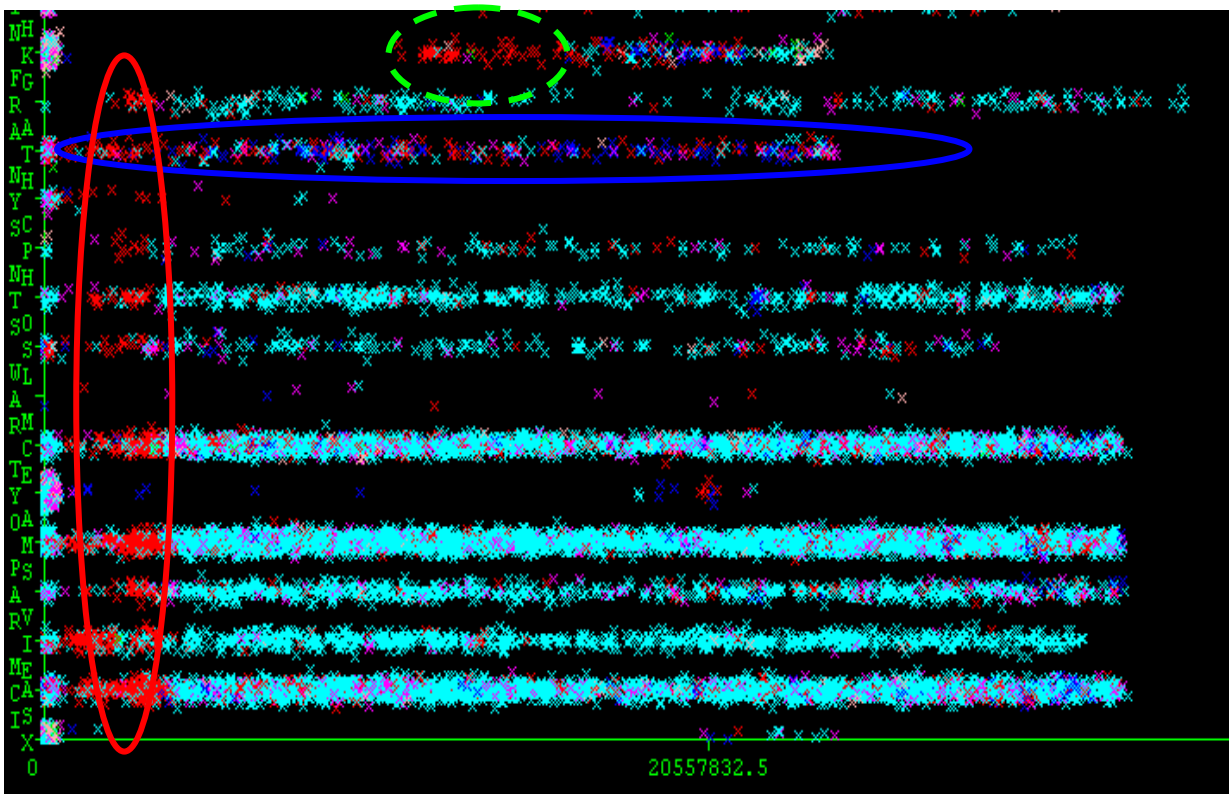


Abbildung 20 – Tageszeit-Marktplatz-Diagramm (Athen, Börsenöffnung)

Das Data Mining-Experiment mit Aufträgen (ohne an UBS London geroutete Aufträge) brachte weitere interessante Erkenntnisse. Die nachfolgenden Erkenntnisse stammen zum Beispiel aus einem Experiment mit EM und J48.

Interpretation der Cluster (siehe Abbildung 21) nach EM:
0 = langsam mit Preisabweichung
1 = sehr langsam mit Preisabweichung
2 = kleine Preisabweichung
3 = o.k. (60 % der Aufträge),
Preisabweichung = 0
4 = grosse Preisabweichung
5 = mittlere Preisabweichung

Das mit J48 erstellte Modell sagt die Cluster folgendermassen vorher:

```
CURRENCY = EUR
|   MARKET = MCI
|   |   AFTEROPENING <= 0.028316
|   |   |   DESTINATION = BROKER X: cluster5
|   MARKET = ISE: cluster1
|   MARKET = ATH
|   |   AFTEROPENING <= 0.08373: cluster5
|   |   AFTEROPENING > 0.08373: cluster1
CURRENCY = USD: cluster2
CURRENCY = SGD: cluster5
CURRENCY = JPY: cluster5
CURRENCY = AUD: cluster2
ELSE cluster3
```

Broker X[14] scheint die Aufträge in MCI beim Opening nicht optimal auszuführen (Cluster 5: Precision = 41 %, Recall = 38 %). An der Börse ISE werden Aufträge generell langsam ausgeführt. Zudem ist in diesem Modell das Problem mit Athen wieder ersichtlich. Solche Informationen können für die UBS sehr wertvoll und Basis für weitere Nachforschungen oder Verarbeitungsregeln sein.

Die Cluster 0 und 4 werden in diesem Modell leider gar nicht vorhergesagt.

[14] Darf wegen Bankgeheimnis nicht genannt werden.

8.2 Erkenntnisse zu Data Mining

Data Mining ist sehr spannend, aber enorm zeitaufwändig. Es gibt einige freiverfügbare Tools und zahlreiche Algorithmen mit diversen Parametern.

Generell muss für die Aufbereitung der Daten sehr viel Zeit eingeplant werden. Je mehr Daten zur Verfügung stehen, desto einfacher gelingt eine stabile, robuste Segmentierung (siehe Varianz in Tabelle 4 und Tabelle 5). Histogramme liefern eine erste Auskunft, welche Attribute sich für die Segmentierung besonders eignen. Die beiden in dieser Arbeit verwendeten Attribute sind nicht optimal, da sie peak-förmige Histogramme besitzen und es deshalb schwieriger wird, gute Cluster zu finden.

Die Validierung der Resultate des Data Mining-Experiments (siehe Kapitel 6.1) lieferte folgende Erkenntnisse: EM clustert die Instanzen sehr schnell, sofern Anzahl Cluster a priori geben sind. Dafür müssen die Cluster manuell interpretiert werden. Die mit Density Based Outlier Analysis geclusterten Instanzen werden ungenau vorhergesagt, dafür sind die beiden Klassen (Ausreisser oder nicht) klar verständlich und benötigen keine Interpretation. Die Varianz ist bei der binären Vorhersage deutlich höher als bei der Vorhersage mehrerer Klassen. Beim DBScan müssen die Parameter durch empirische Tests gewählt werden, damit die beste Kombination gefunden werden kann. Da die effektive Laufzeit $O(n^{2.3})$ beträgt, ist ein Testen nur mit wenigen Instanzen möglich; die Parameter müssen für ein Mehrfaches an Instanzen jedoch angepasst werden.

Das API von RapidMiner lässt sich gut in einer selbst entwickelten Applikation anwenden.

Vorteile:

Neue Ideen für neue Regeln finden: Beispiel (etwas hypothetisch) Aufträge für Athen vor und während Börsenöffnung falls möglich an andere Märkte schicken und erst nach Börsenöffnung nach Athen schicken.

Durch das genaue Analysieren der Daten lassen sich Probleme erkennen (zum Beispiel falsche Timestamps), die bis anhin unentdeckt blieben.

Dank der Vorhersage, wenn diese genauer wäre, könnte ein Agieren und nicht nur ein Reagieren möglich werden. Die für die Vorhersage benötigten Regeln können dank Data Mining gelernt und müssen nicht selber entwickelt werden.

Nachteile:

Das Clustering der Aufträge ist generell fraglich, denn die Cluster ändern sich je nach Algorithmus und Parametern stark. Die Cluster (ab 3 Stück) müssen manuell mit Namen versehen werden, damit das Modell verständlich wird.

Ein weiteres Problem ist die Verfälschung der Instanzen durch vereinfachte Integration. Der in dieser Arbeit verwendete Timestamp für die Ausführungszeit wird vom System TOPAZ vergeben und entspricht nicht dem effektiven Timestamp der Börse (Probleme mit Zeitzonen und fehlender Information sowie beim Opening sind diese je nach Börse so oder so verspätet).

Gewisse Algorithmen (zum Beispiel EM ohne Anzahl Cluster a priori bestimmt, PRISM, RIPPER) benötigen mehrere Minuten oder gar Stunden für das Mining der Daten.

Leider ist die Vorhersage ungenau (ca. 69 % Genauigkeit und Kappa unter 40 %).Eventuell liessen sich die Cluster mit mehr Trainingsinstanzen und weiteren Attributen (Zeitachse) in Zukunft genauer vorhersagen.

8.3 Weitere Erkenntnisse

Bei der Erstellung dieser Diplomarbeit bin ich auf folgende Schwierigkeiten gestossen:

- Die Zieldefinition war zu Beginn unklar. Erst als ich das Ziel der Arbeit exakt definiert hatte, konnte ich effizient daran arbeiten.
- Das Aufbereiten der Daten hat viel mehr Zeit in Anspruch genommen als geplant. Dass ein Data Warehouse mit Metriken (Ausführungslatenz, Preisverbesserungsrate usw.) und andern zusätzlichen Informationen zu den Aufträgen fehlte, bedeutete viel Arbeit. Im Marktdatentestsystem, welches in London betrieben wird, waren nicht immer alle Daten geladen. Dies hat die Analyse erschwert und die Abhängigkeit vom Marktdatenteam in London hat Zeit in Anspruch genommen.
- Das Einteilen der Zeit für eine Diplomarbeit ist schwierig, da ich in anderen Projekten der Bank involviert bin. In dringenden, unvorhersehbaren Situationen muss ich mich diesen zum Teil vollständig widmen. Dank der Erfassung der Arbeitszeit an der Diplomarbeit in Excel hatte ich eine permanente Kontrolle, welche mir für die kurz- und mittelfristige Planung sehr hilfreich war.
- Die stetigen Änderungen der Datenbasis durch das Aufschalten vieler Märkte, verschiedener Broker und die Softwareänderungen in TOPAZ verursachten zusätzlichen Aufwand.
- Leider können nur ca. 25% der Ausgangsdaten analysiert werden (fehlende Preisinformationen, telefonisch gehandelte oder limitierte Aufträge usw. mussten ignoriert werden), dadurch wird die Vorhersage nur für diese eingeschränkte Gruppe gültig. Die restlichen Wertschriftenaufträge können somit nicht vorhergesagt werden.
- Bei der Untersuchung weiterer Attribute stellte ich Folgendes fest, was mir zu Beginn nicht offensichtlich erschien: Die Verwendung eines Attributes bei der Segmentierung und später bei der Klassifizierung beeinflusst die Struktur des Klassifikationsmodells so stark, dass das Modell dann zwar eine „sehr genaue“ Vorhersage liefert, diese jedoch nutzlos ist, da sie (fast) ausschliesslich bekannte Information vorhersagt. Wird zum Beispiel das durchschnittliche logarithmische Volumen in Franken in die Segmentierung miteinbezogen, so resultieren „Durchschnittsvolumencluster“ anstelle von Qualitätsclustern. Da sich nur drei Attribute eignen, konnten die Auswirkungen auf die Robustheit der Cluster nicht abschliessend geprüft werden.
- Da das in den ersten Monaten verwendete Data Mining-Tool (Weka) zum Beispiel nicht über eine grosse Anzahl Clustering-Algorithmen verfügt, habe ich zu lange mit zu wenigen Algorithmen gearbeitet. Der Einsatz von RapidMiner brachte eine wesentliche Verbesserung, wobei das Testen und Vergleichen der verschiedenen Algorithmen zeitaufwändig war.
- RapidMiner stellt nur maximal 1’000 Instanzen in einem Diagramm dar. Das Umwandeln der Daten von ARFF in IV mit dem selbstentwickelten Java-Programm ist zwar etwas aufwändiger und unflexibler, dafür lassen sich dort 50’000 oder mehr Instanzen darstellen.
- Die API-Dokumentation sowie das Handbuch waren für die Entwicklung der eigenen Applikation nicht ausreichend. Im Handbuch ([RapidMinerTutorial2007, S.487]) gibt es nur wenige Beispiele, wie das RapidMiner API verwendet werden soll. Somit musste ich einiges selber herausfinden, zum Beispiel wie die Klassen zusammen

anzuwenden sind oder wie die Übersetzungstabellen der Attribute geladen werden können.
In RapidMiner werden generell alphanumerische Attributswerte je Attribut mit einer Übersetzungstabelle in numerische Werte umgewandelt. Jede Instanz wird zum Beispiel in einem double-Array gespeichert. Das gespeicherte Klassifikationsmodell enthält nur die numerischen Werte. Damit das Modell später im eigenen Programm verwendet werden kann, müssen die Übersetzungstabellen zusätzlich gespeichert werden. Deshalb muss das für das Aufstellen des Modells verwendete ExampleSet in eine ARFF-Datei geschrieben werden. Dieses Format wird eigentlich in Weka verwendet und die Instanzen werden in einer einfach strukturierten Textdatei geschrieben. Die in dieser Datei enthaltenen Instanzen, welche nach der @data Zeile folgen, sind für den Klassifizierer irrelevant und können aus der Datei in einem Texteditor gelöscht werden. Im Klassifizierer-Prototyp wird lediglich die Definition der Attribute mit den Übersetzungstabellen benötigt.
Da aber der Quelltext von RapidMiner frei verfügbar ist, ist ein Debuggen der eigenen Applikation und des RapidMiner APIs möglich, um solche Probleme lösen zu können.

- Und natürlich gab es auch noch kleinere Missgeschicke: So habe ich 8 Wochen vor der Abgabe der Arbeit versehentlich die aufbereiteten Daten in der Datenbank gelöscht, wodurch ich fast 2 Tage für das Wiederherstellen der Daten verloren habe.

8.4 Fazit

Meiner Meinung nach macht es Sinn, weitere Data Mining-Projekte im Bereich der Auftragsverarbeitung zu betreiben, da daraus wesentlich neue Erkenntnisse gewonnen werden können. So könnten zum Beispiel dank weiterer Integration von Daten und Kostenmodellen vollständige Analysen von Erfassung des Auftrages bis zur Abwicklung erstellt werden, wodurch die mutmasslichen Kosten pro mögliche Verarbeitungsart oder -pfad vorhergesagt werden könnten. Es ist sehr schwierig, eine faire Kostenersparnis zu berechnen, aber vielleicht genügen die folgenden Überlegungen:

- Wird nur ein Millionenauftrag nicht gut verarbeitet, entstehen unter Umständen grosse Kundenforderungen.
- Jeder unnötig manuell verarbeitete Auftrag verursacht Aufwand und somit Kosten. Durch den Einsatz solcher Modelle könnte dieser Aufwand höchstwahrscheinlich reduziert werden.

Die Vorhersage der Ausführungsqualität von Wertschriftenaufträgen gemäss den erstellten Modellen ist meiner Meinung nach zu ungenau, um produktiv eingesetzt werden zu können. Das Überprüfen der Zieldefinition von Best Execution wäre mit einem genaueren, verbesserten Modell möglich. Um das Modell zu verbessern, schlage ich folgende Änderungen und Erweiterungen, welche vor allem aus Zeitgründen in dieser Arbeit nicht berücksichtigt worden sind, vor:

- Cluster einmalig manuell definieren, um die Einteilung der Qualität selber bestimmen zu können. Somit entfällt das Interpretieren eines erneuten Clusterings und damit manueller Aufwand. Es ist nicht klar, ob die Vorhersage mit einer „zufälligen“ Segmentierung schlechter oder genauer wird.
- Aufträge aus Kundensicht und nicht nur aus Marktsicht analysieren (Auftragssplitting berücksichtigen). Einsatz einer In-/Out-of-limit Minding Komponente in TOPAZ, um auch limitierte Aufträge analysieren zu können.
- Es sollten genauere Timestamps für Ausführungszeit der Wertschriftenaufträge verwendet werden, was vermutlich sehr aufwändig ist.
- Um die Differenzen verschiedener Aufträge besser vergleichen zu können, könnte diese als „Anzahl Preisschritte“ im Verhältnis zum durchschnittlichen Spread (in Preisschritten) berechnet werden.
- Erfassungszeitpunkt beim Segmentierungsmodell mitberücksichtigen, um den Wandel der Systeme einzubeziehen. Tests mit *transacttime* als drittes Attribut beim Clustering lieferten 5% bessere Ergebnisse für Kappa (37.5%) bei der Vorhersage.

Anhang A Tools

A.1 Weka

Homepage: http://www.cs.waikato.ac.nz/~ml/weka/

Version: 3.4.10

A.1.1 Zusammenfassung

Weka ist eine Sammlung von Algorithmen für Data Mining-Tasks. Diese können direkt im GUI ausgeführt oder über ein Java-API aufgerufen werden. Das Tool steht kostenlos unter der GNU General Public Licencse (GPL) zur Verfügung.

Für die Segmentierung hat sich der Weka-Explorer in den ersten Iterationen des Segmentierungsprozesses sehr bewährt. Leider verfügt Weka nicht über Outlier Clustering Algorithmen.

A.1.2 Datenbankabfragen

Damit die Datenbankverbindung mit Oracle hergestellt werden kann, muss *ojdbc14.jar* ins Installationsverzeichnis von Weka kopiert und in der Datei RunWeka.bat dem Classpath hinzugefügt werden:

```
%_java% -classpath . RunWeka -i .\RunWeka.ini -w .\weka.jar;ojdbc14.jar […]
```

Nachdem der Weka-Explorer gestartet ist, lässt sich die Datenbankabfrage über „Open DB…“ einrichten. Die Einstellungen können abgespeichert werden, was sich als sehr vorteilhaft erweist, wenn mit verschiedenen Quellen und Tabellen gearbeitet wird.

A.1.3 Memory Settings

Der Java Heap Space kann in der Datei *RunWeka.ini* folgendermassen spezifiziert werden:

```
maxheap=512m
```

Somit lassen sich grössere Datenbankabfragen ausführen, ohne dass Weka wegen OutOfMemoryExceptions abstürzt. Sofern auf dem verwendeten Computer genügend Arbeitsspeicher vorhanden ist, sollte diese Einstellung gleich nach der Installation von Weka vorgenommen werden.

A.1.4 Filter

Wenn beim Klassifizieren Attribute aus der Datenquelle (ARFF-Datei) weggelassen werden sollen, können diese entweder einzeln im Tab „Preprocess“ manuell oder mittels eines Filters „*weka.filters.unsupervised.attribute.Remove*“ automatisch entfernt werden. Bei diesem Filter müssen die zu entfernenden Attribute als Nummer angeben werden (Bespiel: 1-5, 7, 10). Der Filter kann gespeichert und beim erneuten Laden der ARFF-Datei wieder angewendet werden.

A.1.5 Histogramme

Über „Visualize All…“ werden Histogramme aller Attribute angezeigt, womit die Verteilung der Werte sehr schnell geprüft werden kann. Abbildung 22 zeigt eine solche Übersicht eines ganz kleinen Testsets. Bei dem Attribut Shares zeigt sich zum Beispiel, dass die Testdaten sehr schlecht verteilt sind, da sich alle Instanzen ausser eine im ersten Bin befinden. Da die Histogramme der Attribute *Price* und *Currency* fast deckungsgleich sind, müssen die Testdaten genauer betrachtet werden. Es zeigte sich, dass keine direkte Korrelation besteht, aber bei dem Attribut *Price* einige Instanzen den Werte 0 enthalten. Bei *Vwap* fehlen die Daten komplett, was im Histogramm sofort ersichtlich ist.

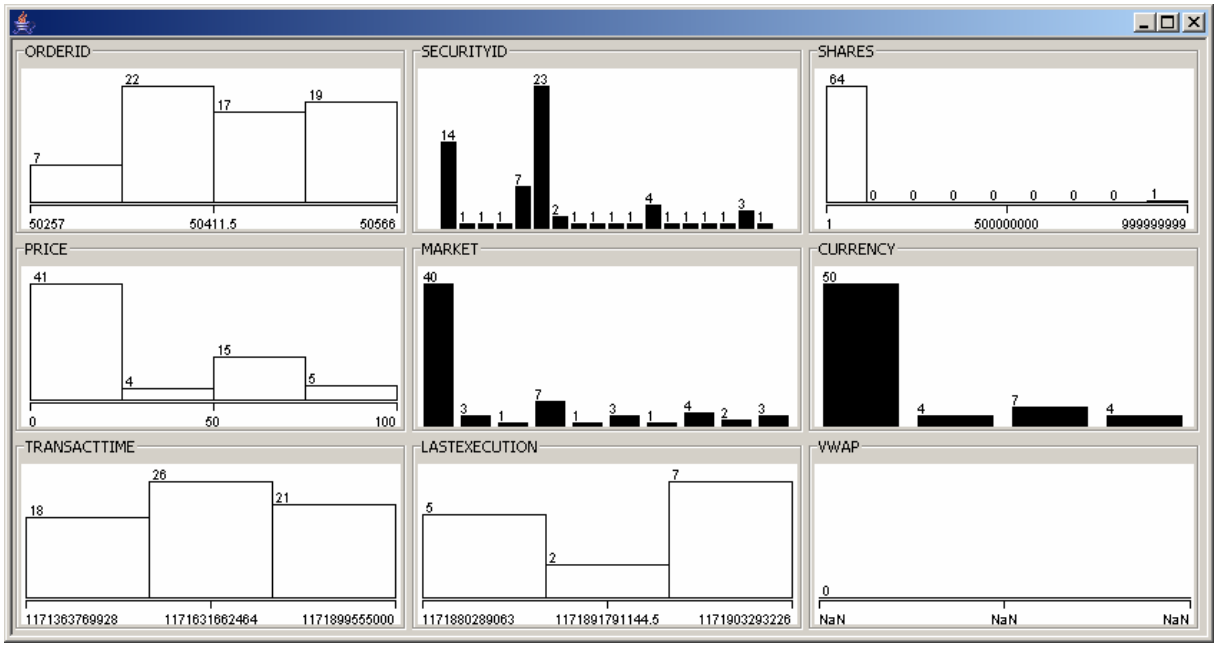


Abbildung 22 – Histogramme in Weka

A.2 RapidMiner

Homepage: http://rapid-i.com/

Version: 4.0beta

A.2.1 Zusammenfassung

RapidMiner ist ein sehr modernes Data Mining-Tool mit vielen Features und Algorithmen. RapidMiner wird vom Unternehmen Rapid-I kostenlos unter der GNU General Public Licencse (GPL) zur Verfügung gestellt oder kann kostenpflichtig lizenziert werden. Gemäss einer Umfrage auf KDnuggets.com ist RapidMiner (ehemalig YALE) im Jahr 2007 das meistgenutzte freiverfügbare Data Mining-Tool.

In RapidMiner können ganze Data Mining-Prozesse (Experimente) graphisch modelliert werden. Das Laden, Clustern, Manipulieren und Segmentieren der Daten ist baumartig strukturiert. Via Start-Button wird das gesamte Experiment gestartet. RapidMiner enthält eine grosse Anzahl Algorithmen (Pre- & Postprocessing, un- & supervised Learner usw.) und hat zudem die Algorithmen von Weka integriert.

Das Erstellen eines Ablaufs ist zu Beginn etwas umständlich, da die einzelnen Anweisungen implizite In- und Output-Objekte (*Example-Set*, *Model* etc.) liefern und benötigen. Via F1 können pro Anweisung diese Objekt-Klassen angezeigt werden. Explizite Variabeln (*a* = …) gibt es nicht, dafür könnten die Resultate in Dateien (mit Namen) gespeichert und später wieder geladen werden.

Um die Daten manuell zu prüfen und graphisch zu analysieren, können Breakpoints gesetzt werden. Alternativ lassen sich die Daten mittels *IOMultiplier* und *IOSelector* zwischenspeichern. Die Daten werden am Ende des Prozesses in Registerkarten angezeigt.

Immer wiederkehrende Anweisungen können als „*Building Blocks*“ zusammengefasst und an anderen Stellen wieder verwendet werden. Damit lässt sich der Data Mining-Prozess gut strukturieren und mittels Name des „*Building Blocks*“ zudem dokumentieren.

Die Clustering- und Segmentierungsalgorithmen können in einer *XValidation* eingefügt werden. Dabei ist zu beachten, dass im *PerformanceEvaluator* (in *XValidation*, *ModelApplier* enthalten) das gewünschte Hauptkriterium (*main_criteria*) zum Beispiel accuracy ausgewählt ist.

Das Java-API kann in eigenen Java-Applikationen verwendet werden. Zum Beispiel können Experimente geladen und mit neuen Instanzen ausgeführt werden.

A.2.2 Tipps und Tricks

Die einzelnen Schritte können durch den Benutzer beschriftet werden. Dabei dürfen keine Sonderzeichen verwendet werden, da Dateien mit ä, ö, ü nicht mehr geladen werden können.

Damit die Datenbankverbindung mit Oracle hergestellt werden kann, muss lediglich *ojdbc14.jar* ins *jdbc* Verzeichnis von RapidMiner kopiert werden.

Bevor die geclusterten Instanzen in eine Datenbank geschrieben werden können, muss das Attribut *cluster* umbenannt werden, da dies ein SQL-Schlüsselwort ist.

A.3 Open Inventor Tools – IvView

SGI Inventor: http://oss.sgi.com/projects/inventor/

Homepage: http://merlin.fit.vutbr.cz/wiki/index.php?title=Open_Inventor_Tools

Das Tool IvView eignet sich sehr gut, um 3D-Szenen zu rendern und mittels Maussteuerung aus verschiedenen Perspektiven zu betrachten.

Einige in dieser Arbeit verwendeten 3D-Darstellungen wurden mittels IvView erstellt. Die Daten im Weka-Format (ARFF) können in das Format IV (SGI Inventor Format) durch ein selbst entwickeltes Java-Programm umgewandelt (ARFF2IV) werden.

Die Dateien werden in ASCII codiert. Die Szene wird in Gruppen aufgeteilt (Seperator), in welchen vordefinierte Objekte (*Swtich, DEF, Cube*) verwendet und mittels Transformationen (Transform) positioniert werden können. Folgender Abschnitt zeigt einen Ausschnitt aus einer generierten IV-Datei. Das Resultat ist in Abbildung 23 ersichtlich.

```
#Inventor V2.0 ascii
Separator {
   LightModel { model PHONG }

   # Definition einer Instanz
   Switch {
      DEF punkt Cube { width 0.3 height 0.3 depth 0.3 }
   }

   # Zuordnung der Farben der Instanzen
   DEF cluster0 Material { diffuseColor 1.0 0.0 0.0 }
   DEF cluster1 Material { diffuseColor 0.0 1.0 0.0 }
   DEF cluster2 Material { diffuseColor 0.0 0.0 1.0 }
   # ...

   # nun folgen die einzelnen Instanzen
   Separator {
      Transform { translation  3.23 1.84 4.37 }
      USE cluster1 USE punkt
   }
   Separator {
      Transform { translation  2.05 0.75 4.55 }
      USE cluster0 USE punkt
   }
   Separator {
      Transform { translation  4.86 2.56 1.21 }
      USE cluster2 USE punkt
   }
   # ...
}
```

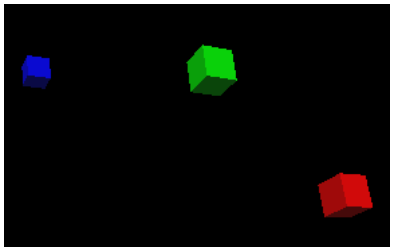
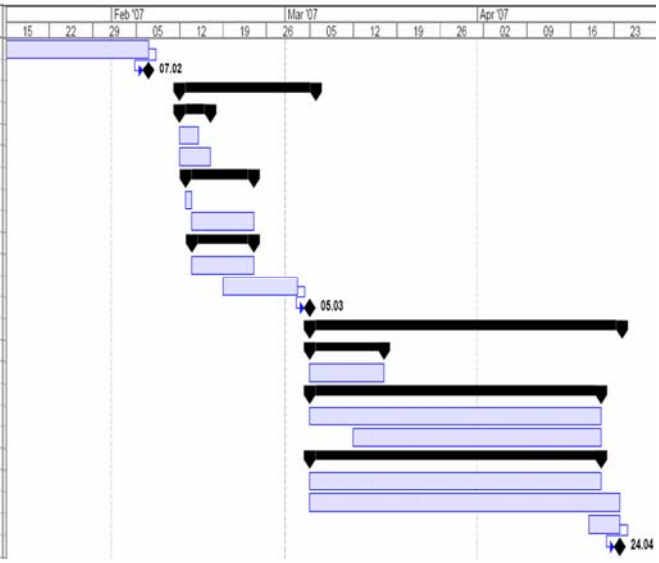


Abbildung 23 – IV-Beispieldatei

Anhang B Projektplan

ID	Task Name	Start	Finish
1	Definition der Aufgabenstellung	15.01.07	06.02.07
2	Kick-off	07.02.07	07.02.07
3	**1. Iteration**	**12.02.07**	**05.03.07**
4	**Analyse**	**12.02.07**	**16.02.07**
5	Zieldefinition der Arbeit	12.02.07	14.02.07
6	Datenquellen (Auftragsdaten, Vahvu, RI	12.02.07	16.02.07
7	**Design**	**13.02.07**	**23.02.07**
8	ETL Prozess	13.02.07	13.02.07
9	Segmentierung	14.02.07	23.02.07
10	**Realisierung**	**14.02.07**	**23.02.07**
11	ETL Prototyp	14.02.07	23.02.07
12	Dokumentation	19.02.07	02.03.07
13	Meeting mit Herrn Wolfertz	05.03.07	05.03.07
14	**2. Iteration**	**05.03.07**	**24.04.07**
15	**Analyse**	**05.03.07**	**16.03.07**
16	Datenquellen (Holiday, Markets)	05.03.07	16.03.07
17	**Design**	**05.03.07**	**20.04.07**
18	Segmentierung	05.03.07	20.04.07
19	Klassifizierung	12.03.07	20.04.07
20	**Realisierung**	**05.03.07**	**20.04.07**
21	ETL Erweiterung	05.03.07	20.04.07
22	Dokumentation	05.03.07	23.04.07
23	Präsentationsvorbereitung	19.04.07	23.04.07
24	Design-Review	24.04.07	24.04.07



ID	Task Name	Start	Finish
25	**3. Iteration**	**14.05.07**	**18.06.07**
26	**Analyse**	**14.05.07**	**30.05.07**
27	ETL (Review)	14.05.07	16.05.07
28	Open Inventor	28.05.07	30.05.07
29	**Design**	**14.05.07**	**15.06.07**
30	Segmentierung	14.05.07	23.05.07
31	Klassifizierung	04.06.07	15.06.07
32	**Realisierung**	**14.05.07**	**31.05.07**
33	ETL Optimierung	14.05.07	16.05.07
34	3D Visualisierung	30.05.07	31.05.07
35	Dokumentation	14.05.07	15.06.07
36	Meeting mit Herrn Wolfertz	18.06.07	18.06.07
37	**4. Iteration**	**18.06.07**	**08.08.07**
38	**Design**	**18.06.07**	**06.07.07**
39	Segmentierung- und Klassifizierungsmo	18.06.07	29.06.07
40	Klassifizierer Prototyp	02.07.07	06.07.07
41	**Realisierung**	**02.07.07**	**13.07.07**
42	Klassifizierer Prototyp	02.07.07	13.07.07
43	Dokumentation	18.06.07	31.07.07
44	Abgabe der Arbeit	08.08.07	08.08.07
45	**Abschluss**	**30.07.07**	**22.08.07**
46	Poster für Diplomausstellung	02.08.07	08.08.07
47	Präsentationsvorbereitung	30.07.07	20.08.07
48	Schlusspräsentation	22.08.07	22.08.07

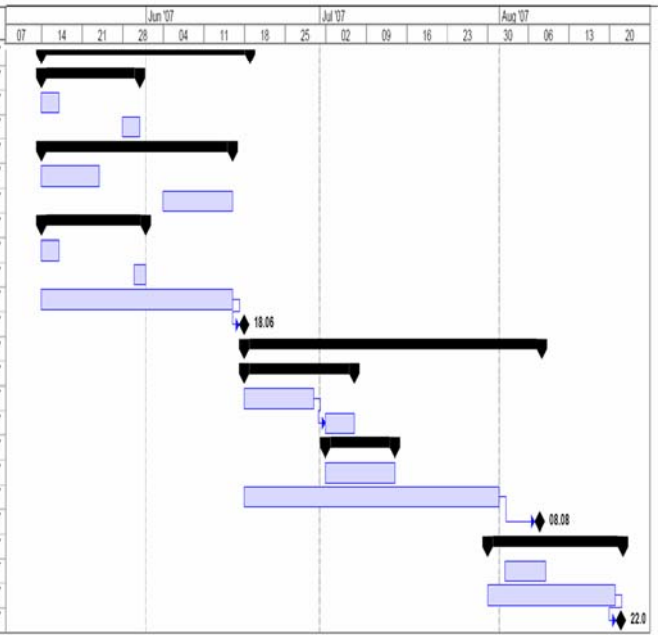


Abbildung 24 – Projektplan

Anhang C Marktphasen

C.1 Virt-X

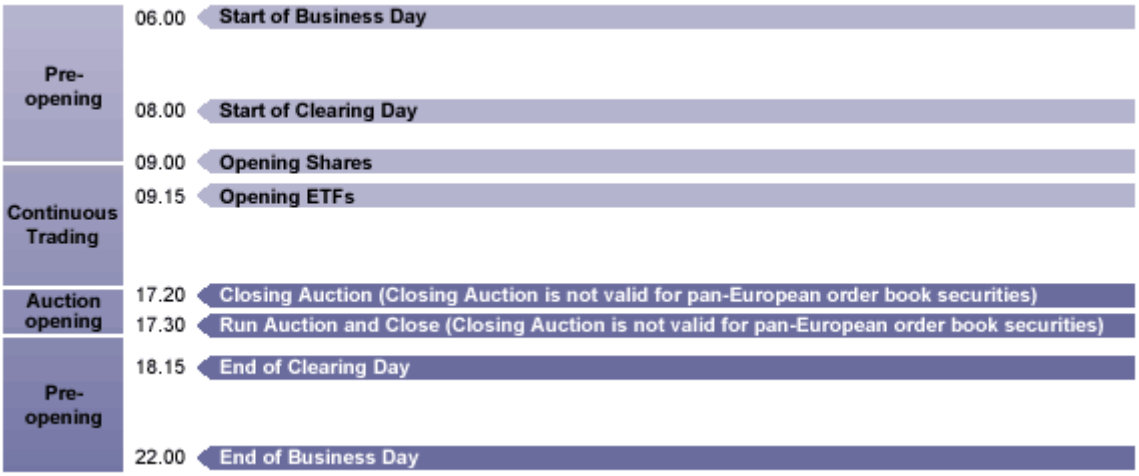


Abbildung 25 – Marktphase Virt-X, Quelle [VTX2007]

C.2 Athen

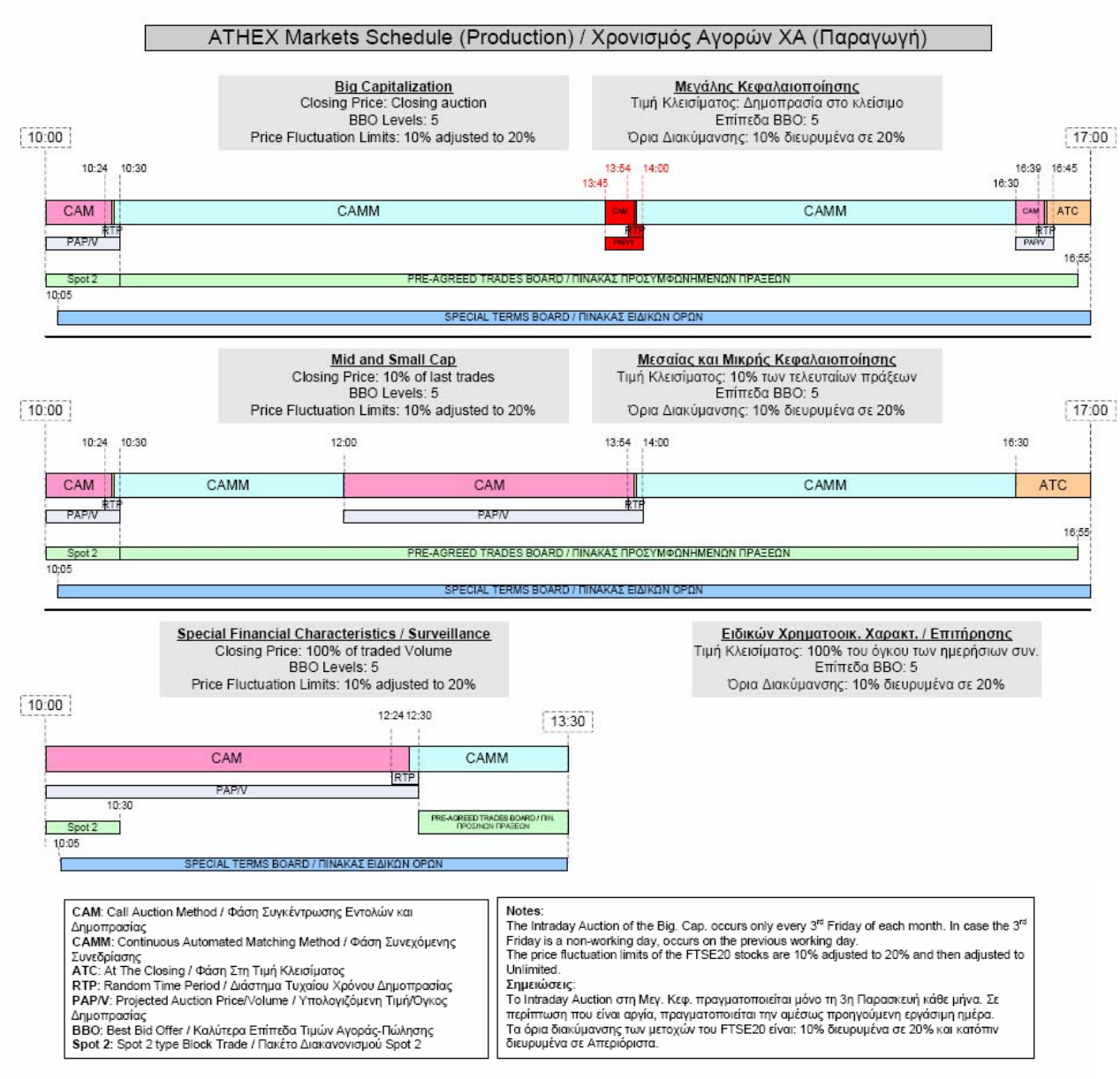


Abbildung 26 – Marktphasen Athen, Quelle [ASE2007]

Anhang D Glossar

Begriff	Definition
Ausführungslatenz	Die Ausführungslatenz des Brokers (kurz: Latenz oder Latency) ist die Zeitdauer von Erhalt bis zur Ausführung, wobei zum Beispiel die Öffnungszeiten der Börsen mitberücksichtigt werden.
Bid-/Ask-Preis	Bid-Preis = Preis des Käufers, Ask-Preis = Preis des Verkäufers
Bid-/Ask-Volumen	Bid-Volumen = Anzahl Stück, welche zum Kauf angeboten werden (analog Ask-Volumen beim Verkauf)
Börse, Markt	Ort, an welchem die Wertpapiere gehandelt werden.
Broker	Drittbank, welche den Auftrag für die UBS an der Börse handelt.
Executor	Routet Aufträge an die Börse (elektronisch oder telefonisch)
Händler	Handelt Aufträge direkt am Markt (Market Maker, Proptrader)
In-/Out-of-limit Minding	Unter In-/Out-of-limit Minding wird das Überwachen der Auftragslimite verstanden. Beispiel: Limitierter Kaufauftrag, Aktienkurs erreicht respektive fällt unter Limite => erst jetzt beginnt die Laufzeit zu laufen.
Kundenberater	Nimmt telefonisch Kundenaufträge entgegen
Marktphasen	Unterteilung in kontinuierlichen Handel und Auktionen (Öffnungsphase, Schlussphase und Stop-Trading; verzögerte Ausführung der Aufträge)
Order Routing	Das Übermitteln eines Auftrags an die entsprechende Börse.
Preisschritt	Der Preisschritt definiert den minimal erlaubten Unterschied zwischen zwei (ungleichen) Preisen in einem festgelegten Wertebereich. Ist der Preisschritt zum Beispiel 0.05, so sind die Preise 0.05, 0.10 oder 1.55 zulässig; die Preise 0.07, 0.48 1.32 sind nicht zulässig. Ab dem Wert 100 könnte zum Beispiel der Preisschritt 1 sein und somit sind nur noch ganzzahlige Preise zulässig.
Spread	Differenz zwischen Kauf- und Verkaufskurs.
Strategie	Grosse Aufträge werden oft mit einer Strategie gehandelt, um den Markt nicht zu stark zu beeinflussen. PV33% = „33% Partizipation des Volumens“: Das Ziel dieser Strategie ist die Ausführung eines Auftrages über einen Zeitraum zu verteilen, in welchem das dreifache Volumen des Auftrages gehandelt wird.
Telefonische Aufträge	Aufträge, welche per Telefon an einen Broker geschickt werden, da ein elektronischer Anschluss fehlt.
Überwacher	Überwacht den gesamten Auftragsfluss

VWAP	Volume Weighted Average Price ([VWAPInvestopedia2007])
Wertschriften	Aktien, Obligationen, Fonds, Optionen usw.
Wertschriftenauftrag, Auftrag	**Beispiel**: Ein Kunde erteilt der Bank folgenden Auftrag: **Kauf** (= Auftragsart) **100 Stück** (= Auftragsgrösse) **UBSN** (= Tickersymbol = Code für UBS Aktie) **bestens** (=ohne Limite) **gültig bis Ende Monat.** An der Börse werden die **100 Stück** à CHF **95.50** (= Ausführungspreis) gehandelt. Der Auftrag ist nun vollständig ausgeführt. Hätte der Kunde eine Kauflimite von CHF 95.00 angegeben, so hätte der Kaufauftrag erst bei oder unterhalb dieser Limite ausgeführt werden können.

Tabelle 7 – Glossar

Anhang E Inhalt der CD-ROM

Datei / Ordner	Beschreibung
diplomarbeit.pdf	Diese Dokumentation (PDF)
iv\demo.iv	Demo eines 3D-Modells der Auftragsdaten (IV)
experiment\dboutlier.xml	RapidMiner Experiment (XML)
experiment\dbscan.xml	RapidMiner Experiment (XML)
experiment\em.xml	RapidMiner Experiment (XML)
src\arff2iv\ARFF2IV.java	Programm um ARFF- in IV-Dateien umzuwandeln (Java)
src\etl	ETL-Programm (Java)
src\prediction\Prediction.java	Vorhersage Prototyp (Java)
rapidminer-4.0beta-tutorial.pdf	Tutorial / Handbuch von RapidMiner[15]

Tabelle 8 – Inhalt CD-ROM

[15] Mit freundlicher Genehmigung von Ingo Mierswa, Managing Director, Rapid-I Mierswa & Klinkenberg GbR

Anhang F Abbildungsverzeichnis

Anhang G Tabellenverzeichnis

Anhang H Quellennachweis

ASE2007	http://www.ase.gr/content/en/MarketData/TimeSpread
AttributWikipedia2007	http://de.wikipedia.org/wiki/Attribut
BestExecutionWikipedia2007	http://en.wikipedia.org/wiki/Best_Execution
ConfusionMatrixWikipedia2007	http://de.wikipedia.org/wiki/Confusion_Matrix
HanKamber2001	Jaiwei Han, Micheline Kamber, Data Mining, Concepts and Techniques. Morgan Kaufmann, 2001
HocheKorgelWrobel2007	Susanne Hoche, Mark-André Krogel, Stefan Wrobel, Otto-von-Guericke-Universität Magdeburg, Institut für Wissens- und Sprachverarbeitung, http://kd.cs.uni-magdeburg.de/data-mining-ueberblick.pdf
InstanzWikipedia2007	http://de.wikipedia.org/wiki/Instanz_(Informatik)
KappaWikiPedia2007	http://de.wikipedia.org/wiki/Cohens_Kappa
MarketImpactWikipedia2007	http://en.wikipedia.org/wiki/Market_impact
RapidMinerTutorial2007	http://downloads.sourceforge.net/yale/rapidminer-4.0beta-tutorial.pdf
VTX2007	http://www.virt-x.com/trading/trading/provisions/exchange_periods.html
VWAPInvestopedia2007	http://www.investopedia.com/terms/v/vwap.asp

Alle URLs wurden am 2.8.2007 geprüft.

Anhang I Bestätigung

Hiermit bestätige ich, Fabian Merki, dass ich die vorliegende Diplomarbeit „Ein Prototyp für die Segmentierung und Klassifizierung von Börsenaufträgen“ im Rahmen der geltenden Reglemente selbstständig ausgeführt habe.

Zürich, den 7. August 2007

Fabian Merki