Data Mining in CRM -
a Case Study for an Online Gaming Company

Project Work submitted in partial Fulfillment of the Requirements
for the Degree of Master of Science in
Information and Media Technologies
at the Technical University of Hamburg-Harburg

Project Work Examiner: Prof. Dr. Ralf Moeller

Submitted by:

Natalya Furmanova
No. of Matriculation: 21044449
Petersburgerstr 36
10249 Berlin
Tel.: 0176 32566764

Hamburg, February 2013

NORTHERN INSTITUTE OF TECHNOLOGY MANAGEMENT

Abstract

Institute of Software, Technology and Systems
Master of Science in Information and Media Technologies
Data Mining Methods in CRM
by Natalya Furmanova

Data Mining and Machine Learning methods have been utilized by businesses in recent years in order to improve Customer Relationship Management (CRM). The problems of new customer acquisition, loyalty and attrition of the existing customers have been addressed in modern research of application of the classification, pattern recognition, clustering techniques to the databases containing vast amounts of data. This paper considers the application of the two methods – Association Rule Mining and Classification by Random Forest – to the data stored by a small Internet Gaming Business, in order to determine patterns in customer behavior that could potentially help improve customer retention and engagement. The paper is split into three parts. Introduction gives overall information about the subject of Data Mining and specifics of Data Mining application in CRM. It is followed by Chapter 1 which discusses the application of Association Rule Learning in order to see the relationship between old and newly introduced products that the company offers. Chapter 2 applies Random Forest Classification to the expanded problem of Chapter 1 in order to consider additional factors affecting customer motivation. The paper is concluded by the discussion of the success of the methods, as well as criticism and suggestion of the future research.

Acknowledgements

I would also like to thank my family and close friends for supporting me and giving me strength to complete the research. I would also like to thank Prof. Ralf Moeller for allowing me to do research on the topic of my choice.

Table of Contents

Abstract I

Acknowledgements II

List of Figures..... IV

List of Tables..... V

List of Abbreviations..... VI

List of Abbreviations..... VII

Introduction..... 1

Chapter 1. Association Rule Learning..... 3

1.1. Business and Problem Understanding..... 3
1.2. Formal Method Description and Literature Review..... 4
1.3. Data Analysis and Understanding 9
1.4. Data Extraction and Transformation 10
1.4.1. Data Extraction..... 10
1.4.2. Data Transformation..... 12
1.5. Modeling..... 13
1.5.1. Frequent Itemsets Discovery 14
1.5.2. Rules Generation with Apriori Algorithm 14
1.6. Rules Strength Assessment with Interestingness Measures..... 15
1.7. Business Conclusions and Recommendations..... 19

Chapter 2. Classification with Random Forest 20

2.1. Problem Understanding..... 20
2.2. Formal Method Description and Literature Review..... 21
2.3. Data Understanding..... 23
2.4. Data Extraction and Transformation..... 24
2.5. Iterative Modeling and Results Assessment..... 25
2.6. Conclusion and Method Limitations..... 31

Conclusion and Possibilities for Future Research 32

Appendix A 33

Appendix B..... 41

Bibliography..... 42

Declaration of Authorship..... 46

List of Figures

Figure 1.4. R Data Structures with Extraction Results.....12

Figure 1.5. Transaction Structure Extract 13

Figure 1.6. Frequent Itemsets Minsupport 0.001..... 14

Figure 1.7. Rules With Support, Confidence, Lift.....15

Figure 1.8. Scatter Plot Rules Vizualization..... 16

Figure 1.9. Rules After Pruning..... 17

Figure 1.10. Rules with Interestingness Measures.....18

Figure 2.4. Prediction Error Performance of the Classifier.....26

Figure 2.5. Training and Test Error Rate of the Classifier.....27

Figure 2.6. Mean Decrease in Accuracy of Predictor Variables.....28

Figure 2.7. Plots of Partial Dependence of Class “1” Outcome.....29

Figure 2.8. Class Prototypes.....30

List of Tables

Table 1.2. Interestingness Measures 6

Table 1.3. Relevant Tables for Business Problem 9

Table 2.1. Independent Variables 24

Table 2.2. Dependent Variables 24

Table 2.3. Important Parameters of randomForest Function.....25

List of Listings

Listing 1.1. Apriori Algorithm Pseudocode..... 6

List of Abbreviations

CRM	Customer Relationship Management
DM	Data Mining
RF	Random Forest
AR	Association Rules

Introduction

Data Mining (DM) is described as the process of searching for meaningful patterns in large amounts of data, stored predominantly in databases [1]. A science on the crossroads of Information Technology, Database Management, Mathematics and Statistics, it applies innovative analytical methods to transform raw data into meaningful associations, predictions and dependencies. Data Mining has found its applications in a wide range of disciplines, from Medicine to Security to Pattern Recognition. Machine learning is a term sometimes used interchangeably with Data Mining, since it represents an area of Artificial Intelligence where learning from data is involved. Data Mining process uses a lot of techniques of Machine Learning and applies them to the Large Databases in search of patterns. Research has been mounting in the recent years in the application of DM methods in the area of Customer Relationship Management. The models help discover cross-selling opportunities, predict customer churn, identify the most profitable customers, among multiple other applications. Manhart [2] divides the DM Methods into groups according to the business purpose: analysis of dependencies between transactions (Basket Analysis, Sequential Patterns), Separation of Customers into homogenous groups (Clustering), Modeling and Rules Definition (Decision Trees-based methods and Neural Networks) and Missing Features Completion (Neural Networks and Sequential Patterns Analysis). The methods are being applied based on the nature of the data and the business goals: such as increase customer loyalty, improve retention or new customer acquisition. Thus, Data Mining serves as a way of acquiring of the business insights from the vast quantities of data stored in the companies databases.

The next chapters of this paper discuss the application of two Data Mining and Machine Learning methods – Association Rule Learning and Random Forest Classification – to the CRM challenges of the Online Gaming company (Sauspiel GmbH). In Chapter 1, Association Rule Learning, applied to the problem of finding interdependencies

between the purchases of new and established products, is discussed and evaluated. In Chapter 2, Classification by Random Forest model is described and analyzed as it is applied to the modified problem of Chapter 1, where the emphasis is put on finding additional factors affecting the customer activity. The structure of discussion in Chapters 2,3 follows the process outlined by CRISP-DM methodology [3], adjusted in order to avoid redundancy in describing parts common to both problems. Conclusion assesses the results of the methods application and discusses future research possibilities.

Chapter 1. Association Rule Learning

1.1. Business Problem Understanding

Sauspiel GmbH is a company that provides online product (online social casual card gaming) as freemium and subscribtion product. Alongside with the free-of-charge games there are cash games and premium games (a part of premium subscription – with some additional features like participation in the leagues). Cash gaming, as one of the main sources of profit for Sauspiel, is a high priority and thus represents the biggest interest for this research. Cash games are offered online in real time, with four players possible at each online playing table. One user can participate in one game at a time. The commission gets paid to Sauspiel has increased its freemium product offering in June 2012 by introducing "cash tournaments". Tournaments are similar to the regular cash games in the fact that the participants buy into the tournament, thus providing the basis for the payout of the winners. Tournaments differentiate themselves by the large number of participants creating a significantly bigger payout pool, and the fact that it consists of many games played in real time on elimination principle. On the seller's side, the difference is that commission is paid to Sauspiel from the buy-in (10% of the buy-in). Tournaments are offered once a week.

The company has experimented with two buy-in prices for the tournaments – 5€ and 10€. Predominantly, the price of 5€ has been used. Buy-in of 10€ has been sporadically used and free tournaments infrequently take place as the promotional or "thank you" event. However, the management sees the ambiguity in the effects of participation in differently priced tournaments on the motivation of the users to play the simple cash games afterwards. There is a possible dependency between playing variously-priced tournaments and further gaming activity (until the next weeks tournament). While the tournaments with an entry fee might encourage further playing due to the experienced

excitement, the promotional free tournaments potentially motivate users who previously have only played the free version of the game, to try the paid product (cash games). Another factor is that unlocking the user account for cash gaming with Sauspiel due to signing up for a tournament removes the psychological barrier to further use the account for normal cash games. If such dependencies are confirmed analytically, the tournaments have potential to become a powerful up- and cross-selling tool [14] and a catalyst for increasing the cash gaming activity, bringing in increased revenue.

In order to analyze the dependencies discussed above, a fitting initial approach would be to consider the participation in games as product purchases within many customer transactions and explore such transactions with the help of Basket Analysis (also known as Association Rule Learning). Such analysis might potentially highlight the correlations between the purchases of different products and attempt to prove the causative relationship, which in turn would help answer the business question: “does playing a tournament with a specific buy-in price cause playing cash games for a certain percentage of users?”. Although the problem as is does not represent a classical boolean Basket Analysis problem yet (due to temporal peculiarities of the data), it can be mapped to such by a series of transformations. The next parts of the chapter discuss the theoretic underpinnings of the Association Rule Learning method and the step-by-step application of it to the business problem described.

1.2. Method Description and Literature Review

Association Rule Mining (Basket Analysis), also referred to as Association Rule Learning, is one of the unsupervised Data Mining methods CRM benefits from. It has found applications predominantly in the retail industry – for instance, supermarkets or online retailers – where the products are purchased in various combinations. The method uses the purchase transactions data from the retailer's database in order to find out which products are consistently purchased together, as well as the direction (antecedent and consequents) in the resulting sets. Pioneering work on this model has been done by R. Agrawal, who defines it in the following way [4]:

"Let $I=\{I_1, I_2, \ldots, I_m\}$ be a set of n binary attributes called items. Let T be a database of transactions. Each transaction t is represented as binary vector, with $t[k]=1$ if t bought item I_k, and $t[k]=0$ otherwise. There is one tuple in the database for each transaction. Let X be a set of some items in I. We say that a transaction t satisfies X if for all items I_k in X, $t[k]=1$. *Association rule* is an implication of the form $X \Rightarrow I_j$, where X is a set of some items in I, and I_j is a single item in I that is not present in X. The rule $X \Rightarrow I_j$ is satisfied in the set of transactions T with the confidence factor $0 \leqslant c \leqslant 1$ *iff* at least *c%* of transactions in T that satisfy X also satisfy I_j". The classical Association Rule Mining process consists of two distinct steps:

1. "finding frequent itemsets $Y=\{I_1, I_2, \ldots, I_k\}, k \geqslant 2$ (groups of items that appear together in the percentage of transactions set by the threshold titled minimal support),
2. generating the rules for each Y comparing the proportion of support of Y to support of $X \subset Y$, the potential antecedent of the rule (a subset of the itemset that contains all the items in the *itemsets* excluding the potential consequent) with the minimal *confidence* level threshold *c* set by the miner. All the inferences of the type $X \Rightarrow Y - X$ are the resulting association rules that satisfy confidence *c*. The confidence measure can be described in terms of conditional probability: it reflects how many transactions contain item I_i given they also contain I_j " [4].

Association Rule Learning uses Apriori algorithm, which essentially performs the breadth-first search of the transaction database in order to find all itemsets containing products that are purchased together and determine the itemsets in left hand side and right hand side of the association. The algorithm is described in the pseudocode in Listing 1.1 [5]

A number of other algorithms have been developed in order to improve performance of the modeling process, such as Eclat – a depth-first search algorithm [6,7], OPUS search [8] and others.

$Apriori(T, minsupport)$
$L_1 \leftarrow \{large\ 1-itemsets\}$
$k \leftarrow 2$
$while\ L_{k-1} \neq \emptyset$
$C_k \leftarrow \{c | c = a \cup \{b\} \wedge a \in L_k - 1 \wedge b \in \cup L_{k-1} \wedge b \notin a\}$
$for\ transactions\ t \in T$
$C_k \leftarrow \{c | c \in C_k \wedge c \subseteq t\}$
$for\ candidates\ c \in C_t$
$count[c] \leftarrow count[c] + 1$
$L_k \leftarrow \{c | c \in C_k \wedge count[c] \geq minsupport\}$
$return \bigcup_k L_k$

Listing 1.1. Apriori Algorithm pseudocode [5]

Academic literature distinguishes three types of Association Rule Learning: boolean, quantitative and temporal :

- *Boolean Rules* – the rules that determine if the association between two classes (products) exists at all.
- *Quantitative Rules* – associations between quantities of products that occur together in the database
- *Temporal Rules* – rules that discover associations with some temporal order, for instance "purchase of one item precedes purchase of another item by two days" association.

There are also combinations of the above three types, such as "Temporal Quantitative Rules".

Another classification of the types of Association Rules is Intra-transactional (associations occurring within one transaction), or inter-transactional (associations between transactions occurring at various points in time) [9].

A lot of research has been performed in the area of assessment of the quality of associations, or “interestingness”. [10]. *Support*, *Confidence* and *Lift* are the three basic and most widely used measures that assess association rules and are defined in the probabilistic terms the following way [4]:

Support (s):

$$s = P(B, A) \quad (1)$$

Confidence (c):

$$c = P(B|A) \quad (2)$$

Lift (l):

$$l = \frac{P(B|A)}{P(B)} \quad (3)$$

However, as multiple authors underline [10,11,12,13], the above measures might be ineffective and biased if the dataset's structure is imperfect (for instance, highly unbalanced sets or rare appearance of important items). In order to approach this problem, other measures have been introduced as well as algorithms based on these measures (as opposed to support-confidence algorithm).

The quality of Associations can be assessed via multiple indicators. The interestingness measures serve both as the inputs to algorithm (*support*, *confidence*) and in order to assess the strength of rules already after they have been derived. A number of such measures is displayed in Table 1.2 in the probabilistic notation. Those have been most commonly used and are relevant to data sets with different nature. For example, unbalanced datasets (with very few representatives of some classes and many of others), suffer from a certain bias in interestingness measures as well as require different *minsupport* parameters for the algorithm. To address this issue, modifications have been applied to the classic algorithm (for example, multiple *minsupport* measures for various classes) [12] and the new interestingness measures have been developed, as described in the paper by Hashler and Hornik [13]. Some of the measures displayed in Table 1.2 are important for the investigation of this paper and will be utilized in the further research

of this chapter and are marked with (*).

#	Measure	Formula
1	ϕ – coefficient *	$\frac{P(A,B)-P(A)P(B)}{\sqrt{P(A)P(B)(1-P(A))(1-P(B))}}$
2	Odd's ratio (α) *	$\frac{P(A,B)P(\neg A,\neg B)}{P(A,\neg B)P(\neg A,B)}$
3	Yule's Q	$\frac{\alpha-1}{\alpha+1}$
4	Yule's Y	$\frac{\sqrt{\alpha}-1}{\sqrt{\alpha}+1}$
5	Kappa (kappa)	$\frac{P(A,B)+P(\neg A,\neg B)-P(A)P(B)-P(\neg A)P(\neg B)}{1-P(A)P(B)-P(\neg A)P(\neg B)}$
6	Mutual Information (M)	$\frac{\sum_i\sum_j P(A_i,B_j)\log(\frac{P(A_i,B_j)}{P(A_i)P(B_j)})}{min(-\sum_i P(A_i)\log(P(A_i)),-\sum_j P(B_j)\log(P(B_j)))}$
7	J-Measure	$P(A,B)\log(\frac{P(B\|A)}{P(B)})+P(A,\neg B)\log(\frac{P(\neg B\|A)}{P(\neg B)})$
8	Gini index (G)	$P(A)\left[P(B\|A)^2+P(\neg B\|A)^2\right] + P(\neg A)\left[P(B\|\neg A)^2+P(\neg B\|\neg A)^2\right]$
9	Laplace (L)	$\frac{NP(A,B)+1}{NP(A)+2}$
10	Conviction *	$\frac{P(A)P(\neg B)}{P(A,\neg B)}$
11	Interest (I)	$\frac{P(A,B)}{P(A)P(B)}$
12	Cosine (IS) *	$\frac{P(A,B)}{\sqrt{P(A)P(B)}}$
13	Piatetsky-Shapiro's (PS)	$P(A,B)-P(A)P(B)$
14	Certainty Factor (F)	$\frac{P(B\|A)-P(B)}{1-P(B)}$
15	Added Value (AV)	$P(B\|A)-P(B)$
16	Jaccard (Gamma)	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$
17	Klosgen (K)	$\sqrt{P(A,B)}(P(B\|A)-P(B))$

Table 1.2. Existing Additional Interestingness Measures [10], besides *Support*, *Confidence* and *Lift*. The measures used for this research are marked with (*).

1.3. *Data Analysis and Understanding*

A look at the organization of the data in the Sauspiel database has identified that the transactions are represented by a number of tables and fields. In order to select the tables containing the needed information, the author of this paper has studied the database schema and discovered that there were two tables responsible for cash (and simple) gaming by User ID that allowed to directly extract the number of games played per user per day. In order to find the information about the participation in tournaments, two tables needed to be considered (one responsible for the details of the tournament such as date and buy-in type and one detailing the IDs of users who took part in each tournament with a specific ID). Table 1.3 outlines the detail of the selected tables and fields.

Initial querying of the data has discovered several peculiarities. Firstly, the data has temporal character (the games are played consequently by a user, on a given day or week). In order to perform Basket Analysis, the transactions would have to be merged into "mega-transactions" of the length 1 week, similar to the method applied in the research by Rana et. al [15]. The rationale for choosing the transactions to be one week long is that each week one tournament occurs, and such organization of data would allow to consider the purchases starting with the day of the tournament and 6 consequent days that playing the tournament could influence. Next tournament would follow in exactly one week and would put a start to the next "mega-transaction" (and so on). Such "mega-transactions" would allow to map the problem from temporal to a static one, since everything that happens within one week would be considered as a purchase within this transaction. Since the pace of cash gaming is very fast (a game lasts on average around several minutes), it is reasonable to take the lowest granularity level possible in this case, i.e. 1 week as opposed to considering 2-week-long transactions which would contain 2 tournaments and all the games in between.

Secondly, there is an imbalance in the amount of the tournament games of each type

(ratio of 2:2:16) that have already occurred. This can have potential effect on the usage of Apriori algorithm and the choice of interestingness measures. And thirdly, in the course of each week the types of tournaments played exclude each other. However, since the dependencies between different types of tournaments are irrelevant for this research, this peculiarity does not significantly impact the experiment.

Table	Field	Type	Meaning
tournaments	tournament_id	Integer	ID of the tournament in the table
	updated_at	Date	Date of the occurrence of the tournament
	buy_in	Integer	Price to enter the tournament
	balance_type	Integer	Type of the tournament (real money or premium)
tournament_users	user_id	Integer	User id of the given tournament player
	tournament_id	Integer	Tournament ID
daily_rankings	user_id	Integer	User ID
	games_played	Integer	Amount of games played by the user on a given date
	balance_type	Integer	Type of games counted for a given statistical record
	start_of_period	Date	Date of the given statistical record

Table 1.3. Relevant Tables for the Business Problem

1.4. Data Extraction and Transformation

1.4.1. Data Extraction

The data has been extracted based on the following limitations:

- only users who played any kind of cash tournaments have been included into the sample
- time period: June 6, 2012 – September 9, 2012 (from the introductions of tournaments to the date of the data extraction)
- products considered for Basket Analysis: free games, cash games, tournaments

with different buy-ins (10,5,0€)

The rationale for the above limitation is as following: excluding the users who have never played tournaments does not add any informative transactions to the set, because it is not possible to infer the dependency involving playing tournament if the tournament has not actually been played. Besides, reducing the number of transactions would increase performance of the algorithm. The five products have been selected for the experiment in order to address the scope of the business problem (for instance, subscription gaming and tournaments have not been taken into account due to a very small number of users who actually participate in both subscription and cash gaming).

The data has been extracted using R statistical software with the use of library representing the SQL interface in R (*RMySQL, RPostgreSQL*). For tournaments data, the data has been extracted into two data structures, *tinfo* and *tuser* – for tournament information and tournament users list, correspondingly (tables *tournaments*, *tournament_users* correspondingly). These structures are a direct result of the SQL query. Besides the *tinfo* and *tuser* tables, a sequence of amounts of games played has been extracted for each *user_id* in *tuser* table with a daily frequency within the test period. Structure named *umatrix* was created to contain the unique ids of the tournament users, the date of first tournament. Structures *echtgeld_seq, seqnorm* contain number of games played per user per day for cash games and free games correspondingly. They have been extracted using unique *user_id* of tournament users , by quering the table *DailyRankings* with *balance_type*=0 for simple games and *balance_type*=2 for cash games. Figure 1.4 shows the samples of the R data structures that represent the extracted and transformed data. The extraction code can be studied in Appendix A.

After extracting the data into the initial structures, additional processing was necessary in order to:

- transform the data about tournaments played by each user into similar structures as *echtgeld_seq* and *seqnorm*.
- represent the data in the transactions-like mode in order to be able to input the data into the existing R function that calculates the frequent itemsets.

```
    D1  D2  D3  D4  D5  D6  D7 D8  D9 D10 D11 D12 D13 D14 D15 D16 D17 D18  ...  id type
101 287 123 129 235 169  14 278 72  89 142 190  99  52  28  34 217  59  51 ...  6736   2
102   0   0  17  16   0   0  18  0  21   0  10  17   0   0   0   0   4  14 ...  6740   2
103   0   0   0   0   0   0   0  0   0   0   0   0   0   0   0   0   0   0 ...  6794   2
104  41  68  57  50  35   0   0  0   0   0   0   0   0   0   0   0  19  16 ...  6873   2
105 171 174 191  65 135 137  93 44 114 340  69  34 125 167  35 308 278   0 ...  6955   2
```

```
  id          updated_at buy_in
6 14 2012-06-07 17:23:11    500
1  6 2012-06-08 20:58:33   1000
```

Figure 1.4. Example rows of the R data structures containing extraction results (upper - *echtgeld_seq* , lower - *tinfo*)

1.4.2. Data Transformation

Since the Boolean Associations were selected as a method in order to trace the initial correlations between the products, the transactions have to be of boolean type ("true" (or 1) for the products that were bought in the purchasing transaction and "false" (or 0) for the products that were not bought). The set of transactions can be represented by a matrix-like structure where rows represents transactions, columns represent products. In order to transform the data into a structure appropriate for the boolean associations mining task, there need to be applied several steps of transformation:

- create "mega-transactions" of length 1 week from the temporal sequences acquired during the Data Extraction phase,
- convert quantities of the games played into the boolean values (if a number of games was played by a user during that week, replace with "1", if no games were played – leave "0"),
- present the data as the list of transactions that contain some combination of the five products mentioned (three types of tournaments, cash games, simple games) for each *user_id* for each week of the sample.

By means of applying the algorithm the code for which can be seen in the Appendix A, the initial data has been transformed in the structure seen on Figure 1.5.

```
     a b c d e
[1,] 1 0 0 0 0
[2,] 0 0 0 1 1
[3,] 0 0 1 0 0
[4,] 0 0 0 0 1
[5,] 1 0 0 0 1
     .........
```

Figure 1.5. Transactions Structure Extract

The transactions data structure uses the following schematic names for its columns:

a – *tournaments with 10€ buy-in,*

b – *tournaments with 5€ buy-in,*

c – *free tournaments,*

d – *free games,*

e – *cash games.*

The total amount of transactions in the list is approximately 15000. The list represents boolean transactional data where each row is a "mega-transaction" of length one week, starting with the day one of the tournaments occurs and including the following 6 days (until the next tournament occurs). Within this "mega-transaction" a user buys a certain combination of products (or plays certain kinds of games). With the acquired structure it becomes possible to mine for the boolean Association Rules.

1.5. Modeling

Modeling is performed in the R language with the help of library *arules*. The library implements the classic Apriori algorithm via counting through an unbalanced prefix tree [16], allowing for effective and quick search. The two-step process looks for frequent itemsets based on the *minsupport* limitation given by the miner, then induces the rule based on the *minconfidence* .

1.5.1. Frequent Itemsets Discovery

The first step of the algorithm is executed using the function *apriori* from the arules library, with the option of finding frequent itemsets. The function takes the list of transactions as the main input and finds sets of items that occur in the database more often than a certain threshold, defined by *minsupport* input parameter. Due to the problem of the unbalanced and sporadic representation of tournaments in the set, the approach is to define separate *minsupports* for each class analytically. For the algorithm, the *minsupport* of 0.001 has been used in order to be able to encompass all the possible itemsets. The resulting frequent itemsets are depicted on Figure 1.6. According to the list, frequent itemsets including each of the three types of tournaments and cash games have been found by the algorithm (set 7, 9, 11), as well as one more specific set 13, which includes three items. Correspondingly, support values of the itemsets are proportional to the frequency of the appearance of each type of tournaments. However, there is a need for some established measure to be used that would account for unbalanced classes representation. The current dataset fits the "rare meaningful associations" problem, as described by Selvi and Tamilarasi [12], and thus support is allowed to be relatively low, whereas emphasis is put on the *confidence* in the next step.

```
   items     support
1  {a}     0.011549877
2  {c}     0.045167118
3  {b}     0.081558911
4  {d}     0.167053813
5  {e}     0.242999097
6  {a,d}   0.001613111
7  {a,e}   0.006775068
8  {c,d}   0.009872242
9  {c,e}   0.008968899
10 {b,d}   0.011033682
11 {b,e}   0.039811589
12 {d,e}   0.018583043
13 {b,d,e} 0.003806943
```

Figure 1.6. Frequent itemsets satisfying *minsupport*=0.001 (R Output)

1.5.2. Rules Induction With Apriori Algorithm

In the second step of the algorithm, the function *ruleInduction()* of the R library *arules* is used. This function takes *minconfidence* as one of the input parameters, together with the itemsets collection from Step 1 (Figure 1.6). The output of the function is a collection of the association rules with their *confidence* and *lift* values. The amount of itemsets from part one. The *minconfidence* parameter has progressively been set lower in a series of experiments down to 0.1 with the purpose of analyzing the rules resulting from the algorithm. The goal was to see not only the stronger correlations but also the weak ones involving the tournaments. The resulting rules are displayed on the Figure 1.7. The rules are now ready to be analyzed and pruned.

```
   lhs      rhs     support  confidence      lift     itemset
1  {a} =>   {d} 0.001613111  0.1396648    0.8360468       6
2  {a} =>   {e} 0.006775068  0.5865922    2.4139686       7
3  {c} =>   {d} 0.009872242  0.2185714    1.3083893       8
4  {c} =>   {e} 0.008968899  0.1985714    0.8171694       9
5  {b} =>   {d} 0.011033682  0.1352848    0.8098277       10
6  {e} =>   {b} 0.039811589  0.1638343    2.0087849       11
7  {b} =>   {e} 0.039811589  0.4881329    2.0087849       11
8  {d} =>   {e} 0.018583043  0.1112399    0.4577789       12
9  {d,e} => {b} 0.003806943  0.2048611    2.5118176       13
10 {b,d} => {e} 0.003806943  0.3450292    1.4198787       13
```

Figure 1.7. Rules with *support*, *confidence* and *lift* (R output)

1.6. Rules Strength Assessment with Various Interestingness Measures

The set of rules depicted on Figure 1.7 contains rules derived with a two-step apriori algorithm based on the criteria *minsupport*=0.001 and *minconfidence*=0.1, in order to get an overview of the various possible rules. However, only some of those rules can be considered meaningful or interesting. First, the rules are analyzed with the common three measures: *support*, *confidence* and *lift*. Figure 1.8 visualizes the rules as a scatter plot on the 2-dimensional plane (x axis representing support, y- confidence) and gradient shading represents the 3rd dimension (lift). Visually three stronger and a group of weaker rules can be distinguished. The weaker rules are concentrated in the bottom left corner of the graph (low confidence, low support, low lift besides one rule). The three rules that stand out are located in the top left part of the plot and, bottom right part and top right part. After examining the values and the rules on Figure 1.8 corresponding

to the points of the scatterplot, the rules that appear stronger in comparison with others, are Rule 2 ({*a*} => {*e*}), Rule 7 ({*b*} => {*e*}) and Rule 6 ({*e*} => {*b*}).

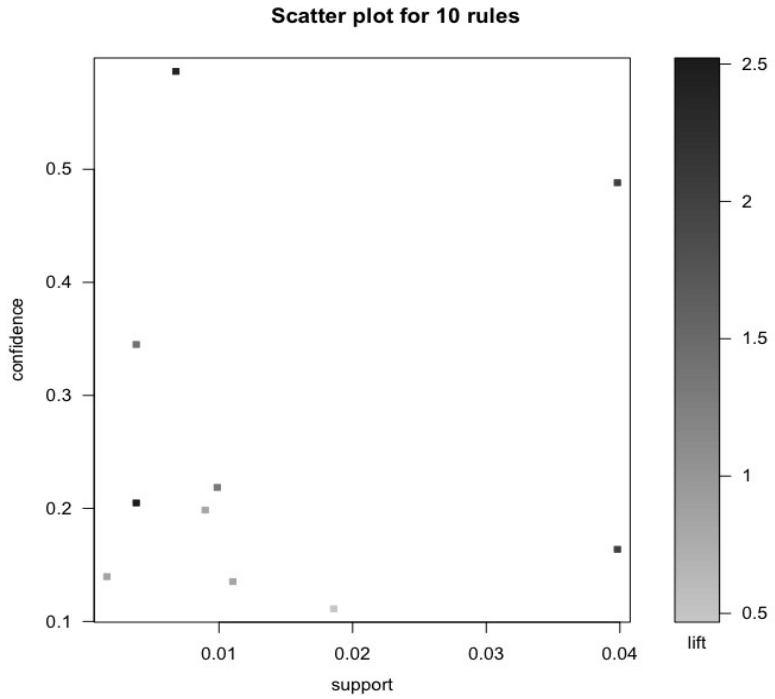


Figure 1.8. Scatter Plot Rules Visualization (R Output)

Out of three rules singled out, all have either significantly higher confidence than the others - Rule 2 ({*a*} => {*e*}), Rule 7 ({*b*} => {*e*}) - or significantly higher support Rule 6 ({*e*} => {*b*}), Rule 7 ({b} => {e}). As mentioned before in this paper, support of the rule (or the frequency of appearance of all parts of the rule together in the database) is bound to be biased to reflect irregularities of the data. The Rule 6, however, also has a low confidence of just 0.16, which makes it a candidate for pruning.

As a result of the stepwise analytical pruning, two rules are left – Rule 2 and Rule 7 – which both imply the possible correlation or cause relationship between tournaments of with buy-in 5€ and cash games and those of tournaments with buy-in 10€ and cash games. The use of additional interestingness measures is helpful to assess the further meaning and significance of the rules. Figure1.9 shows the two rules left and the values of their three interestingness measures.

	lhs	rhs	support	confidence	lift
2	{a} =>	{e}	0.006775068	0.5865922	2.4139686
7	{b} =>	{e}	0.039811589	0.4881329	2.0087849

Figure 1.9. Rules after Pruning

The two rules on the Figure 1.9 show relatively strong confidence; however, the appearance of left hand side of the rule together with the right hand side of the rule might be a correlation (chance relationship). In order to assess this, lift is a helpful interestingness measure. According to formula (3), lift is a proportion of conditional probability of right hand side given left hand side exists, to the probability of right-hand-side itself. Thus if $P(B|A)=P(B)$, the conditional probability is as high as the probability of appearance of the item itself, implying that the presence of left-hand-side does not boost the probability of the appearance of the right-hand-side. This indirectly implies the chance relationship between left-hand-side and right-hand-side (they happen to be in the same transactions). The causation power deteriorates further if the division result is less than 1. This value suggests, to the opposite, that the presence of the left-hand-side decreases the chance of appearance of the right-hand-side item (the probability given left-hand-side is actually lower than the simple probability of the appearance of the item by itself in the transaction). Thus, only lift higher than 1 guarantees a meaningful rule. As seen on Figure 1.9, the Rules 2 and 7 have lift 2.41 and 2 correspondingly, which proves some level of dependency, with Rule 2 having stronger dependency than Rule 7. However, there is a need to mention that due to the rarity of item {a} itself in the database the lift value can get inflated [13]. The rules need to be additionally analyzed with the help of other interestingness measures in order to confirm the findings. The results of calculations of various interestingness measures for the rules are outlined in the Figure 1.10 (part 1, part 2).

	rule	confidence	lift	conviction	chiSquare	cosine	coverage
2	{a} => {e}	0.5865922	2.413969	1.831124	116.2214	0.1278859	0.01154988
7	{b} => {e}	0.4881329	2.008785	1.478901	449.5740	0.2827948	0.08155891

Figure 1.10. Various Interestingness Measures (part 1)

```
          rule       doc     hyperLift   fishersExactTest oddsRatio        phi
2      {a} => {e} 0.3476079  1.842105     7.159535e-23    4.518371    0.08659747
7      {b} => {e} 0.2669021  1.809384     2.787590e-87    3.356943    0.17031886
```

Figure 1.10. Various Interestingness Measures (part 2)

A look at the Figure 1.10 suggests the following (using Table 1.2):

- *conviction* > 1 for both rules: both display relationship between sides [17]
- *chiSquare* >> 3.85 indicate that left-hand-sides and right-hand-sides are not independent (whereas it seems that there is more dependence in Rule 7 ({b}=>{e}), confirmed by Fisher's Exact Test (a statistical significance test, p-value); p-value is much smaller for Rule 7, again;
- *odds Ratio* > 1 (higher for the first rule), indicating that the odds of finding transactions containing right-hand-side given left-hand-side are higher than those not containing left-hand-side; also indicates degree of relationship;
- *hyperLift* > 1 for both rules (a hypergeometric measure that accounts for the low counts of the left-hand-side [13]), with the Rule 2 having a bigger value;
- *doc* (difference of confidence) > 0 for both rules, with the first rule having a higher difference. (according to formula $conf(X \Rightarrow Y) - conf(\neg X \Rightarrow Y)$. [18]) ;
- $\varphi > 0$, [10] for both rules, with φ(Rule 7) > φ(Rule 2), indicating positive corellation (however, not indicating cause-and-effect);
- *cosine* (Rule 2) < *cosine* (Rule 7) reflects the fact: *support* (Rule 2) < *support* (Rule 7);
- *coverage*(Rule 1) < *coverage* (Rule 7); coverage is a *support* of the *left-hand-side*; this is also a statistical significance measure confirming the rarity of the rules involving item {a}.

The analysis shows that a lot of correlation and causation measures indicate the precedence of Rule 1 over Rule 2; however, any statistical-significance-related measures indicate that strength of statistical confidence is on the side of Rule 2. In other

words, Rule 1 is stronger than Rule 2, but Rule 2 can be seen as more believable, considering the sample size.

1.7. Business Conclusions and Recommendations

The rules, induced and analyzed in the previous parts of the chapter, can be translated into the business language as following:

1. Tournaments with buy-in 10€ appear to affect the cash gaming activity in the week following the tournament; moreover, tournaments with buy-in 5€ also have such an effect. However, the effect from the tournaments with buy-in 10€ appears stronger, as proven by several probabilistic measures. It is worth mentioning that the relationship between tournaments with buy-in 5€ enjoys higher statistical assurance.
2. In order to both increase assurance that the rule holds for the general population of users and improve cash sales, it is advisable to increase the amount of offerings of tournaments with buy-in 10€. This will allow to receive a bigger sample and iteratively improve the answers of the Association Rule Mining experiments.
3. Free tournaments do not appear as a motivating factor for the consequent short-time cash gaming. This is not a strong promotional or upselling tool, as evidenced by low proportion of transactions containing both free tournaments and cash gaming.

The answers ignite more questions, such as search for the other factors that motivate cash gaming, given the fact that a user played a tournament. Association Rule Learning in the way it has been performed, tries to determine relationship between product purchases using mathematical tools, without consideration for other factors. The next chapter will try to approach several factors in combination by exploring prediction techniques.

Chapter 2

Random Forest Classification

2.1. Business Problem Formulation and Rationale For Choosing the Method

Upon analyzing the outcome of the Association Rule Mining experiment from Chapter 2, conclusion has been made that discovery of associations between user activity in tournaments realm and cash games realm should be treated with caution and supported by an approach that looks at the problem from the more holistic perspective. Additional factors, intrinsic and extrinsic, might possibly affect the motivation of the players. Thus, the second DM problem has been shaped as the natural continuation of the problem formulated in Chapter 2: to examine which other factors might affect the motivation of the users to play cash games besides the mere fact of participating in the tournament. The problem is limited to a short term task (one week of activity following the tournament), similar to the problem of association rules. The scope of the problem has been additionally narrowed down to the first tournament played by each user in order to study the behavior in the beginning of the cycle of the product usage.

The Random Forest classification method has been chosen for modeling the solution to this problem firstly for its proved effectivity for classification tasks, and secondly due the ability of the method to assess the importance of the predictors (to be described in the next part of this Chapter). This feature of Random Forest Classification is significant for the focus of the business problem and thus makes a strong argument for the choice of this method.

2.2. Formal Method Description and Literature Review

Classification is one of the best-known Data Mining methods. Also referred to as Supervised Learning, its goal is to "build a concise model of the distribution of class labels in terms of predictor features and use a resulting classifier to assign class labels to the testing instances where the values of predictors are known but value of the class label is unknown" [19]. Classification is an iterative process consisting of the following steps:

- acquire a sample containing representatives of various classes (with class labels) and features of the classes,
- divide the sample into "training" and "test" set,
- build classifier by modeling the relationship between the class label to the features (predictors) in the training set,
- test the prediction effectivity of the classifier by applying the model to the test set in order to determine the class labels.

Each outlined step is crucial to the success of classification. One of the success factors is properly chosen and collected data. Zhang et al [20] point out the necessity for careful selection and preprocessing of the data due to the dangers of noise and missing feature values, among other factors. Selecting the appropriate features is equally central for a strong data foundation of an accurate classifier – Yu et al [21] explore the process of selecting the relevant features and removing redundant ones in order to find the subset most suitable for training the classifier. The most important step of the classification is arguably selection of the supervised training algorithm. There is a number of classification algorithms described in the research literature [22]: logic-and-rule-based (Decision Trees, Rule Induction), perceptron-based (Neural Networks, surveyed by Zhang [22]), and statistical algorithms (Bayesian Networks, described by Jensen [23], and Support Vector Machines, described by Hearst et al [24]). In the recent years, interest to ensemble classifiers (classifiers consisting of the combination of the classifiers) has grown significantly due to possible benefit of increased classification

accuracy. Kotsiantis [22] mentions various types of ensembles:

- using various training parameters with one method and assigning weights to the misclassified training instances (boosting),
- using two or more classifying methods together (voting),
- using various training with the same algorithm using different subsets of the sample (bagging).

Research of this Chapter of the paper focuses on one of the popular ensemble classification methods which uses the two latter strategies of the previous paragraph. Random Forest Classification (RF) algorithm was introduced by Breiman [25]. The classifier is an ensemble of decision trees, where each tree is built by using a subsample of the training set generated by sampling with replacement (bagging). In addition, the features to be selected at each split of the decision tree are also sampled using bagging principle. Finally, the class label assigned to an instance is decided based on the majority of votes of the trees in the forest. Combining these strategies has allowed Random Forest classifiers to become one of the most efficient classifiers, rivaling Support Vector Machines and Neural Networks. An important peculiarity of Random Forest method is its ability to assess the importance of each predictor used in classification, giving it additional advantage over other methods.

Random Forest has found application in various subject areas. Wu and Li research Gene Expression from Microarray Data by means of Random Forest [26], Bernhardt et al use RF for a hand-written digit recognition task [27], Xu et al apply the technique in the Terrorist Profiling [28], to name a few. In the area of Customer Relationship Management RF has been innovatively used to assess the customer future loyalty and retention based on the past activity and other known factors about the customer. The interpretation of the method in the research by Bart Larivière and Dirk Van Den Poel [29] has inspired and served as a basis for the experiment described in this Chapter.

2.3. Data Understanding

The step of Data Understanding in the context of the modified tournaments problem is performed in cooperation with the subject-matter expert in order to determine potential factors that might affect the user's motivation for playing cash games shortly within taking part in a tournament. The users are split into two classes: those who play cash games after the tournament and those who do not. The following variables have been identified as potential predictor features:

- age of the user,
- gender of the user,
- type of the tournament played
- pay-off as a result of a victory in the most important rounds of the tournament
- cash games participation in the past.

The initial analysis of the Sauspiel database has determined that all the above dimensions are available in the database, with the exception of one: age of the user. The users are prone to leaving this field blank unless the field is mandatory. For the reason of the high proportion of unknown age among tournament-playing users, the initially identified predictor group has been narrowed to 4 latter dimensions. These dimensions represent both innate characteristics of the user and the previous purchasing behavior of the user.

One of the preconditions for creating a successful dataset for RF classification is to insure that the features are not correlated. Due to the fact that the algorithm samples a number of features, the correlation between the dimension variables is able to cause bias and redundancy in the algorithm itself and rating the importance of the variables [30]. With this consideration in mind, the chosen variables have been compared and no dependencies have been found. For the same reason, other potential variables have been

removed from consideration. The examples include frequency of playing cash games before the tournament and wins in the previous cash games.

2.4. Data Extraction and Preparation

In order to form the sample dataset for data mining, the time constraint (tournaments occurring June 2012 through September 2012) has been used, with the total number of tournament participants (instances) equal to 1936. In order to acquire the features and class labels for the instances, tables users, profiles, tournament_users, tournaments, daily_rankings, rake_transactions have been accessed by SQL queries. A matrix of records of size 1936x5 has been achieved by transformation and cleansing of the results of the SQL queries. The four first columns representing independent and dependent variables (features) are described in Tables 2.1 and 2.2 correspondingly.

Independent Variable Name	Description	Possible Values	Type
type	type of tournament played, per buy-in	0,5,10	categorical
gender	gender of the user	male,female	categorical
payoff	whether a user won a certain Payoff in the tournament	1 or 0	categorical
echtgeld_before	whether a user played echtgeld games Before the played tournament	1 or 0	categorical

Table 2.1. Independent Variables

Dependent Variable Name	Description	Possible Values	Type
echtgeld_week	whether a user played echtgeld in the week following the tournament	1 or 0	categorical

Table 2.2. Dependent Variable

The values have been converted to categorical format ("factor" in R), to reflect the meaning of each variable (gender, presence of payoff, existence of previous gaming and the type of buy-in represent a category rather than the actual number or string). One half of the rows (by random selection) have been allocated into the training set, with the rest

forming the test set. The complete R script for extraction and transformation of sample data can be examined in Appendix B.

2.5. Iterative Modeling and Results Analysis

For modeling with Random Forest algorithm R package *randomForest* has been utilized. The package contains a number of robust methods implementing Breyman's ensemble classification algorithm and tools for analysis and visualization of the results. The function randomForest from this package performs the modeling by iteratively producing trees and testing the classifier in place. RandomForest takes the training and test set as input parameters, alongside with some tuning parameters and options for additional calculations. The parameters are mentioned in Table 2.3 and are used for the experiment.

Parameter Name	Description
ntree (tuning parameter)	number of trees in the model
mtry (tuning parameter)	number of independent variables randomly sampled during each node split
importance (additional option)	calculate predictor importance?
proximity (additional option)	calculate distance between each pair of instances?
norm.votes (representation option)	Normalize the tree voting?
do.trace (additional option)	trace the outcome of each tree?

Table 2.3. Parameters of randomForest Function

In order to determine the values for tuning parameters, a series of experiments has been performed with increasing numbers of trees (*ntree*) and various numbers of dimensions to sample in each node of the individual trees (*mtry*). Since Random Forest is an incapsulated model (“black box”) and relies heavily on randomization, experimental approach is considered optimal for determining the best parameters [25]. Random Forest uses unpruned decision trees offset by other optimization (voting), thus eliminating the need to determine the pruning criteria. Sampling with replacement is a

random process performed automatically by the algorithm without human interference. The experiments have been started by growing *mtry* from 2 to 4, and *ntree* from 20 to 500. Another argument in favor of such a high number of decision trees is the unbalanced nature of the dataset (instances with tournaments of type 0 and 10 occur rare in the dataset and are thus less likely to be a part of the out-of-bag samples). The out-of-bag label prediction error has been at its lowest with *mtry*=4 and has been stable with the *ntree=500* .Based on the above criteria, the results achieved with these parameters have been accepted as optimal for the given setup. Figure 2.4 illustrates the relationship between prediction error rate and the number of trees in the forest.

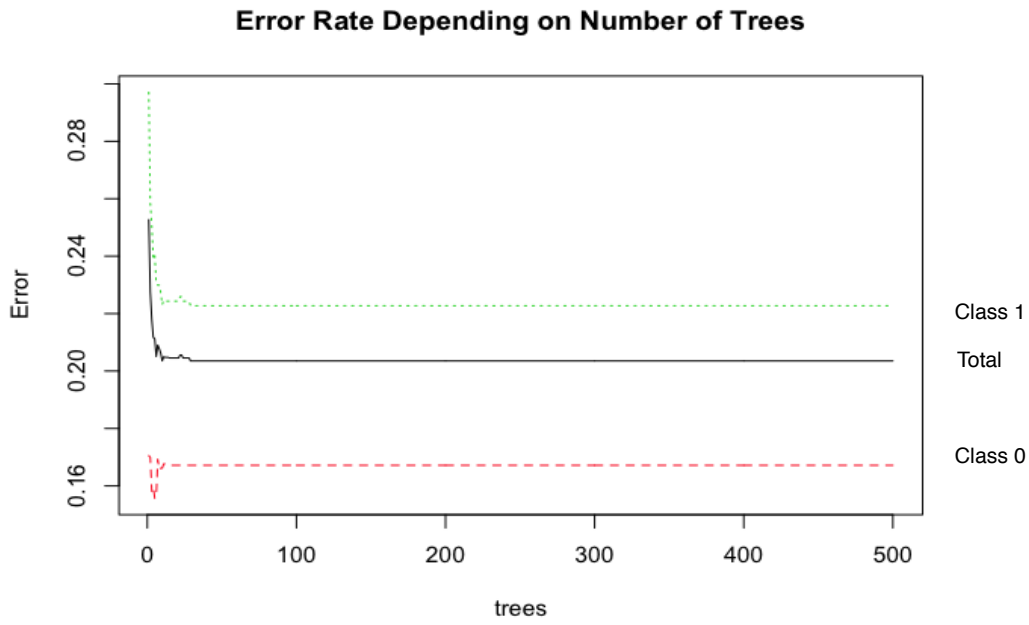


Figure 2.4 Prediction Out-of-Bag Error Performance of the Classifier (R Output)

It can be seen from Figure 2.4 that the prediction error of the sample taken "out of bag" (training set) has had significant fluctuations up until approximately *ntree*=30. It has been since then unchanged (stable). However, without loss of performance – the algorithm performed relatively quickly with a large number of trees due to a small number of features – the author of this research has decided to include 500 trees in order to account for the size of the training set, since every tree is built with a bootstrap sample from the initial training set. Thus, higher number of trees provides for a higher chance for all of the members of the sample to be in the bootstrap sample.

As mentioned earlier, Random Forest is a “black box” algorithm, meaning the process of classifying is not straightforward for humans or easy to encompass as it happens. However, it is possible to extract partial information which helps shed light on the efficiency and the inner workings of the algorithm. The randomForest function contains output components such as error estimate (by class), prediction confusion matrix, individual tree structures and votes, proximity matrix for instances and estimated importance of each of the variables in predictions. Armed with this information, the miner is able to interpret the results of the classification and identify the reasons for the flaws of the classifier. The visualization of the example decision tree that could be in a random Forest classifier can be examined in Appendix B. However, randomForest algorithm does not care for pruning of the trees. According to Breiman [25], the trees must be grown to the largest extent possible in order to reduce the prediction bias.

The author of this research has found it suitable to start the analysis with considering the error estimates. As seen on Figure 2.5, the prediction error rate hovers around 20% for both out-of-bag and test data. The error rate of the out-of-bag data is expected to be slightly higher due to the bias in the data caused by sampling with replacement: some instances are repeated in the out-of-bag samples (on average one third of the values), which is supported by the results of the experiment. It is noticeable that class “0” (users who do not play cash games during the week following the tournament) is predicted more accurately than class “1” (users who do play cash games) – 15% vs 24% in test set. The latter seem to be misclassified into the former more often than the other way around, reflecting the bias towards class “0”.

```
OOB estimate of  error rate: 20.35%
        Confusion matrix:
     0   1 class.error
0 279  56   0.1671642
1 141 492   0.222748
Test set error rate: 19.94%
     Confusion matrix:
     0   1 class.error
0 330  56   0.1450777
1 137 445   0.2353952
```

Figure 2.5. Training and Test Error Rate of the Classifier (R Output)

The classifier created by the modeling, although does not reach very high accuracy, still shows results significantly higher than chance (50%). This leads the researcher to believe that the random forest has been able to identify a significant part of the dependencies between features and class labels, and that the chosen features do have predictive strength in determining the class. The next step of the analysis is to assess the importance and the affect of the variables on the strength of the classifier. Accessing the output parameter *importance* allows to see the various measures of importance of the variable:

- absolute values derived by the random forest algorithm (by class),
- Mean Decrease in Accuracy of the prediction upon permuting the values of the specified predictor while building the trees [31],
- mean decrease in Gini Index or quality of a split for each variable [32, p.129]

While the former measure is native to the algorithm calculations and does not represent illustrative information in terms of numbers, the two latter measures reflect common and understandable concepts. Gini gain has been acknowledged by the creator of the method as the less illustrative [31] which is why this research uses on the Mean Decrease in Accuracy measure for assessing the power of the predictors. Figure 2.6 contains the scaled Mean Decrease in Accuracy values for each predictor variable used for modeling.

Variable	MeanDecreaseAccuracy
gender	0.003445548
type	0.254297948
payoff	0.003542329
echtgeld_before	0.011913323

Figure 2.6 Mean Decrease in Accuracy of Predictor Variables (R Output)

The importance of the predicting variables can thus be interpreted in the following way: type of the tournament played has significantly higher effect on the following week's behavior of the user than any other values, followed by the *echtgeld_before* variable

(previous cash gaming activity of the user), which has a significantly lower power compared to *type,* yet is much stronger predictor than both *payoff* and *gender.* This

finding, besides strengthening the previous discoveries, highlights that out of the other three additional factors identified by the data and business experts, only one indeed contributes to the gaming motivation: the already existing experience of playing cash games. Gender and the outcome of the tournament for the user, to the opposite, do not play a significant role in this particular scenario. In order to further illustrate the marginal dependence of the actual outcome (class label = "1") on the particular categorical value of the predictors, the plotting function of the R package *randomForest*, *partialPlot*, is used. Figure 2.7 shows the marginal effects for two important variables – *type* and *echtgeld_before*.

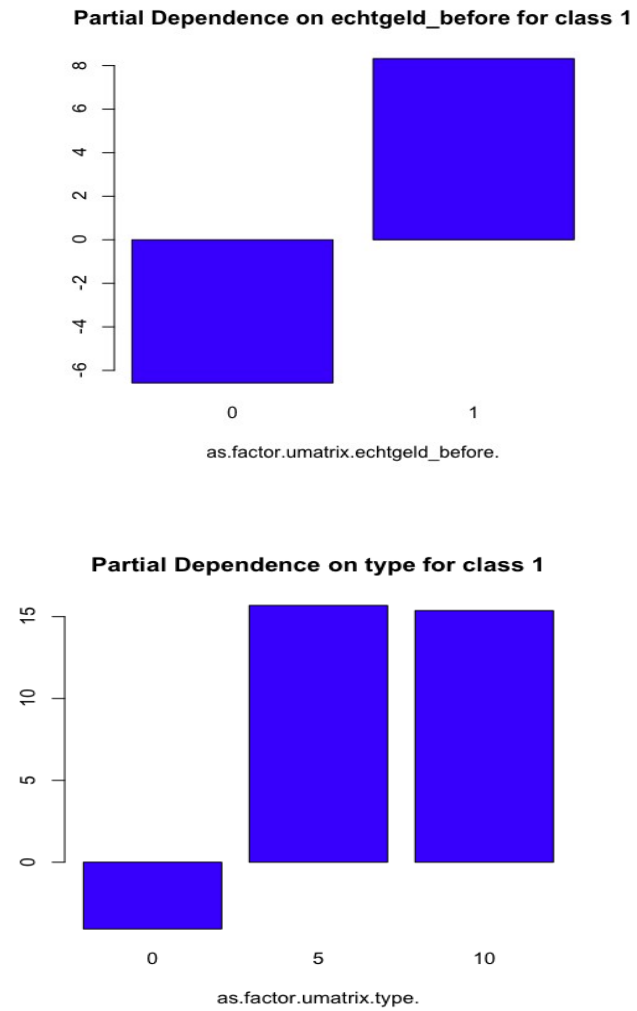


Figure 2.7. Plots of Partial Dependence of Class "1" Outcome (R Output) - *echtgeld_before* Variable (top) and *type* Variable (bottom)

As seen on the graphs of Figure 2.7, previous cash gaming activity (*echtgeld_before*=1)

has a strong positive influence on the prediction of positive week's activity, given the participation in the tournament with a buy-in of 5€ or 10€ (*type*=5 or *type*=10) has occured. The value of *echtgeld_before*=0 has a negative marginal effect on prediction in favor of this class, as well as the tournament *type*=0 (free tournament).

To conclude the analysis, class prototypes for each class have been derived using the function *classCenters* of package *RandomForest*. Seen on Figure 2.8, the prototypes represent the combination of the feature values, most "typical" for each class, or the most observed combinations within each set.

class	gender	type	payoff	echtgeld_before
0	"male"	"0"	"0"	"0"
1	"male"	"5"	"0"	"1"

Figure 2.8 Class Prototypes (R Output)

The prototypes are essentially medoids – the class members whose dissimilarity is minimal to all other members of the class [37]. They are calculated using the proximity matrix (which was prepared by setting the *proximity* parameter of the function *randomForest* to TRUE). Proximity matrix is a matrix of distances between the points representing members of the training set, where each distance between every two cases is the number of times the both occur in the same class (terminal node of the tree), normalized by the number of trees [25].

Visualizing an "ideal" user who displays either behavior of class "0" or of class "1" supports and ties together the previous findings of this experiment. However, attention must be directed to the fact that there can be multiple class centers and the algorithm of the classCenter function is limited to one and the conclusions cannot be drawn by looking only at one prototype per class: for instance, users with value *type*=10 have and almost equal chance of being classified into class "1" by the model, as seen on Figure 2.7.

2.6. Conclusion and Method Limitations

The results of the Random Forest Classification have several business implications. Firstly, the model has highlighted the strength of the effect of type of the tournament played on the motivation of the user to take part in cash gaming in the following week. Tournament with 5€ buy-in has slightly won over tournament with 10€ buy-in, although this result might be biased due to the low proportion of the latter tournaments. Secondly, it has been confirmed that for users who have not used the paid version of the game, organizing free tournaments is not an effective promotional strategy: free tournaments, although rewarded with a pay-off in case the participant wins, still do not significantly motivate the user to start cash gaming. Additionally, it must be noted that the scope of this experiment has considered only the first tournament played by each user and short-term prediction, without looking into the future possibilities. Lastly, the experiment has proved that neither gender, nor the positive payoff have a significant effect on the further gaming motivation, contrary to what has been hypothesized. Thus, enabling targeted marketing, increasing the winning pool or similar strategies would not produce significant positive results, such as revenue increase from cash gaming.

The experiment results also highlight some of the shortcomings, both of dataset and the algorithm. Firstly, the predictive accuracy has not reached optimal levels, such as 90%. The error rate can be attributed to missing explicit and latent predictors, which are not recorded in the database or have been missed by the subject-matter experts (such as weather conditions, for example). Secondly, the Random Forest Classification performs at its best with large numbers of predictive features, due to the sampling of the features in the decision-trees. Although out of scope for this research due to time limitations of the project, these correlated flaws can be addressed in the future iterations of the model.

Conclusion and Possibilities for Future Research

Two Data Mining methods have been applied to a real-life business data within the research exercise described in this paper: Association Rule Mining and Random Forest Classification. The experiments have been developed in order to provide data-based analytic approach to directing the Marketing strategy of Sauspiel GmbH, an Internet Freemium Online Gaming business, in relation to a new recently introduced product. Broad research of the academic literature both in fields of Customer Relationship Management and Data Mining has served as a basis for the methods choice and application. The results of the experiments, described in the previous chapters, have highlighted the differences between the intuitive perception of the customer motivation and the results of mathematical modeling. While the first experiment (Association Rule Mining) has shown dependencies between one type of tournaments and the consequent participation of cash games, the second one (Random Forest Classification) has highlighted the effect of the first purchase in conjunction with several other factors, and has confirmed the strong association between tournaments and consequent cash gaming.

The research has provided the analytical foundation for steering the direction of Pricing and Marketing at Sauspiel and has opened the possibilities for future iterations of Data Mining experiments. The possible directions constitute broader predictions using Random Forests, as well as application of Temporal Data Mining techniques to study user behavior and throughout lifetime, make predictions of customer retention and cluster customers according to their usage patterns.

Appendix A

R Scripts for Data Extraction – Association Rules

```
#
# Author: Natalya Furmanova
# Script for extraction and transformation of
# data for Association Rule Mining
# October 15 2012
#


library(DBI)
library(RMySQL)
library(cluster)
drv <-dbDriver("MySQL")
con <- dbConnect(drv, user = [redacted],password = [redacted],
 port = [redacted],host =[redacted], dbname = [redacted])



#find all tournaments (echtgeld) and their entry price, and date
tinfo<-dbGetQuery(con,"SELECT id,updated_at,buy_in FROM tournaments
 WHERE balance_type=2")
tinfo<-tinfo[1:18,]

#find all the tournament users by tournament
tuser<-dbGetQuery(con,paste("SELECT user_id,tournament_id FROM
 tournament_users WHERE tournament_id IN
 ( ",paste(unlist(tinfo$id),collapse=","),")",sep=""))
uniq_uids<-sort(unique(tuser$user_id))

#prepare cutoff date

cutoff_date<-as.Date(as.character(tinfo$updated_at[1]))-7
#2012-06-01


#initialize sequence matrix!
dates_seq<-
 seq(from=cutoff_date,to=as.Date(as.character(max(tinfo$updated_at)))
 +7,by=1)

umatrix<-data.frame(id=uniq_uids,
```

```
 echtgeld_before=0,sequence=0,first_t_date=Sys.Date())


echtgeld_seq<-
 data.frame(mat.or.vec(length(umatrix$id),length(dates_seq)))
for (i in 1:length(umatrix$id)) {

#date of first tournament for each user
umatrix$first_t_date[i]<-
 as.Date(as.character(tinfo$updated_at[which(tinfo$id==tuser[which(tuse
 r$user_id==umatrix$id[i])[1],2])]))


# echtgeld games for each player
umatrix$echtgeld_before[i]<-ifelse(dbGetQuery(con,paste("SELECT
 COUNT(*) as c FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period < ' ", paste(umatrix$first_t_date[i],collapse=",") ,"
 ' )",sep=""))==0,0,dbGetQuery(con,paste("SELECT SUM(games_played) as c
 FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period < ' ", paste(umatrix$first_t_date[i],collapse=",") ,"
 ' )",sep="")))


 for (j in 1:length(dates_seq)) {

          echtgeld_seq[i,j]<-ifelse(dbGetQuery(con,paste("SELECT
 COUNT(*) as c FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period = ' ", paste(dates_seq[j],collapse=",") ,"
 ' )",sep=""))==0,0,dbGetQuery(con,paste("SELECT SUM(games_played) as c
 FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period = ' ", paste(dates_seq[j],collapse=",") ,"
 ' )",sep=""))[1,1])
 }

}

seq2<-echtgeld_seq

seqnorm<-data.frame(mat.or.vec(length(seq2$id),length(dates_seq)))

#
for (i in 1:length(seq2$id)) {
     for (j in 1:length(dates_seq)) {
               seqnorm[i,j]<-ifelse(dbGetQuery(con,paste("SELECT
 COUNT(*) as c FROM daily_rankings WHERE (user_id=",
 paste(seq2$id[i],collapse=",") ," AND balance_type=0 AND
 start_of_period = ' ", paste(dates_seq[j],collapse=",") ,"
 ' )",sep=""))==0,0,dbGetQuery(con,paste("SELECT games_played FROM
```

```
daily_rankings WHERE (user_id=", paste(seq2$id[i],collapse=",") ," AND
balance_type=0 AND start_of_period = ' ",
paste(dates_seq[j],collapse=",") ," ' )",sep=""))[1,1])
    }
}

#need to convert to tuples where there are events and temporal aspect


# tournaments

#1) first - take the dates of the tournaments

echtgeld_seq$id<-umatrix$id
echtgeld_seq$type<-2 # for balance_type=2, 10210 for $10 tourn, 1025
 for $5 tourn, 1020 for free tournaments...
seq_mini2<-echtgeld_seq[3,740]
seq_mini10<-seq_mini2
seq_mini10[,1:101]<-0
seq_mini10$type<-10

# lets combine tuser and tinfo tables because it will be easier. tuser
 has uid:tid, tinfo tid:tdate:buyin...

#need, for every tuser-tid,date,buyin(transform to 100)

tuser$date<-Sys.Date()
tuser$type<-0

 for (i in 1:length(tinfo$id)) {
    tuser$date[which(tuser$tournament_id==tinfo$id[i])]<-
as.Date(as.character(tinfo$updated_at[i]))
    tuser$type[which(tuser$tournament_id==tinfo$id[i])]<-
tinfo$buy_in[i]/100
}

#now we have a comprehensive table with tusers,tournament dates and
 buyin.

#now we will go through all the dates_seq on which that type of
 tournament is held, and if there is a tuser on that DATE equal to the
 seq_mini10$id[i], put 1 in that users field  with the index of that
 date

for (i in 1:length(seq_mini10$id)) {
 column_index<-
 which(dates_seq==as.Date(as.character(tinfo$updated_at[1])))
 subset<-tuser[which(tuser$date==dates_seq[column_index]),]
 seq_mini10[i,column_index]<-
 ifelse(length(which(subset[,1]==seq_mini10$id[i]))>0,1,0)

 column_index<-
```

```
 which(dates_seq==as.Date(as.character(tinfo$updated_at[2])))
 subset<-tuser[which(tuser$date==dates_seq[column_index]),]
 seq_mini10[i,column_index]<-
 ifelse(length(which(subset[,1]==seq_mini10$id[i]))>0,1,0)

}

seq_mini5<-seq_mini2
seq_mini5[,1:101]<-0
seq_mini5$type<-5

index_vector_in_tinfo<-which(tinfo$buy_in==500)
for (j in 1:length(index_vector_in_tinfo)) {

 column_index<-
 which(dates_seq==as.Date(as.character(tinfo$updated_at[index_vector_in
 _tinfo[j]])))
 subset<-tuser[which(tuser$date==dates_seq[column_index]),]
for (i in 1:length(seq_mini5$id)) {
 seq_mini5[i,column_index]<-
 ifelse(length(which(subset[,1]==seq_mini5$id[i]))>0,1,0)
 }
}

seq_mini0<-seq_mini2
seq_mini0[,1:101]<-0
seq_mini0$type<-0

index_vector_in_tinfo<-which(tinfo$buy_in==0)
for (j in 1:length(index_vector_in_tinfo)) {

 column_index<-
 which(dates_seq==as.Date(as.character(tinfo$updated_at[index_vector_in
 _tinfo[j]])))
 subset<-tuser[which(tuser$date==dates_seq[column_index]),]
for (i in 1:length(seq_mini0$id)) {
 seq_mini0[i,column_index]<-
 ifelse(length(which(subset[,1]==seq_mini0$id[i]))>0,1,0)
 }
}

#create the transactions database
fake_db<-
 data.frame(uid=seq_mini2$id,date=as.Date(as.character(dates_seq[1])),b
 alance2=as.vector(unlist(seq_mini2[,1])),balance5=as.vector(unlist(seq
 _mini5[,1])),balance10=as.vector(unlist(seq_mini10[,1])),balance0=as.v
 ector(unlist(seq_mini0[,1])))

#
#Author: Natalya Furmanova
#This script creates matrix with dimensions a b c d e from raw
 collected data
```

```
#in order to provide a boolean matrix structure
#Date: October 25 2012

library(arules)

transact<-mat.or.vec(length(tinfo$updated_at)*length(seq2$id),5)

dimnames(transact)[[2]]<-c("a","b","c","d","e")

transact<-data.frame(transact)

for (i in 1:length(tinfo10)) {
 index<-(which(dimnames(seq10)[[2]]==tinfo10[i]))
 for (j in 1:length(seq10$id)) {

 transact$a[j+(i-1)*length(seq10$id)]<-seq10[j,index]

 transact$b[j+(i-1)*length(seq10$id)]<-0

 transact$c[j+(i-1)*length(seq10$id)]<-0#

 transact$d[j+(i-1)*length(seq10$id)]<-
 ifelse(sum(seqnorm[j,c(index:index+6)])>0,1,0)

 transact$e[j+(i-1)*length(seq10$id)]<-
 ifelse(sum(seq2[j,c(index:index+6)])>0,1,0)


     }# take from seq10 by userid value where dimname is equal to
 tinfo10 value

#what would the index be...

}
#b=j+(i-1)*length(seq10$id) # b=j*i
b=i*j
print(b)

for (i in 1:length(tinfo5)) {
 index<-(which(dimnames(seq5)[[2]]==tinfo5[i]))
 for (j in 1:length(seq5$id)) {

 transact$b[j+(i-1)*length(seq10$id)+b]<-seq5[j,index]
 transact$a[j+(i-1)*length(seq10$id)+b]<-transact$c[j+(i-
 1)*length(seq10$id)+b]<-0
 transact$d[j+(i-1)*length(seq10$id)+b]<-
 ifelse(sum(seqnorm[j,c(index:index+6)])>0,1,0)
 transact$e[j+(i-1)*length(seq10$id)+b]<-
 ifelse(sum(seq2[j,c(index:index+6)])>0,1,0)


     }# take from seq10 by userid value where dimname is equal to
```

```
 tinfo10 value

}
b=(length(tinfo5)+length(tinfo10))*length(seq5$id)
print(b)
for (i in 1:length(tinfo0)) {
 index<-(which(dimnames(seq0)[[2]]==tinfo0[i]))
 for (j in 1:length(seq0$id)) {

 transact$c[j+(i-1)*length(seq10$id)+b]<-seq0[j,index]
 transact$b[j+(i-1)*length(seq10$id)+b]<-transact$a[j+(i-
 1)*length(seq10$id)+b]<-0
 transact$d[j+(i-1)*length(seq10$id)+b]<-
 ifelse(sum(seqnorm[j,c(index:index+6)])>0,1,0)
 transact$e[j+(i-1)*length(seq10$id)+b]<-
 ifelse(sum(seq2[j,c(index:index+6)])>0,1,0)


    }# take from seq10 by userid value where dimname is equal to
 tinfo10 value

}
transact<-as.matrix(transact)
```

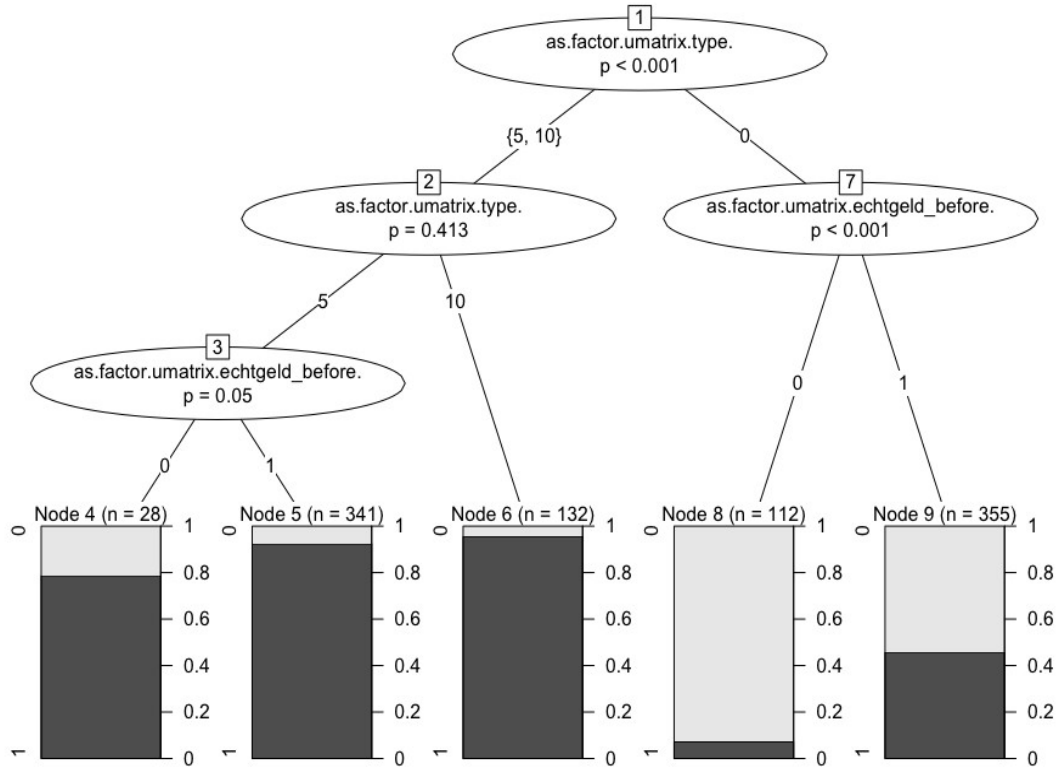
Script for Data Extraction and Transformation for Random Forest Algorithm

```
#Author : Natalya Furmanova
#This script extracts and transforms data into
# a matrix of features and dependend variable
# Date: November 15 2012

for ( i in 1:length(tuser$user_id)) {

 for (j in 1:length(umatrix$id)) {

     if
 ((tuser$user_id[i]==umatrix$id[j])&&(tuser$date[i]==umatrix$first_t_da
 te[j])) {
          umatrix$tournament_id[j]=tuser$tournament_id[i]
          umatrix$type[j]=tuser$type[i]

     }

 }

 }

for (i in 1:length(umatrix$id)) {
 umatrix$payoff[i]=ifelse(dbGetQuery(con,paste("SELECT COUNT(*) FROM
 tournament_users WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND tournament_id=",
 paste(umatrix$tournament_id[i],collapse=","),"  AND payout_data IS NOT
 NULL)",sep=""))>0,1,0)

}


# gender?

for (i in 1:length(umatrix$id)) {
 umatrix$gender[i]=dbGetQuery(con,paste("SELECT gender FROM users WHERE
 id=", paste(umatrix$id[i],collapse=",") ," ",sep=""))

}

umatrix$gender=unlist(umatrix$gender)
```

```
v1=c(which(is.na(umatrix$gender)))

for (i in 1:length(v1)) {
 umatrix$gender[v1[i]]=dbGetQuery(con,paste("SELECT gender FROM avatars
 WHERE user_id=", paste(umatrix$id[v1[i]],collapse=",") ," ",sep=""))



}

for (i in 1:length(umatrix$id)) {

 umatrix$echtgeld_week[i]=ifelse(dbGetQuery(con,paste("SELECT COUNT(*)
 as c FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period >= ' ", paste(umatrix$first_t_date[i],collapse=",") ,"
 ' AND start_of_period <= ' ", paste(as.Date(umatrix$first_t_date[i])
 +6,collapse=",") ," ')",sep=""))==0,0,dbGetQuery(con,paste("SELECT
 games_played FROM daily_rankings WHERE (user_id=",
 paste(umatrix$id[i],collapse=",") ," AND balance_type=2 AND
 start_of_period >= ' ", paste(umatrix$first_t_date[i],collapse=",") ,"
 ' AND start_of_period <= ' ", paste(as.Date(umatrix$first_t_date[i])
 +6,collapse=",") ," ')",sep=""))[1,1])


 }


umatrix$echtgeld_week[which(umatrix$echtgeld_week>0)]=1


umatrix$echtgeld_before[which(umatrix$echtgeld_before>0)]=1

umatrixrf=data.frame(as.factor(umatrix$gender),as.factor(umatrix$type)
 ,as.factor(umatrix$payoff),as.factor(umatrix$echtgeld_before))

echtgeld_week=as.factor(unlist(umatrix$echtgeld_week))
s1=sample(length(umatrix$id),1/2*length(umatrix$id))
x=umatrixrf[c(s1),]
y=echtgeld_week[c(s1)]
xtest=umatrixrf[-c(s1),]
ytest=echtgeld_week[-c(s1)]
```

Appendix B

Example of a Decision Tree in a Random Forest



Bibliography

1. Usama, F., Piatetsky-Shapiro G., Smyth, P. From data mining to knowledge discovery in databases. *Advances in knowledge discovery and data mining*. American Association for Artificial Intelligence, 1996, pp. 1-34
2. Manhart, K. *Analytisches CRM: Methoden und Fallbeispiele*. [Online]. (URL http://www.tecchannel.de/server/sql/1772689/analytisches_crm_methoden_fallbeispiele_business_intelligence/Grundlagen). 2008, October 10. (Accessed 10 September 2012)
3. Shearer C. *The CRISP-DM model: the new blueprint for data mining.* J Data Warehousing, 2000, 5, pp.13—22
4. Agrawal, R., Imielinski, T., Swami, A. Mining association rules between sets of items in large databases. *In*: Buneman, P. and Jajodia, S., eds. *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*. ACM Press, 1993, pp. 207-216
5. Agrawal , R., Srikant, R. Fast Algorithms for Mining Association Rules in Large Databases. *In: Proceedings of the 20th International Conference on Very Large Data Bases*, *September 12-15, 1994,* pp.487-499
6. Zhang, C., Zhang, S. *Association rule mining: models and algorithms.* Springer-Verlag Berlin, Heildelberg, 2002
7. Zaki, M. J. Scalable algorithms for association mining. *IEEE Transactions on Knowledge and Data Engineering,* **12** (3), 2000, pp.372–390
8. Webb, G. (1995) OPUS: An Efficient Admissible Algorithm for Unordered Search. *Journal of Artificial Intelligence Research, **3**, 1995, pp. 431-465*
9. Berberidis, C., Vlahavas, I.Detection and Prediction of Rare Events in Transaction Databases. *International Journal on Artificial Intelligence Tools,*

XX(X), 2007, pp. 1-20

10. Tan, P.-N. et al. Selecting the Right Interestingness Measure for Association Patterns. *In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, July 23-26, 2002.* ACM New York, NY, USA, 2002, pp. 32-41

11. Lallich, S. et al. Association rule interestingness: measure and statistical validation. *Quality Measures in Data Mining.* Springer, 2007, pp. 251-275

12. Kanimozhi Selvi , C.S. and Tamilarasi , A. Mining of High Confidence Rare Association Rules. *European Journal of Scientific Research,* **52(**2), 2011, pp.188-194 – change to 12

13. Hahsler, M. and Hornik, K. New probabilistic interestingness measures for association rules. *Intelligent Data Analysis,* **11**(5), 2007, pp. 437-455

14. Tang, H., Yang, Z., Zhang, P., Yan, H. Using Data Mining to Accelerate Cross-Selling. *In*: *ISBIM '08 Proceedings of the 2008 International Seminar on Business and Information Management, Wuhan, Hubei, China., December 19, 2008,* IEEE Computer Society Washington, DC, USA, 2008, 1, pp. 283-286 .

15. Rana, D. P. et al. Inter Transactional Association Rule Mining using Boolean Matrix. *International Journal of Research in Computer and Communication Technology*, **1**(2), 2012.

16. Borgelt, C. and Kruse, R. Induction of association rules: Apriori Implementation. *In: Proceedings of the 15th Conference on Computational Statistics, Berlin, Germany, 2002.* Physica Verlag, Heidelberg, Germany, 2002, pp. 395-400

17. Brin, S. et al. Dynamic itemset counting and implication rules for market basket data. *In:* Peckham, J., ed. *Proceedings of the International Conference On Management of Data SIGMOD, Tucson, May 13-15, 1997.* ACM SIGMOD, 1997, pp. 255-264

18. Hofmann, H., Wilhelm, A. Visual comparison of association rules. *Computational Statistics,* **16**(3), 2001, pp. 399–415

19. Kotsiantis, S.B. Supervised Machine Learning: A Review of Classification Techniques. *Informatica* , **31**, 2007, pp. 249–268

20. Zhang, S., Zhang, C., Yang, Q. Data Preparation for data mining. *Applied Artificial Intelligence*, **17**, 2003, pp. 375-381

21. Yu, L., Liu, H. and Guyon, I. Efficient feature selection via analysis of relevance and redundancy. *Journal of Machine Learning Research,* 2004, **5**, pp. 1205-1224.

22. G. Zhang. Neural Networks for Classification: a Survey. *IEEE Transactions On Systems, Man, and Cybernetics – Part C: Applications and Reviews*, **30** (4) , 2000, pp. 451-462

23. Jensen, F.V. Bayesian networks. *Wiley Interdisciplinary Reviews: Computational Statistics,* **1** (3), 2009, pp. 307–315

24. Hearst, M.A. et al. Support Vector Machines. *IEEE,* **13** (4), 2004, pp. 18-28

25. Leo Breiman. Random Forests. *Machine Learning*, **45** (1), 2001, pp. 5-32.

26. Xiaoyan Wu. Classification and Identification of Differential Gene Expression for Microarray Data: Improvement of the Random Forest Method. *In: proceedings of* ICBBE 2008, *The second conference on Bioinformatics and Biomedical Engineering*, *16-18 May 2008.* 2008, pp. 763-766

27. Bernard, S., Heutte, L., Adam, S. Using Random Forests for Handwritten Digit Recognition. *In: Proceedings of the Ninth International Conference on Document Analysis and Recognition, ICDAR 2007, September 23-26, 2007*. 2007, **2,** pp.1043-1047

28. Xu, J., Chen, J., Li, B. Random forest for relational classification with application to terrorist profiling. *In: Proceedings of IEEE International Conference on GRC '09, August 17-19, 2009.* 2009, pp.630-633,

29. Larivière, B, Van Den Poel, D.Predicting customer retention and profitability by using random forests and regression forests techniques. *Expert Systems With Applications,* **29,** 2005, pps 472-484

30. Tolosi, L. and Lengauer, T. Classification with correlated features: unreliability of feature ranking and solutions. *Bioinformatics,* **27 (**14), July 2011, pp. 1986-1994

31. Pang, H. et al. Pathway analysis using random forests classification and

regression. *Bioinformatics,* **22** (16), 2006, pp. 2028-2036

32. Zheng Rong Yang. *Machine Learning Approaches to Bioinformatics (Science, Engineering, and Biology Informatics).* World Scientific, 2010, 4.

Declaration of Authorship

I, Natalya Furmanova, declare that this Project Work paper titled, 'Data Mining Methods in CRM – a Case Study for an Online Gaming Company' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this work has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the research is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____