CONFIDENTIAL

Optimizing Payment Network Routing by Refund Prediction

H. van der Voort

Master of Science Thesis

TU Delft Delft University of Technology

Optimizing Payment Network Routing by Refund Prediction

THESIS

submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE
TRACK INFORMATION ARCHITECTURE

by

H. van der Voort
born in Leiderdorp, the Netherlands

Web Information Systems
Department of Software Technology
Faculty EEMCS, Delft University of Technology
Delft, the Netherlands
http://wis.ewi.tudelft.nl

Adyen b.v.
Simon Carmiggeltstraat 6-50
Amsterdam, the Netherlands
https://www.adyen.com

Optimizing Payment Network Routing by Refund Prediction

Author: Huub van der Voort
Student id: 1509446
Email: H.vanderVoort@student.tudelft.nl

Paying at an online shop is easy for a shopper but the process behind it can become rather complex. The actual transaction travels through the systems of multiple different stakeholders like processors, banks and payment schemes. Multiple available stakeholders allow for optimizations. This work focuses on optimization of transaction routing between payment schemes to minimize transaction costs.

Payment routing between schemes is only possible for dual branded payment methods and becomes challenging when schemes differ in functionality. The difference in support for refunds is an example of such a difference which is crucial for retail merchants. In this work a refund predictor for individual transactions is used to predict refund behavior which enables routing between schemes with different support for refunds.

The research can be split into two parts; First it is shown that existing evaluation methods cannot deal with the presented criteria, specific to the routing context. The difference in individual transaction costs seem to be uncovered by all methods. In addition an evaluation method, called Current Optimal Instance Score (cOIS), is defined which is based on the realistic loss function from literature. The proposed evaluation method is evaluated in comparison to the loss function from literature. Choosing the right parameters of a predictor using this new evaluation method improves the performance such that 8% of the costs are reduced.

For the second part a refund predictor is designed which is able to predict if an individual transaction eventually will be refunded when it enters the system. This predictor is used and optimized for route optimization between schemes which differ in refund support and fees. A sample weight strategy is designed to add some weight to transactions which make the biggest difference in costs.

In this research it is shown that transactions costs can be optimized by routing transactions with the knowledge of a refund classifier. Using cOIS, the proposed evaluation method, the final system is validated at a score of 0.487. This result shows that we are halfway in optimizing costs, from the current costs to the theoretical optimum. In practice this system resulted in a cost reduction of 36% which equals to 1.4% of a merchant its profits.

Thesis Committee:
Chair: Prof. dr. ir. G.J.P. Houben, Faculty EEMCS, TUDelft
University supervisor: Dr. D.J.M. Tax, Faculty EEMCS, TUDelft
Company supervisor: Ir. B. Wolters, Adyen b.v.
Committee Member: Dr. M.M. de Weerdt, Faculty EEMCS, TUDelft

Table of Contents

List of Figures

List of Tables

Preface

During my technical study Computer Science at the TU Delft I got more and more interest for the non-technological business side. The master track Information Architecture taught me specific technologies but also how businesses are designed, how to manage IT and policies and most important how to align business and IT. The combination of IT and business intrigued me. The next step was to find a company that has challenges in both these fields. I had been in contact with Adyen after a guest lecture in the Software Architecture course taught by Arie van Deursen. Adyen is a good fit as a technological company operating in a business world which requires continuous business-IT alignment.

Together with the company and my professor I was looking for a research topic that contributed to science and was interesting for both the company and me. The search for a research topic looked surprisingly much at a science, IT and business aligning assignment. After multiple iterations the final subject was constructed. The idea was to use state of the art machine learning tools to solve a small part of a complex business problem. Some thought the idea was to challenging and possibly unfeasible however a quick feasibility study showed that there definitely was some room for a research in this area. After months of research I was able to reduce transaction costs for retail merchants by 36%, estimated at 1,4% of their total profits.

This thesis is completed with the input, assistance and support of many people I would like to thank. First of all I would like to thank my professor Geert-Jan Houben for pointing me in the right direction during the search for research topics and valuable feedback during the halfway presentation. This was the perfect moment to score out some ideas and elaborate and finish the existing ones. My external committee member Matthijs de Weerdt for the constructive and valuable feedback rounds halfway and after green light. My university supervisor David Tax for the almost weekly in-detail discussions about scientific problems and his input for the thesis structure from a research perspective. Moreover, my company supervisor Bert Wolters for day to day support, ideas, feedback and interesting discussions about the thesis concepts and its applications.

In addition I would like to thank Adyen for giving me the opportunity to graduate. My colleagues for teaching me the inner workings of their amazing platform and the payment world, Maikel Lobbezoo for the thorough feedback on the thesis throughout the process and other graduate students for the discussions and joy during the process. The atmosphere in the company is great and the many talks at the barista bar really contributed, even to the

thesis. I enjoyed my time as student and can recommend the company to anyone who is interested to conduct a thesis project in Adyen.

Last but certainly not least I would like to thank my family and friends for supporting me throughout the project. Especially my girlfriend who always tried to help me even when I confronted her with in detail technical problems. Thank you!

Huub van der Voort
Leiden, the Netherlands
March 20, 2015

Chapter 1

Introduction

Performing an online payment is easy for shoppers. The shopper enters 16-digits and an expiry date, and the payment is done. For merchants this can become complex. The network between the shopper and the actual fund transfer can be complex due to multiple (different) stakeholders, configurations, required domain knowledge and changing environments. An introduction to the payment network, including its stakeholders, can be found in Appendix A. This section introduces one specific problem that arises with this complexity; optimizing the route of an individual transaction through the full payment network.

1-1 Payments for Merchants

In the retail market a shopper pays in an on- or 'offline' shop in exchange for products or services. The relation between the shopper and the payment network can be shown by a simple scheme, shown in Fig. 1-1. When a shopper wants to buy a certain product, a payment is initiated at one of the shops of a merchant. The transaction is then processed in a so called 'payment network'. But what does a payment network look like?

Figure 1-1: Simplified overview of a shopper buying goods at a shop.

An overview of the payment network, see Fig. 1-2, shows the relation between different parties. The merchant wants to accept payments for its online orders and connects to the payment network by connecting to a bank, also called acquirer. The acquirer captures and processes the transactions initiated by the merchant. The acquirer bank connects to schemes like VISA or MasterCard. The schemes handle the communication for the fund transfer between the

acquirer and issuer (the bank of the shopper). The issuer authorizes the transaction. A more in detail explanation of the payment network can be found in Appendix A.

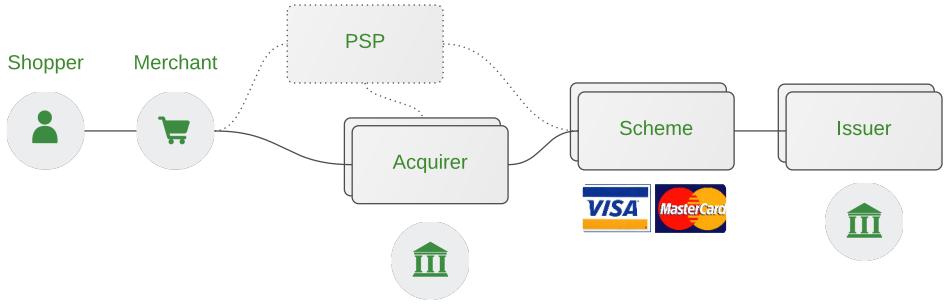


Figure 1-2: Schematic view of stakeholders (and their relations) in Payment Network from a payment perspective.

Every payment method (scheme) requires at least one connection to a supporting acquirer. A merchant generally wants to accept the largest number of payment methods possible to get the maximum amount of revenue [1]. This results in many connections to different acquirers in order to maximize shopper reach and revenue.

The different connections to acquirers comes with extra complexity. Every connection requires different agreements, contracts, a technical connection and all need to be maintained. Most merchants want to focus on their core business instead of these payment related connections. Payment Service Providers (PSP) manage the full payment flow and therefore take the payment complexity out of the hands of a merchant.

1-2 Process Optimization

A Payment Service Provider (PSP) offers solutions to a merchant for the full payment process. These companies are connected to multiple merchants and therefore reusing the available connections to the acquirers. This way worldwide coverage can be achieved by the PSP. Merchants can configure payment methods and acquirers, driven by the insights into payments. Advanced and interactive reports help the merchant decide what configuration will suit its needs.

A PSP actively supports merchants in choosing the best configurations. In general, a PSP earns per transaction, and therefore benefits if a merchant's business grows. Helping the merchant by optimizing and giving insights into its payment process is a win-win situation for a merchant and its PSP. Both parties want to optimize parts of the payment process such as route configuration, current and future connections, technical performance etc. The focus of this thesis is optimizing transaction routing.

Optimization (of payment routes) requires explicit objectives. Transaction routing can be optimized with different objectives. For example, the optimal route regarding the speed of the connection can be different than the optimum regarding the costs. Tradeoffs arise for some optimization questions, for example in fraud control. A route with a lot of security checks will decrease the number of fraudulent transactions, while on the other hand it increases uncompleted transactions from legitimate shoppers. Optimizing transaction routing with

multiple objectives can be a challenging job. The scope of this thesis is limited to the objective of minimizing (transaction) costs.

1-3 Transaction Routing

The complex payment network as introduced in Section 1-1 exists of multiple stakeholders. Parties with connections to multiple stakeholders are privileged with the choice to define a route for its transactions. Fig. 1-3 shows the possibility to choose between the used acquirer and payment scheme. For every transaction, fees should be paid to the acquirer, payment scheme and issuer. Choosing between different payment schemes involves decisions between fees and supported features, while the decision between acquirers involves additional charges, and performance related aspects.

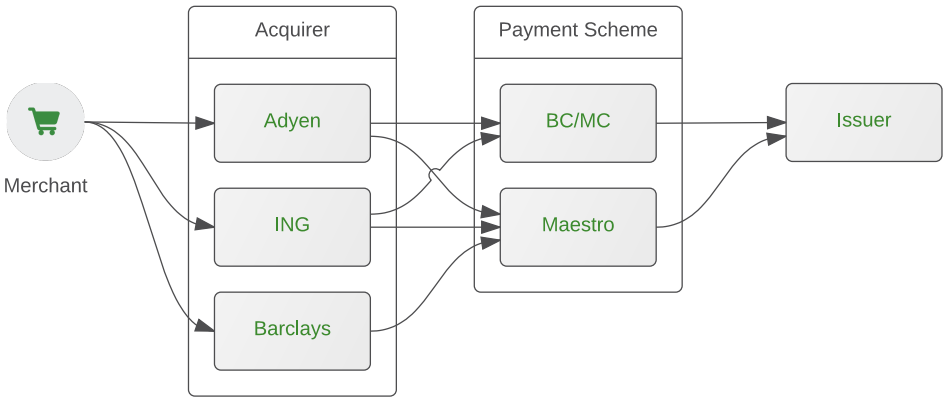


Figure 1-3: Example of route possibilities for transactions in the Belgium payment network

In general, a merchant can decide which route would lead to the highest benefits. An overall decision is made based on information on aggregation level, over all transactions. Such decisions end up in static configurations for multiple payment methods and acquirers. Environmental changes require periodical (manual) revisions of these routing configurations.

Information is lost when deciding on an aggregation level, which results in a semi-optimized configuration. Deciding on aggregation level means that all transactions are bundled together, and the costs are calculated over bundles. In practice, costs are based on individual transaction attributes. Two examples are the transaction amount for blends[1] and the card type for interchange[2]. Different cost functions for processing a transaction allow for costs optimization. For example a certain turning point (for the transaction amount) could exist for blend pricing versus a fixed pricing. Note that next to costs there are more objectives, aspects and interests involved in payment routing on which we do not focus in this thesis.

Acquirer routing with simple strategies are quite common in the payment industry, however routing between payment schemes is not exploited in practice. A calculation example in which the costs of two Belgian schemes are compared is shown in Fig. 1-4. The costs for processing a transaction dependents on the transaction amount, see Fig. 1-4a. To minimize

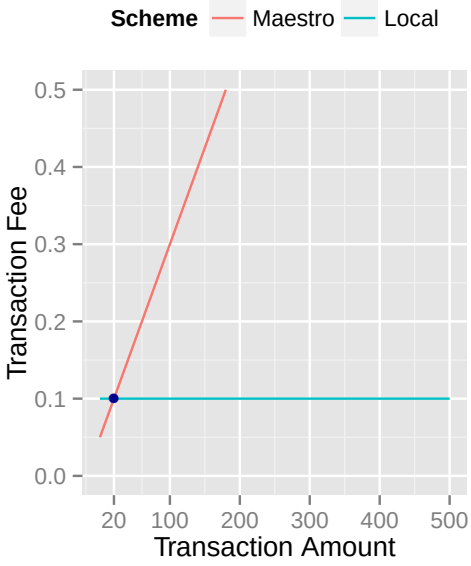
[1] A blend price function is a percentage over the transaction amount.
[2] Interchange is the fee that the issuer charges for using cards issued by this bank.

the costs, transactions should be processed to the least-cost scheme (below and above €20). The example shows that applying route optimization can save the merchant more than €23k a month, a cost reduction of 85%!

Calculating scheme costs
Assume a merchant receives 40k transactions (txs) a month uniformly distributed between €0 and €500 and processes all txns via Maestro. Using the following maestro pricing this costs €27k. The costs per scheme are as follows (shown in Fig. 1-4a):
- Maestro price: $0.050 + 0.0025 \times txn\ amount$
- Local price: 0.10

Optimizing between the routes would lead transactions higher then 20€ to the local scheme. Maestro costs €120 (over 1,6k txs under €20) while Local costs €3.840 (over 38,4k txs above €20) and both sum up to €3.960.

Least-cost routing would save the merchant in a month over €23k, equal to *85% cost reduction*.



(a) Fee for transaction amount grouped by scheme

Figure 1-4: Minimizing transaction costs by using the least-cost route

The calculation example shows that route optimization requires a property of a transaction, namely the transaction amount. A simple system-wide rule can be created to optimize the transactions. Unfortunately, this optimization is 'too good to be true'. In the next section it is explained how we are able to route the transactions.

1-4 Routing in Belgium

Routing possibilities between schemes are not default for every transaction. The ability to choose between payment schemes depends on the choice of payment method, done by the shopper. If iDeal is chosen, the transaction has to be routed via payment scheme iDeal. On the other hand, some cards are dual branded. If a shopper wants to pay with this card, the transaction can be processed via either one of the brands, with the same card details. In this thesis we focus on the Belgium market because Bancontact/Mister Cash (BC/MC) is a payment method which is dual branded and offers scheme routing possibilities.

BC/MC cards are branded with Bancontact/MisterCash and Maestro, because in this way shoppers can pay with Maestro outside Belgium, when shops do not accept BC/MC. The card contains two different chips for both brands, for example a magnetic strip and chip. Dual branded means that a transaction can be processed through one of the two networks (BC/MC or Maestro brand in case of the BC/MC cards).

The situation where two brands exists on one card provides transaction routing opportunities. Although Maestro is intended for international transactions, it can also be used for domestic

transactions, and vice versa. Different price models for processing a transaction allows for costs optimization between payment schemes. Belgium is not the only context where this routing plays an important role in total optimization. There are more actual examples of these route decisions from the industry.

Deciding on a particular brand or route involves not only the costs, but also other characteristics. There are certain route differences, which limits the route decisions. One of these differences is feature support. In Belgium, one route supports refunds, while the other does not. In the next section we will elaborate on this difference.

1-5 Refund Behavior for Routing

The difference in feature support between payment schemes adds a challenge to payment routing. Each scheme could have different features like recurring, refunds, and payouts. A merchant decides if it wants to use a specific feature, this depends on the type of business. The difference in features between schemes can limit the availability of schemes and ability to route dynamically. In this thesis we want to exploit the difference in support for refunds.

A refund is a (partial) return of funds for completed payments and can be initiated by a shopper or merchant. A thorough explanation of refunds can be found in Appendix A-4. Merchants generally want the ability to refund so they can initiate a refund with just 'one press on the button'. Imagine a situation where half of the transactions is refundable, and

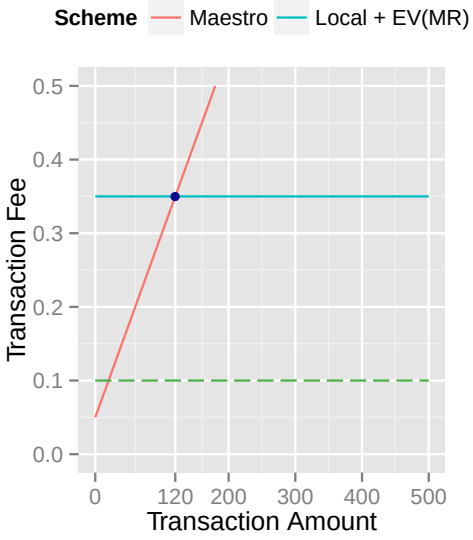
Manual Refunds Let's send the transactions with an *expected lower value* to the local route.

Callcenter staff is used to retrieve account numbers and process manual refunds. Assume that 25% of txs of a retail merchant result in refunds, are uniformly distributed between €0 and €500, and an employee paid €10 per hour can process 10 txns an hour.
- Costs per manual refund (MR): €10 / 10 = €1.00
- Expected Value of txn: $€1.00 \times 0.25 = €0.25$

These extra costs are added to the Local scheme, shown in Fig. 1-5a.

Maestro costs $€1.920$ (over 9,6k txns under €120) while Local costs $€3.040$ (over 30,4k txns above €120), plus manual refund costs of €7.6k. The three add up to €12.560.

Least-cost routing with manual refunding would still save the merchant in a month over €14.4k, which equals to *53% cost reduction*.



(a) Fee (including expected manual refund costs) for transaction amount grouped by scheme. Dashed green line is the Local fee without expected manual refund costs.

Figure 1-5: Minimizing transaction costs with schemes that differ in refund support.

the other half is not. The merchant should then do the refund manually which requires extra money and effort.

Referring back to the calculation sample in Fig. 1-4, transactions via the local scheme, 96% of all, will not be able to be refunded automatically. A more realistic example incorporates manual refund costs and is shown in Fig. 1-5. Least-cost routing decides to send transactions above €120 to the local scheme, which saves this merchant almost €14.4k a month, a cost reduction of 53%. This cost reduction is lower than the ideal example in Fig. 1-4. The overhead of 7,6k manual refunds is in practice not a preferable situation.

The influence in costs of (manual) refunds in payment routing shows the necessity to predict the usage of refunds for a single transaction. To identify if a transaction becomes a refund, refund patterns should be extracted. Identifying refunds is also called classification. In the context of routing, the classification of refunds should be performed when payment is first received. This implies restrictions on the available data which can be used for the pattern recognition task. With dynamic routing the route of a transaction is decided on the fly just before it is routed. Routing individual transactions dynamically is necessary to optimize the merchants transaction costs.

A third calculation sample shows the usefulness of refund prediction, see Fig. 1-6. Instead of routing based on the expected value, a prediction for every transaction determines the route. The Maestro fee crosses the Local fee plus manual refund (not the expected value) at €420. Transactions between €20 and €420 are routed based on their refund prediction. The results of a perfect predictor show that almost 65% of the costs can be reduced. However, in practice such a refund predictor is not expected to perform perfect and the reduction will likely be

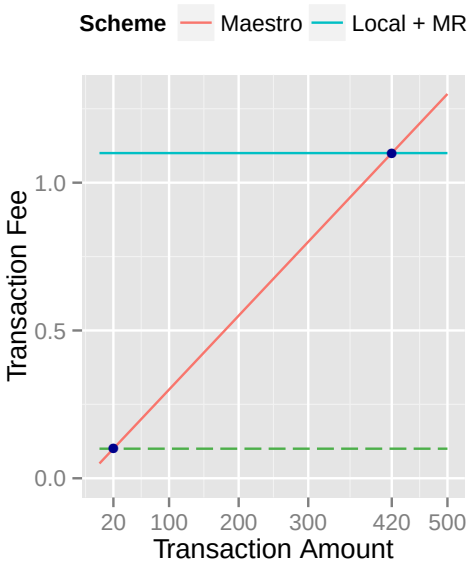
Refund Prediction Predicting refunds allows to route transactions accordingly to the matching routes.

Refund prediction could reduce manual refunds and therefore transaction costs. Assume we can predict refunds perfectly and apply least-cost routing to local, except when refund is predicted.

In Fig. 1-6a the manual refund fee is shown. The intersection shows the relevant section to predict. Transactions with a value between €20 and €420 are routed through prediction.

The costs of transactions before and after this domain are resp. €120 and €2.240, representing 1.6k and 6.4k transactions. The remaining 32k transactions are predicted and routed, resulting in €7.2k costs.

Least-cost routing with perfect refund prediction could save the merchant in a month over €17k, which equals to almost *65% cost reduction*.



(a) Fee for transaction amount grouped by scheme, manual refunds on top of local scheme is added. Dashed green line is the Local fee without manual refund costs.

Figure 1-6: Minimizing transaction costs by routing according to predicted refunds

lower.

In the retail branch shoppers are allowed to return their goods for different reasons. Examples of return reasons are if the size is incorrect, or the color turns out to be different. This could be the reason why the retail branch generally has an higher refund ratio compared to other branches. Information about the bought products, also known as the shopper cart, could most likely identify some refund patterns.

Low-level information can be retrieved from the transaction itself. Transactions can be combined to extract information about velocity and changes in time. Aggregation can be done for different entities, think about acquirers, schemes, categories or users. User history is extracted by aggregating transactions from one user, and is expected to be a valuable information source to identify refund behavior.

This introduction slowly converged to a specific machine learning problem within an interesting financial application field. The main research question in this thesis is the following.

Main RQ: How can we use classification methods to predict refund behavior per transaction in order to optimize payment routing?

Two important concepts can be extracted from this question; first the means, the classification of refunds, and the main goal, optimization of payment routing. The research question and its sub questions are elaborated in detail in Chapter 3.

Identification of refund patterns is an exploratory research. In this thesis we will test the assumption that we can predict the usage of payment features, specifically refunds, based on payment data.

1-6 Industrial Partner

Research in payments requires access to real world data. Adyen is our industrial partner and enables this research by supplying selected datasets. Adyen is a Payment Service Provider (PSP) offering global payment solutions to medium-to-large Merchants. A PSP fills this gap by offering one connection from the merchant (or its webshop) to the PSP which allows the merchant to process transactions via different international and local payment methods.

Multiple services are offered like payment processing, acquiring, risk and reporting. Adyen processes millions of transactions worldwide with currently over 250 payment methods and 85 acquirers. The combination of different services, multiple merchants and reporting over different connections generate lots of valuable data. The ability to do an internship in such a company is a great opportunity.

Adyen partners up with merchants and their drive to increase the number of completed transactions while keeping the fees as low as possible, which is a win-win situation for both parties. Keeping this as a starting point, every merchant should understand why they should adapt transaction routing, what the different opportunities are, and how many this will increase their total revenue, while not being confronted with the complex payment network. This asks for a solid framework capable of mapping the complexity to simplicity, where numbers, insights and suggestions will show a merchant the potentials and guide them in decision making.

1-7 Thesis outline

This chapter introduced some concepts of the payment world. A full description of the payment process and all involved stakeholders can be found in Appendix A. We also described the main challenge and opportunity of Dual Branding and Transaction Routing. The rest of the thesis is structured as follows;

In the next chapter a detailed overview is provided of all related fields and techniques both from industry as well as from scientific literature. In Chapter 3 this research is described in more detail. The gaps of literature and industry are combined into a solid research. The main challenges and research questions which define the rest of the work can be found there.

The first part of the body starts at Chapter 4, where existing evaluation models are tested and critically reviewed. In addition, an evaluation method is introduced in Chapter 5. The evaluation part can be skipped, but make sure to read Section 5-5 and the conclusion of Chapter 5, as the introduced method is used in the next chapters.

The second part starts with the design of the predictor in Chapter 6. This chapter describes the features and classifiers used in the predictor. All experiments are executed in Chapter 7. A baseline for further research and several small experiments are elaborated and combined in a final model.

Finally the contributions and conclusions of the work can be found in Chapter 8. This chapter also contains the limitations and pointers to future research.

Chapter 2

Related Work

2-1 Introduction

As stated in the introduction, predicting refund behavior in payments is the central subject of this thesis. The prediction is then incorporated into transaction routing with minimizing costs as main goal. In this chapter we will describe relevant literature for this problem.

In order to maximize performance out of this predictor, transaction routing should be studied to understand how it will be applied. This section will explain the usage of *transaction routing* in practice. Classification of refunds is, to the best of our knowledge, not yet studied. Therefore a *similar classification* problem[1], fraud detection, will be studied. Both transaction routing and classification shape the artifact created in this thesis, and will be described first in this chapter.

Note that the evaluation of this combination is at least as important as shaping and optimizing the components. Therefore *evaluation techniques* will be studied in Chapter 4.

After explaining both fields in high-level, several detailed fields will be described. The main focus of this thesis is on the classification part, not the optimization part. Therefore we will study some additional fields, applicable specifically to refund routing. The following fields will be described in their own section; *a*) incorporating historical transactions, *b*) handling imbalanced data, and *c*) categorical data encoding.

2-2 Transaction Routing

A transaction can be routed through many different routes. First the merchant can choose the Payment Service Provider (PSP) to process the transaction, second the acquirer, and third the payment scheme or brand, in case of dual branding. In literature, including the documents of some commercial companies described in this section, only the first two routing options are elaborated.

[1] The similarity is in the usage of payment data, and the characteristics that comes with this data.

2-2-1 Changing Environment

The payment network is changing due to, among others, increased demand for mobile and e-commence, distribution of new devices. 'Transaction Routing' is seen as a critical requirement to compete in the payment network[2]. Changes in the network create challenges and opportunities for different stakeholders in the payment network .

According to NewNet Communication Technologies, flexible and new systems should be designed that are able to cope with the implications of the changing payment environment[2]. NewNet provides secure communication technologies in the payment network. The focus of NewNet is on transaction- and network routing due to the different sources, destinations, and changing volumes. However, as a networking company, NewNet does not address objectives from the payment domain, such as conversion rates and external dependencies like authorization rates.

2-2-2 Acquirer Routing

In the payments industry, companies that process transactions are investigating and exploiting possibilities to route transactions via different acquirers with different strategies. There is no unilateral term for this routing concept and in practice this is called multi-acquiring, (intelligent) transaction routing or even fancy names like advanced smart routing. In this thesis, we will call this concept acquirer routing, or transaction routing if not specific to acquirers.

Acquirer routing is possible because merchants or service providers maintain multiple connections to different acquirers which can be leveraged in different ways. Most companies leverage these connections by optimizing the processing costs to be cost-efficient[3, 4, 5].

Different strategies for acquirer routing are being used in industry[6];

- **By Priority** A predefined order in acquirers to send the transactions. If the preferred acquirer fails, the next acquirer in the list will be contacted.
- **Balanced** Divide the transactions between different acquirers following predefined percentage share in terms of volumes.
- **Equal** Transactions are distributed equally over all acquirers.

Using multiple acquirers results in less dependency, therefore more reliability and better timeliness [6]. Previous strategies and their goals show that the industry does not address the ability and potential of routing transactions to different acquirers based on other objectives, such as performance, conversion and authorization rate differences. Transactions can be routed in such a way that multiple objectives are satisfied or even optimized.

Cost-effective routing is currently solved by defining static configurations containing (one or more) rules specifying the route [3], following the priority strategy. Such rules can be based on transactional attributes like scheme, country, recurring, currency and amount, but also if a shopper is a returning shopper or first-time shopper [5]. Using transaction rules help in optimizing profit, costs, conversion, risk exposure and fraud.

Difficulties for rule-based routing systems is that in order to optimize the total situation multiple rules are necessary. These rules should be maintained for the changing environment, where dynamic changes of external stakeholders can be challenging to incorporate. Incorporating multiple objectives require even more rules, which all can conflict. Tradeoffs between objectives or their dependencies causes complex and unclear configurations.

Imagine a global merchant needs different rules in each country for different types of transactions. An example of a rule is to route domestic transactions to domestic acquirers[6, 3], assuming the profits will be better. However, it might be that high amount, recurring, domestic transactions perform better on a certain time of the day to a specific non-domestic acquirer, now who configured the routing rules? Total optimization of payments is in a changing environment is difficult to achieve - not to mention maintainability - using static routing rules.

2-3 Classification in Payments

Classification of fraud has been studied a lot, which has its application in processing payments but also in for example insurances. Research to credit card fraud is focused on identifying fraudulent transactions or users. A transaction is fraudulent when it will result in a chargeback. A chargeback is when a shopper reports the bank that the transaction was not executed by himself and the money is returned if the request is legit. Note that fraud might not be the only application of classification in payments, however it is closely related to refund classification.

The process of predicting refunds is kind of similar, using the same methodologies and work with the same (or similar) bucket of data. In this thesis, refund patterns are explored by predicting if a transaction will result in a refund. Techniques and insights from fraud prediction could also be useful for predicting refunds.

The nature of payment data and operational issues present some challenges in this area[7]. The volume of transactions is really high.[2] All transactions should be performed within the order of hundreds of milliseconds in order to satisfy shopper demands.

2-3-1 Fraud Research

Research in fraud involves private datasets sizing from a few hundreds till a million samples. They vary in attributes from 4 till 60 attributes, using binary, numerical and categorical formats[8]. These specifications are specific to card fraud, but due to the same bucket of data, similar to refund prediction. Two important aspects imbalance and costs are applicable to both fields and described in the corresponding paragraphs.

Imbalance

The occurrence of fraud (sometimes lower than 1%) is relatively rare and causes a highly imbalanced dataset[7], which can be compared to refund ratios. Refund ratio's of 5-30% are

[2]The Industrial Partner processed in 2014 over 25 billion of transactions.

normal for retail merchants while in other branches this can be different. Class imbalancy is studied in more detail in Section 2-5.

Low fraud ratios make the occurrence of *false positives* more likely. False positives are instances classified as fraud that in fact turn out to be legitimate transactions. The other way around, if transactions are classified as legitimate but turn out to be fraud, the predictor missed fraudulent transactions, this is called a *false negative*. These different error types involve different costs. Literature in fraud focuses on measuring the performance by associating the errors with real costs functions.

Costs

The relation between the costs of false positives (false alarm) and false negatives (missed) is comparable between fraud and refunds, however not in the same order of magnitude. The costs of a false positive in fraud detection is the potential loss of profit[9] while in refund routing it is the difference between the routes (recall Fig. 1-5).

The costs of the false negative in both cases are relatively high. In case of fraud this means fraud occurred and is not detected, where this process involves a lot of administrative costs. Same story for refunds where a transaction is routed to a non-refundable route, which involves administrative costs to refund manually. As stated by Krivko et al. the amount of manual interventions should be kept within a certain range to prevent more administrative work than the employees can handle[7].

The industry values predictions that maximize cost savings or profits. A survey by Phua et al. [8] in fraud detection shows that most of the fraud detection studies define explicit costs to be incorporated in the predictions. Some (recent) studies evaluated the techniques using cross-entropy, mean squared error or Area Under ROC (AUC). No evaluation method is widely used in fraud detection apart from incorporating costs. An existing challenge in fraud research are the misclassification costs, these are unequal, uncertain and can differ per instance and can change over time[8]. The application of evaluation methods is studied in detail in Chapter 4.

2-3-2 Supervised Learning

Fraud detection can be done using different approaches. One of these approaches is by supervised learning. In this approach new instances are labeled, based on a model trained on historical data. Transaction histories including the labels for refund prediction are available from historical transactions.

A challenge for supervised learning in data streams is that you cannot be sure about the label of a transaction in a certain timespan[9]. A transaction can be refunded in a certain time period after the transaction have been processed. Transactions in this timespan can be labeled wrongly and therefore be noisy[7]. This is called *online learning* and is not the main focus of this research. Semi-supervised learning is an option as this combines both supervised and unsupervised models.

In this research only supervised models are tested. The assumption is that refund patterns do not change that often therefore the model can be trained once a month, when all labels are definite. Details about the process of refunds can be found in Appendix A-4.

With a survey on fraud detection, Phua et al. show that the emphasis in fraud research is too much on complex systems like Support Vector Machine and Neural Networks and show that simpler models will perform equally or better in the future[8]. In this thesis we will focus on simple models like Random Forest (RF).

2-3-3 Unsupervised Learning

Another approach which is used less often, is unsupervised learning. This approach tries to find abnormal patterns in account behavior. Individual profiles are built which contain characteristics of transaction activities, like the time of day. However the problem in this approach is that abnormal behavior can identify patterns which can correlate with the negative or positive class, or both, and therefore be manually reviewed[7]. In other words, change in behavior may not be specifically due to refund behavior.

Even hybrid models are created where supervised and unsupervised learning techniques are combined to increase performance.

2-4 Historical Transactions

Classification in payments could be done on two different levels; transaction level and account level. Transaction level involves the usage of transaction attributes (like amount and shopper country) and directly linked information (like risk scores). Account level involves inspecting a succession of transactions done by one singe account.

Historical transactions of one account contain information about the behavior of this user, and possibly patterns. The attributes of *all*, or most recent, historical transactions could be passed to a classifier as features. A smart classifier could find patterns between current and historical transactions. However, it is impractical to pass a series of transactions to a supervised learning algorithm due to high dimensionality and heterogeneity of the transactions[10].

Transaction aggregation Whitrow et al. propose a framework to aggregate information over a period of time[10]. Transaction aggregation is a method to generate features by aggregating transactional attributes over a succession of transactions.

Patterns about the account can be detected such as combinations of high average amount of transactions or the frequency of different type of transactions. However, some information is lost due to aggregation, for example the order of the transactions.

It is shown, and advised to research[10], that the length of the aggregation period impacts the performance. A fraud study by Jha et al. show that an high average spending behavior over the last month has a high change on fraud, while an high average spending behavior over the last three months has a low chance on fraud[11]. Transaction aggregation is capable of capturing different behaviors over different periods.

There are some problems with transaction aggregation. A specific amount of historical data is necessary to aggregate and inspect this information. This aggregated data should be updated continuously to catch possible changes in user behavior.

2-5 Imbalanced Data

Most learning systems assume that the distribution of classes in training data are balanced, while in reality this is almost never the case. This section describes the problem of imbalanced data and a subset of existing solutions from literature are explained. This section focuses on the pre-processing solutions, the solutions for evaluation are studied in Chapter 4.

In practice for retail merchants refunds are imbalanced with a ratio of around 5-25%. Other fields of classification deal with even more imbalanced datasets where the ratio is (below) 1%, like credit card fraud. A dataset is imbalanced when one class occurrence is far below the other class occurrence. In general, datasets are balanced when when the ratio between occurrences is around 40-60, however this is very dependent on the context (data and classifier).

Class imbalance could cause a suboptimal classification performance[12]. The most commonly used solution to the imbalance problem is to balance the training set[8, 9, 10, 13]. There are two approaches that achieve a balanced training set;

1. **Under-sampling** All samples of the minority class are combined with an equal amount of samples, randomly sampled from the majority group. Potential risk is possibly missing important samples.
2. **Over-sampling** All samples of the majority-class are combined with an equal amount of samples, randomly sampled from the minority group. Potential risk is that the learner can over-fit due to duplicate samples.

A third approach is based on the over-sampling technique;

3. **Smote** New samples are generated by interpolating between minority samples that are closely together, in terms of feature space. This method reduced the over-fit risk by spreading the decision boundaries for the minority class into the majority class space.

Batista et al. tested various methods to balance datasets, from under- and over-sampling to complex ensembles of these. In general, over-sampling methods performed better and faster than under-sampling methods. Smote + Tomek and Smote + ENN are proposed by Batista et al. and perform best for tested data with small number of positive classes (2-4%). For more information on these methods, we refer the reader to the original paper[14]. Random over-sampling and Smote perform best for datasets with a similar imbalance as refunds (5-25%).

Other important methods are one-class classification and feature selection. The first one is useful under certain conditions, especially for extremely skewed data with highly dimensional and noisy feature space[12]. Chawla et al. also show that it is important to select the features that lead to high separability between the imbalanced classes. Feature selection methods could be executed on both classes, and then combined properly.

It is well worth to note that a lot of classification research is done assuming that the class imbalance of the training set is somehow similar to the true distribution (ie when used in practice). However, this might not be the case, especially not in payments. Landgrebe et al. show that changing class imbalance from a balanced training situation to an imbalanced situation, could drop the performance for a linear classifier[15]. The refund rate might change from time to time, resulting in a different class ratio and thus classifier performance.

2-6 Categorical Data Encoding

In practice only a selection of classification implementations support input formatted as categorical data. Some of them have restricted number of supported categories due to computational limitations. Therefore we list existing methods to encode categorical data into numerical data.

In order to explain the different encoding methods we take one example feature which we encode with the described method. The sample feature x_1 contains 4 samples of 3 categories (small, medium and big);

$$x_1 = [small, medium, small, big] \tag{2-1}$$

Ordinal This method assigns numbers from 1 to N for every category, where the number of categories is N. The example would be encoded as follows;

$$x_1' = [1, 2, 1, 3] \tag{2-2}$$

To make this format useful, categories should be sortable.

1-out-of-N (or N encoding) 1-out-of-N will add N features for N categories in the original feature. Every generated feature represents one category and is coded either 1 if the category corresponds to this category, or 0 otherwise. The transformed sample looks as follows;

$$x_1' = [1, 0, 1, 0] \tag{2-3}$$
$$x_2' = [0, 1, 0, 0] \tag{2-4}$$
$$x_3' = [0, 0, 0, 1] \tag{2-5}$$

An advantage of this method is that the categories are still interpretable because each category is represented by its own feature. This method is useful for nominal input values. However, the number of features grows with the numbers of categories, this significantly increases the training time.

1-out-of-N - 1 (or N - 1 encoding) This is a variant on N encoding, where the last category is represented by zero's in all other categories. The example feature is transformed as follows;

$$x_1' = [1, 0, 1, 0] \tag{2-6}$$
$$x_2' = [0, 1, 0, 0] \tag{2-7}$$

This method would have one less feature in comparison with N encoding, however still not scalable in terms of categories.

Temperature Temperature encoding adds N features for N given categories, just like N encoding. However a feature and all of its preceding features will be coded with a 1 for a matching category, and all succeeding with a 0. The feature is transformed as follows;

$$x'_1 = [1, 1, 1, 1] \tag{2-8}$$
$$x'_2 = [0, 1, 0, 1] \tag{2-9}$$
$$x'_3 = [0, 0, 0, 1] \tag{2-10}$$

Not every generated feature represents a category. Every feature now represents 'at most this value'. So for the given example, medium is encoded as 'at most medium', so small also fits. This method can only be useful for categories with a meaningful order. Just as the N encoding, the features of Temperature encoding also scales with the number of categories.

Temperature - 1 Temperature -1 is a variant on Temperature, as N - 1 encoding is for N encoding. The last category is represented by the other features, however, now the first feature is removed. The example is transformed as follows;

$$x'_1 = [0, 1, 0, 1] \tag{2-11}$$
$$x'_2 = [0, 0, 0, 1] \tag{2-12}$$

This method would have one less feature in comparison with Temperature encoding, however still not scalable in terms of categories and of no use for nominal input data.

Nishisato scaling [16, 17] Each feature is replaced by a new feature, which is formatted as continuous numerical value. Each category is substituted by the probability of the positive class within this category. Category n (in N number of categories) of feature x_i is quantified as;

$$x_{in} = \frac{p_{in}^{pos}}{p_{in}^{pos} + p_{in}^{neg}} \tag{2-13}$$

where:

$$x_{in} \text{ is the scale value for category } n \text{ in feature } i \tag{2-14}$$
$$p_{in}^{class} \text{ is the probability on } class \text{ for category } n \text{ in feature } i. \tag{2-15}$$
$$class \text{ can be either positive } (pos) \text{ or negative } (neg) \text{ for binary classification.} \tag{2-16}$$

In reality these class probabilities are unknown and therefore estimated by observing the known class distribution. The transformed example for output $y = [pos, neg, neg, pos]$ and feature x_1 becomes $x'_1 = [0.5, 0.0, 0.5, 1.0]$. It is shown that the classes are replaced by the corresponding *positive class fractions*.

Fahrmeir scaling [18] The approach of Farhmeir scaling is similar to the method designed by Nishisato, creating one vector containing continuous numerical values. The scale value of each category is quantified as follows;

$$x_{in} = \begin{cases} \frac{p_{in}^{pos}}{p_{in}^{neg}} - 1 & \text{if } p_{in}^{pos} \geq p_{in}^{neg} \\ 1 - \frac{p_{in}^{neg}}{p_{in}^{pos}} & \text{otherwise} \end{cases} \tag{2-17}$$

There is a problem with this scale value, if either p_{in}^{pos} or p_{in}^{neg} is equal to zero, the value is mathematically ∞ or undefined. To solve this issue we program the probability *very* low (1E-5)[3] when zero. Transforming feature x_1 for the same output y as in Nishisato, it becomes $x_1' = [0, 0.99999, 0, 99.999]$. These values express scaled versions of the positive and negative class, sensitive to the difference between them. To clarify this scale values, scales are calculated for example probabilities p_{in}^{pos}, and shown in **??**.

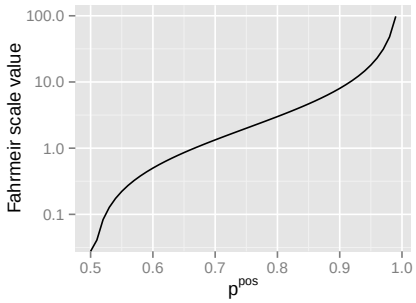


Figure 2-1: Example probabilities and the corresponding Fahrmeir encoding for p_{in}^{pos} from 0.5 to 0.99.

Overview

Categorical data can either be on a nominal or an ordinal scale. If an interval (or even ratio) values can be extracted from the categories, this scale can be used instead of encoded methods. The compatibility of the listed methods for the scale origin of categories is shown in Table 2-1. Depending on the nature of the category scaling appropriate encoding methods can be used.

The Ordinal, 1-out-of-N, 1-out-of-N - 1 and Temperature - 1 are tested in a recent study. Fitkov et al. use a Neural Network to show that Ordinal and Temperature encoding both perform better than both versions of 1-out-of-N[19]. Note that the categories in the used dataset had a meaningful order.

Kauderer et al. tested the 1-out-of-N, Nishisato and Fahrmeir encoding with four different classifiers; C4.5, LDA, KNN, DIPOL and Neural Network[20]. Nishisato performs slighly better than 1-out-of-N and Fahrmeir for all classifiers. Nishisato also performs twice as fast as 1-out-of-N during training, due to the increased number of attributes for the latter method.

[3]The maximum number of samples used is 44k, where the lowest fraction could be 1/44k which is 2.27E-5, still more than our artificial zero value

	Encoding Method				
Scale	Ordinal	1-out-of-N (- 1)	Temperature (- 1)	Nishisato	Fahrmeir
Nominal	-	+	-	+	+
Ordinal	+	+	+	+/-	+/-

Table 2-1: Usefulness of using encoding methods for different types of categorical features.

2-7 Conclusion

In this chapter we summarized related work, and showed what is understudied in literature. The described subjects require different additional research. These challenges and opportunities are elaborated in the next chapter, Chapter 3. In short the most important conclusions of the described fields;

Apart from public documents from payment processors, Payment Routing is not elaborated in detail in literature. Routing challenges are interconnected to multiple stakeholders, their differences, and therefore are bigger and complexer than just routing. The addressed problems are routing between acquirers, not payment schemes, which opens up some opportunities.

Research to fraud detection shows us the influence of class imbalance and costs for classification evaluation. Different techniques are studied to deal with a class imbalance. The influence of costs needs some additional analysis and is incorporated into Chapters 4 and 5.

Transactional data is mostly formatted in categorical attributes. Different encoding methods are defined to transform categorical data into numerical data. These methods are not new, but still effective. Additional experiments should indicate which method(s) works best for refund prediction.

Transaction aggregation is shown to be effective in incorporating historical transactions. Historical attributes can be generated over different time spans, which could have a different effect on the classification performance. This requires some additional experiments, if incorporated into refund classification.

Chapter 3

Research Approach

3-1 Introduction

As introduced in this thesis, the main goal is to optimize payment routing by predicting refund behavior of individual transactions. During the design of this system a lot of aspects can play a role. In this section we define which main challenges and questions are elaborated. After reading this chapter it will be clear on which aspects and challenges we focus during this thesis.

First the key aspects of the main problem are described in the problem definition. After the problem is explicitly stated, an approach is proposed on how to solve this problem. The research approach will be followed by a section devoted to explaining the expected motivation and contribution of the work. Given the problem and main focus detailed research questions are formed to shape the contents of the research. At the end the structure of the research will be explained.

3-2 Problem Definition

Before moving to the contents of the research we will explain the problem very short and concise. A sketch of the situation together with the goal introduce the main problem.

A complex network Dual branding and multiple acquirer contracts enables different routes for transactions via respectively the scheme and acquirer. A gap exists in literature, ie missing scheme routing for payment providers by leveraging dual branding. This thesis will focus on *payment scheme routing* through dual branding.

Deciding on a route between all available stakeholders is a challenging task, especially for merchants which miss domain knowledge in the payment world. The presence of multiple objectives, and their tradeoffs, makes this decision even harder. Complexity is not in the amount of possible routes, but in the objectives and their underlying inter dependencies. To reduce this complexity, only costs will be used as an objective, in particular the *scheme fees*.

Optimization The ultimate goal is to select the best route, or in other words, subset of stakeholders. Choosing a route using rule-sets cannot benefit the differences in feature support between some alternative routes. *Individual transactional attributes* influence route decisions. In order to optimize, routes should be chosen at individual transactional level.

Dynamic Transaction Routing Routing dynamically will decide on a route for every transaction. The term dynamic is involved because because every transaction is treated differently, there is no merchant wide rule-based configuration sending the transactions over one specific route. An open challenge exists in *dynamic transaction routing* between different payment schemes.

Support for Refunds The differences in feature support between payment schemes and the usage preference of merchants stops (or limits) a merchant from dynamic routing. Payment features are recurring payments, payouts, chargebacks and refunds. Refunds are a key asset, and therefore necessary, to a retail merchant. Therefore we focus on transaction routing with *support for refunds*. Refund behavior should be extracted from a transaction using supervised classification techniques.

The former aspects together form the main challenge of this thesis. This challenge can be expressed in the following research question;

Main RQ: How can we use classification methods to predict refund behavior per transaction in order to optimize payment routing?

This question identifies (1) if and how refund prediction using classifiers on payment data is possible, and if so, (2) how we can incorporate this into routing decisions.

The limitations due to stakeholder differences in optimizing payment routes are leveraged. During this thesis we will explore possibilities to deal with these limitations while optimizing the total situation.

3-3 Research approach

In the previous section we have defined a specific research question that is the focus of this thesis. However more than one route leads to Rome, ie multiple strategies to solve this problem. In this section we shortly describe the solution strategy for the problem just introduced.

Given is that a transaction enters a system, where the most profitable route should be selected. There are two important components in this route decision, *a*) the transaction costs per route, per non- and refund class, and *b*) the refund prediction. Both components need to be combined to make a proper route decision.

Two high-level strategies incorporate transaction costs per route and prediction knowledge for a route decision. The strategies are defined as follows;

1. **Prediction of routes** A classifier will be designed that can predict the preferred route. Transaction costs are calculated for both routes and together with refund support added to the featureset. The classifier can be trained to recognize optimal routes.

2. **Prediction of refunds** A classifier will be designed that can predict if a transaction will be refunded. With the knowledge of the prediction, simple logic can decide on the route.

Both strategies are related and solve the main problem, to optimize the route decisions. The first strategy is likely to overfit on transaction amount, due to the direct dependency of transaction costs and route decisions. The second strategy predicts the key information for the decision. The other information for the decision, the route costs, can be determined and this way the following decision logic does not make mistakes.

In this thesis the second strategy is elaborated because the main responsibilities are separated. The predictor focuses on predicting the underlying information while the decision logic decides on the optimal route with the help of the prediction and costs. With the predictor separated, it can be user for different applications. This modular approach enables future research to incorporate more information, from either classifiers of different sources, to increase the decision performance.

3-4 Motivation and Contribution

A drive for innovation in the payment industry created space to ensemble the subject of this thesis. During the introduction we identified the practical problem of inability to decide and optimize between routes due to unequal support for a feature (ie. supporting refunds). The proposed solution is to predict refunds, in order to be able to decide on a route and optimize the network. This research is interesting for both the industry as science.

3-4-1 Industry

Reduce transaction costs Being able to refund is a key asset of retail merchants. These merchants send their traffic to refund supporting routes even if these are way more expensive. With refund classification we create an opportunity for retail merchants to drastically *decrease transaction costs*.

The approach used in this thesis is not specific to retail merchants and therefore also applicable to other *branches*, under the assumption[1] that refund patterns can be recognized there as well.

Next to the merchants, the industrial partner also has benefits from this optimization approach. By offering the merchants cost savings on their transaction costs, it has a *competitive advantage* towards other Payment Service Provider (PSP)s.

[1] See it as hypothesis for future research

Applicability of refund knowledge To the best of our knowledge, refund prediction has not yet been studied in public literature. The identified patterns and methodologies to predict refunds can be used for other purposes.

Imagine the added value of a retail merchant who is able to see the probability that this order is going to be returned. The merchant can *increase the service* by giving some additional advise about the goods in the order, alternative goods, or the refund process. In addition to that, if the algorithm is confident enough about a high probability, the merchant could even decide to cancel the transaction, to save time and effort of delivery and return. Both ideas are possible future applications, which could be built with well performing refund classification.

Classify-optimize-route Refund is not the only feature which opens up possibilities to compare schemes. Other scheme features like *payout or recurring* also lend themselves for this classify-optimize-route approach.

Next to schemes also other stakeholders in the payment process can be used for routing. For example *acquirers* with different features, properties and costs. Identify a key difference, and this could be elaborated with classify-optimize-route approach.

3-4-2 Science

Refund patterns Current payment literature does not cover refund behavior. The closest studies are limited to understanding e-commerce buying behavior. Refund patterns found in this research can contribute to payment literature in two aspects; first it explicitly tries to understand refund patterns, and second future research is able to connect this with e-commerce behavior.

Payments Contribution to payment research is difficult due to the unavailability of public data. In practice still too many research, especially fraud research, is done using old, small or even artificially generated data. Recommendations refer to the fact that more research with payment data is necessary.

Categorical attributes are quite common for transactional data. In general there are not so many generic attributes[2] and all with a fixed number of possibilities. Research to transforming these attributes into numerical attributes might simplify future research in this field.

Transaction Aggregation Feature generation by aggregating historical transactions is introduced in fraud research in 2008. Almost all fraud studies after this paper incorporate transaction aggregation. In harmony with the mentioned understudied payment data, research to transaction aggregation improves the understanding, strong and weak points of the technique. Applying transaction aggregation for refund prediction gives other perspectives to the idea.

[2]In practice a lot of fields are scheme specific.

3-5 Research Questions

This section describes in detail the aspects and questions to solve the main problem described in Section 3-2. The research questions are divided into three groups; *1*) transaction routing *2*) prediction of refunds, and *3*) evaluation of the system.

3-5-1 Transaction Routing

In practice only acquirer routing is applied, during this research we focus on scheme routing. Transactions are routed with dynamic routing decisions. A refund prediction of a transaction can determine the least-cost route for a transaction. In this thesis optimization of payment routes has one objective, cost minimization. While this 'optimization' logic can be rather simple, the intervention of a classifier in the optimization might change the complexity. We would like to see how the classifier influences the optimization question.

RQ1: How does the intervention of a classifier influence optimization of transaction routing?

The different consequences for misclassification ask for a different evaluation strategy to qualify the performance of the classifiers in terms of routing possibilities. As described before, due to different support of payment features, the accuracy of the predicted feature defines routing is possible. This implies that if the accuracy of the prediction cannot be guaranteed, we cannot route. However, selecting the wrong route comes with a price, just like false positives do. We would like to know what the ideal accuracy is, and thus how high should it be to create a favorable situation to use it.

RQ2: What is the minimal classifier performance to enable a profitable situation for transaction routing?

3-5-2 Prediction of Refunds

Refund prediction is to the best of our knowledge not yet explored in existing literature. Before diving into the details, this exploratory research raises a high level question about the ability to classify refunds.

RQ3: Is refund prediction possible with transactional data, and how does it perform in general?

Data Retrieval and Transformation This research is elaborated in cooperation with an industrial partner, and thus we have access to different large and rich payment datasets. These datasets contain information about payments for different merchants located all over the world. Domestic and cross-border payments are processed and logged for years. Per transaction, information is registered about the type of payment, user, used device, delivery,

but also risk scores *etcetera*. This information opens an opportunity to use lots of data fields for classification.

The time that the outcome of the predictor is necessary restricts the use of the available data. On a certain moment in the process a decision is made to route the transaction. All the data that is generated or collected afterward, cannot be used for prediction in routing context, which limits the available data.

RQ4: What information can be retrieved at the time of the refund prediction?

The retrieved information might not be sufficient to predict refunds. Additional data from the merchant might help the prediction. For example the behavior of a user on the merchants (web)shop can correlate with refund behavior. Here behavior can be seen as the time spent deciding to add it to the cart or multiple items in different sizes in one order. Some insights could be gathered about information, to be collected in the future, which helps predicting refund behavior.

RQ5: Which information should have been available for refund prediction?

Features In classification, features are the pieces of information in which patterns can be found. A transaction attribute can be used as feature, like the transaction amount. Structured analysis is necessary to define and generate features. The combination and quality of the features define the performance of the classifier.

The available attributes are not always directly usable as a feature in a classifier. Some attributes are formatted as categories and not every toolbox or classifier can deal with categorical formats. Therefore the right method of transformation should be found for these data attributes.

RQ6: Which encoding methods work best to transform categorical attributes into numerical features?

The first step in exploring features is to apply the payment specific features used in fraud detection. They might not all be equally useful, but they will give insight in the usefulness for refunds. Next to fraud research other domains can be used as well.

The second step is to explore other domains. Some features from other domains can be applicable on the payment domain as well. For example in recent research on Twitter they use strategies where features are derived from the user or topic. Feature generation strategies from other domains can be adapted and used in the payment domain.

RQ7: Which features from fraud detection also apply to refund prediction?

Classification algorithms With the given features, a classification algorithm searches for patterns in the given information. Imagine that a very simple classifier draws a line (like a threshold) through all features, where each side represents one class. Different classification algorithms draw this line differently. Finding the best line that separates the classes, is a way to find patterns which recognize one class.

Multiple classifiers will be tested with the goal to identify and use the one that finds the patterns the best, and thus to retrieve the best possible predictions. The most common classifiers from literature, which are applicable to this context, will be tested for refund prediction and compared.

RQ8: Which classification algorithms perform best for refund classification in context of routing decisions?

When the best performing classifier is selected, we would like to know how we could improve the classifier specifically for this routing situation.

RQ9: How can we improve the performance of the classifier in context of routing decisions?

The performance of the classifier measures the correct predicted transactions. In context of routing decisions, the evaluation of a classifier can become more difficult. We will discuss the evaluation methods in Section 3-5-3.

3-5-3 Evaluation

The performance of the classifiers and different combinations of feature sets can be evaluated using different measurements. Basic measurements, like the classification rate, precision and recall, are focused on the relation between correctly and incorrectly classified instances. Different types of misclassifications impact the overall performance differently. Analysis is necessary, to evaluation methods which are capable of measuring performance of the algorithm in relation to this specific context.

RQ10: Which existing evaluation methods measure performance considering the refund specific characteristics?

Next to basic measurements, a loss function can be used to explicitly give context to misclassifications. Misclassified instances come with a price: the manual work of the refund. These instances are payments routed to cheap networks and which turn out to 'use' the refund feature. For some payment methods this could mean that they should call the shopper for its bank account and transfer the money themselves. The winnings of the cheap route should surpass the loss of these misclassifications.

A loss function combines erroneous classifications with a realistic loss function [10]. The loss function brings the theoretical approach into practice by incorporating the consequences to the evaluation of the classifier. A realistic loss function would use knowledge on the costs of different misclassifications in relation to the routing objectives.

RQ11: How would we ideally measure the classifier performance which expresses the winnings in relation to payment routing?

3-6 Conclusion

The main goal of the research is to enable and optimize transaction routing between schemes. The difference in support for refunds is the biggest challenge. A predictor plays an essential role by predicting refunds for individual transactions, which is used for the route decision. The use of a refund classifier is much wider than just for routing purposes. With a proper refund classifier transaction routing can be optimized with the goal of minimizing costs.

The research part of this thesis follows the subjects of the research questions. The order of the subjects however is changed a little. A proper evaluation method is established by a study into to existing evaluation methods and the introduction of an advanced evaluation method. Optimization of transaction routing is studied during the construction of the evaluation method. This evaluation method is used in further research.

The predictor is designed in the followed chapters by first defining the used information, and second the used classifier. Given the new evaluation method, the predictor's performance is optimized for transaction routing. Several experiments show the usability of the evaluation method as well as the increase of performance. With the knowledge gained from this experiments a final model will be constructed and validated.

Chapter 4

Evaluation Methods

4-1 Introduction

Designing 'the optimal classifier' requires defining a proper instrument to measure. The term performance can be defined in different ways, but essentially it measures how good the classifier is performing in a certain context. Different contexts require different measurement instruments. In this thesis we focus on transaction routing. Specifically to reduce transaction costs by applying transaction routing.

The following four criteria are derived from characteristics of the problem. These should be taken into account in selecting a proper evaluation method:

1. **Robust against small minority class (class imbalance)** The refund ratio for retail merchants is roughly around 5-30%. A class imbalance makes it hard for the classifier to catch the minority group. In routing context, the classifier should predict both classes correctly, and not just the majority or minority. The evaluation method should take into account prediction performance of both classes and therefore be sensitive to class imbalance.

2. **Should supply a decision threshold** The outcome of a classifier generates 'posterior probabilities' that represents the probability on the class. These probabilities require a cutoff, also called decision threshold, to decide on exactly one class. In this thesis we are designing an operable classifier, which requires an exact threshold. The threshold should not be given to the evaluation method, instead it should incorporate, optimize or decide on a threshold itself.

3. **Should handle misclassification costs** Available routes come with different costs and refund support. A transaction route will be decided based on the prediction and some additional logic. Next to that a route can infer additional costs for refunds (manual refunds). The evaluation method should be able to incorporate these (misclassification)costs.

4. **Should handle individual transaction costs** Transaction fee or costs vary for every transaction and is dependent on the transaction amount (or value). This gives an extra dimension to the misclassification costs. The evaluation method should be sensitive to the (misclassification)costs of every individual transaction.

In the rest of this chapter we will test existing evaluation methods against the defined criteria. The methods will be summarized, and their advantages and disadvantageous will show the usefulness in the routing context.

4-2 Overall Success Rate

Simple evaluation of predictors start by counting the number of errors. Therefore, it is applied to unforeseen data. The predictions will be compared against the truth of the instances. A confusion matrix (or error matrix) is designed to show the performance of a machine learning algorithm in a structured table [21]. Confusion matrix is derived from the contingency table, where in the case of a confusion matrix, the frequency distributions of different types of successes and errors are displayed in a structured way.

Instead of a refund predictor we create a non-refund predictor by changing the positive class to non-refunds. This decision follows from the fact that non-refund transactions will be rerouted to the cheaper scheme. In Section 4-5 we will explain that we want to predict as much non-refunds as possible.

		Truth	
		Non Refund	Refund
Prediction	Non Refund	n_{TP} (True Positives)	n_{FP} (False Positives)
	Refund	n_{FN} (False Negatives)	n_{TN} (True Negatives)

Table 4-1: Structure of a Confusion Matrix

An example of a confusion matrix, using the refund classes, is shown in Table 4-1. The columns represent the truth while the rows represent the prediction. The table entries contain the number of instances which belong to that particular group, where Table 4-1 contains the names of how we call these entries.

The numbers are defined as n_{TP}, n_{FP}, n_{FN}. We denote the truth groups by n_{NR}, n_R and the predicted groups by $n_{NR'}$, $n_{R'}$.The principal diagonal contains the correctly classified instances, while the other entries are misclassified.

The terms *True* or *false* tells if this prediction was correct or not. *Positive* and *Negative* refer to the predicted class. In binary classification you have only two classes, where the one you are looking for is called a positive, and the other a negative.

There are two types of errors in binary classification:

1. *False Positive* also known as false alarm or type I error, and
2. *False Negative* also known as missed one or type II error.

Basic performance measures can be calculated from a confusion matrix. These measurements show the relation between different kind of errors and successes of the prediction. The simplest example of such a basic measurement is Overall Success Rate (OSR).

Definition 4.1. *Overall Success Rate is defined as the ratio of correctly classified instances (refunded and non-refunded) in the total set, associated with a specific score, s.*

$$OSR(s) = \frac{n_{TP}(s) + n_{TN}(s)}{n_{TP}(s) + n_{FP}(s) + n_{TN}(s) + n_{FN}(s)} \tag{4-1}$$

A decision threshold is required to calculate the OSR. Literature refers to this decision threshold as a 'specific score'. Setting the decision threshold to 0.7 will classify all instances with a probability higher then this threshold as positive class. The numbers of entries in the confusion table will differ when choosing a different score. A score is required to calculate the OSR, nevertheless there is a possibility to take the maximum over scores of all possible thresholds. This evaluation method does not incorporate any (misclassification)costs.

A simple example shows its weakness to class imbalance. Imagine a classifier, $CLASS_{NR}$, that only 'predicts' non-refunds. The OSR of classifier $CLASS_{NR}$ will be 95% on a merchant with 5% refund rate. However, the classifier missed 100% of the refund transactions. With this class imbalance, the evaluation method does not properly evaluate performance of both classes.

Criteria	Support
Class imbalance	No
Decision threshold	Yes
Misclassification costs	No
Individual transaction costs	No

Table 4-2: Matching evaluation criteria for OSR.

4-3 Marginal Rates

The following measures are called marginal rates in literature [21]. These are more specific than OSR and show the ratios in a group related to one error type. This will give more insight in the predictive power of a specific class. The measures are calculated over the predicted classes or truth groups, resp. the rows and columns of a confusion matrix. We will only explain the measures related to the non-refund prediction power.

Definition 4.2. *True Positive rate (TPr), is defined as the ratio of correctly classified non-refund instances out of the true non-refund set, associated with a specific score, s.*

$$TPr(s) = \frac{n_{TP}(s)}{n_{TP}(s) + n_{FN}(s)} \tag{4-2}$$

Definition 4.3. *False Positive rate (FPr) is defined as the ratio of wrongly classified non-refund instances out of the true refund set, associated with a specific score, s..*

$$FPr(s) = \frac{n_{FP}(s)}{n_{FP}(s) + n_{TN}(s)} \tag{4-3}$$

Note that there are more marginal rates described in literature, some of these are explained later this chapter.

Compared to OSR, marginal rates can distinguish the predictive power of the two classes, and the different types of errors. For example the FPr explain the ratio of negative classes identified as positive.

A single marginal rate is not able to describe the performance of a classifier. Take for example the classifier $CLASS_{NR}$, that only 'predicts' non-refunds. This classifier identifies all non-refunds and therefore the TPr is 1, while at the same time the FPr, error of misidentifying the refund class, is 1 as well. Due to tradeoffs it is not possible to maximize or minimize marginal rates, at least two rates should be balanced to maximize performance.

Criteria	Support
Class imbalance	No
Decision threshold	Yes
Misclassification costs	No
Individual transaction costs	No

Table 4-3: Matching evaluation criteria for marginal rates.

4-4 Area Under ROC

A prediction of a classifier for one transaction consists of two probabilities; to be refunded or not. These two chances are complementary. The probability density function of a class shows how many instances can be predicted correctly for a specific threshold on the given instance probabilities. These probabilities are also called scores. Deciding on a score explicitly decides between a refund and a non-refund. In practice probability density functions of two classes overlap. An example of how these functions can overlap is shown in Fig. 4-1.

The probability of belonging to a class given a specific decision threshold T is written as $P_{<class>}(T)$. Then the FPr and TPr, associated with a specific score, s, can be written as:

$$TPr(s) = \int_s^\infty P_{NR}(T)\,\mathrm{d}T. \tag{4-4}$$

$$FPr(s) = \int_s^\infty P_R(T)\,\mathrm{d}T. \tag{4-5}$$

The overlap of the class probability density functions can be minimized with the help of a Receiver Operating Characteristic (ROC) curve. Choosing a different threshold causes changes in the confusion matrix, and thus the marginal rates. The ROC curve shows the

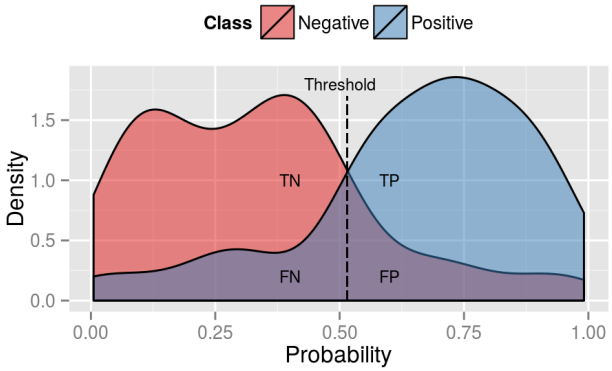


Figure 4-1: Overlap of probability density functions

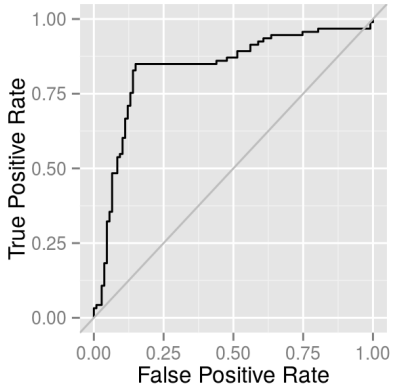


Figure 4-2: Example of ROC curve

relation between TPr and FPr for all available probability thresholds. In other words, it shows the tradeoff between hit rate and false alarms. An example of an ROC curve is shown in Fig. 4-2.

A ROC curve is a proper method to visualize and visually compare the performance of different algorithms. One can quickly see how many false alarms are necessary to achieve a certain success rate. Comparing two algorithms with their ROC curve is possible, but limited. A ROC curve of which all points are higher performs better. Next to that, the difference in steepness, in the left or right side shows different characteristics. However, once algorithms intersect, it is difficult to state that one is better than the other. Ordering of multiple algorithms is not possible based on a ROC curve due to missing metric on ratio scale. A metric on interval scale based on the ROC curve is the AUC.

The term Area Under ROC (AUC) explains itself. It ranges between 0 and 1, where higher scores indicate an *average* better classifier performance. AUC replaced OSR a long time ago and is a robust measure for comparing classification algorithms [22, 23]. The terminology in Definition 4.4 slighly changed from [24] to match the former definitions.

Definition 4.4. *The Area Under ROC (AUC) is defined as follows:*

$$AUC = \int_{-\infty}^{\infty} TPr(T) \cdot P_R(T)\,\mathrm{d}T. \tag{4-6}$$

The Area Under ROC has some beneficial characteristics; It is insensitive to *a*) changing class imbalance, *b*) misclassification costs, and *c*) does not require a decision for a threshold. The latter point makes it is objective in a sense that is does not require input from the user [22, 24]. In other words, it will optimize the average tradeoff over all thresholds. These characteristics made the AUC one of the most popular evaluation method for classification algorithms.

A drawback is that comparison using AUC can give misleading results if two curves cross each other. The conclusion from the AUC scores than would be that they have equal performance, however they are totally different classifiers. An example is shown in Fig. 4-3.

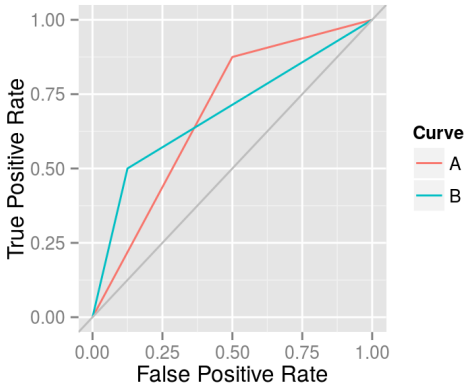


Figure 4-3: Example of two equal AUC scores with equal AUC but different ROC curves, and thus different characteristics.

There are two important reasons why the AUC does not work for transaction routing. First, a classifier selected with AUC, still requires additional analysis to choose the optimal threshold. The classifier is no workable classifier yet, which is not preferable. Deciding on a threshold impacts performance of the classifier and thus the predictions in context. Second, we are looking for the algorithm that performs best in terms of cost reduction. Both evaluation criteria about misclassification costs are not elaborated in AUC.

Criteria	Support
Class imbalance	Yes
Decision threshold	No
Misclassification costs	No
Individual transaction costs	No

Table 4-4: Matching evaluation criteria for Area Under ROC.

Advanced versions of the AUC are the partial Area Under ROC (pAUC) and logarithmic Area Under ROC (logAUC). Both weight another part of the curve differently. pAUC measure only a part of the curve, while logAUC logarithmically weights the curve. It is shown that these measures perform better than AUC if the focus is to identify as much as possible while keeping the error rate as low as possible [25]. However, both pAUC and logAUC do not fulfill more criteria, as defined in the introduction of this section, than the AUC.

In addition to the given drawbacks the marginal rates of the ROC are not the interesting ones in routing context. The ROC plot is based on two marginal rates, the TPr and FPr. In the next section we will describe that Precision and Recall are more important in this context and should be balanced.

4-5 Average Precision

Together, recall and precision both measure the success of the non-refund prediction (i.e. True Positives). Recall measures in the positive truth group while precision measures in the

predicted positive group.

Definition 4.5. *Recall, abbreviated with Rec, is equal to TPr, defined in Definition 4.2*

Note We will switch in terminology from TPr to Recall and back, when necessary. This is because we align to literature as much as possible and the two different terms are used in different concepts.

Maximizing recall can be seen as **maximizing the routing opportunity**. Recall shows the ratio of predicted non-refunds, identified from the true non-refunds. In a routing context, non-refunds are routed to a cheaper scheme. The more non-refunds the classifier can identify, the more the system can route to a cheaper scheme.

Definition 4.6. *Precision is defined as the ratio of correctly classified non-refund instances out of the predicted non-refund set, associated with a specific score, s.*

$$Prec(s) = \frac{n_{TP}(s)}{n_{TP}(s) + n_{FP}(s)} \tag{4-7}$$

Maximizing precision can be seen as **maximizing the successes in all re-routed transactions**. All predicted non-refund transactions will be re-routed to the cheap scheme. The more re-routed transactions really are non-refunds, the less the system has to pay for manual refunds.

These understandings of precision and recall show that the Precision-Recall (PR) curve might be a better focus than the ROC. Davis and Goadrich show that an algorithm that optimizes AUC (of a ROC curve) does not necessarily optimize the area under the PR curve [26]. This supports the decision to focus on the PR curve.

A PR curve is the curve showing the relation between recall and precision for all thresholds. The curve is similar to ROC, described in Section 4-4, but uses different marginal rates. The PR curve is optimal in the upper right corner, while that is the upper left corner for the ROC curve.

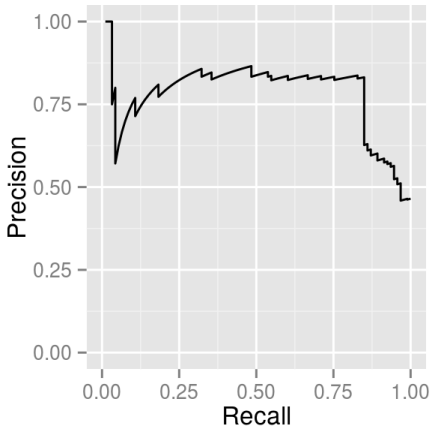


Figure 4-4: Example of Precision-Recall curve

Another difference is that this curve is tooth-shaped, see an example of a PR curve in Fig. 4-4. The reason for this is that when you predict a transaction as non-refund and it turns out to be a refund, recall does not change while precision drops. Now if instead, it turns out to be a non-refund, both recall and precision will increase.

While ROC is insensitive to imbalanced data, the PR curve is sensitive to imbalanced data. Precision is a metric that spans between the two classes. Changing the ratio between the classes influences the precision. Maximizing recall and precision, are both in favor of the positive class.

Due to the distinctive saw-tooth shape, the area under the curve should be estimated. In practice interpolation is used to approximate the area under the PR curve. Interpolated precision for a given recall is defined as the highest precision for any recall higher than the given recall value. However, we define the average precision as metric, it is shown by Boyd et al. that this estimator is one of the most robust ones [27].

Definition 4.7. *The Average Precision (AP) is defined as follows.*

$$AP = \frac{1}{n} \sum_{i=1}^{n} \widehat{Prec}(Y_i) \tag{4-8}$$

where n is the number of non-refunds (positive class), and $Y_1 \dots Y_n$ the posterial probabilities of the non-refunds.

The construction of the AP (due to the curve) is similar to that of AUC. Both are more or less the area under a curve. Concerning the criteria in route context, AP suffers from similar disadvantages as AUC, described in Section 4-4.

Criteria	Support
Class imbalance	Yes
Decision threshold	No
Misclassification costs	No
Individual transaction costs	No

Table 4-5: Matching evaluation criteria for Average Precision.

4-6 F-measure

There is a tradeoff between Recall and Precision. If more transactions are identified as non-refund (increase recall), precision will decrease due to the increase of false positives. The *F-measure* is a weighted combination of precision and recall for a specific threshold[21].

Definition 4.8. *F_β measure is defined as a weighted average of precision and recall, associated with a specific score, s. Recall is weighted β times as much as precision. Common values of β are 0.5, 1 and 2. $F_{0.5}$ puts more weight on precision, while F_2 puts more weight on recall.*

F_1 is the harmonic mean of the two.

$$F_\beta(s) = (1+\beta^2) \cdot \frac{Prec(s) \cdot Rec(s)}{(\beta^2 \cdot Prec(s)) + Rec(s)} \tag{4-9}$$

$$F_1(s) = 2 \cdot \frac{Prec(s) \cdot Rec(s)}{Prec(s) + Rec(s)} \tag{4-10}$$

$$= \frac{2 \cdot n_{TP}(s)}{2 \cdot n_{TP}(s) + n_{FP}(s) + n_{FN}(s)} \tag{4-11}$$

The F_β measure attempts to arrange the precision-recall tradeoff. It can successfully change the weight of one and balance 'Route opportunity' against 'successful rerouted transactions'. Even misclassification costs can be converted into the weights of precision and recall. The optimal β for the F_β measure can be calculated. A class imbalance can be taken into account in the weight of the F-measure. Next to that, to calculate the F_β measure, threshold is required, which can be optimized.

Criteria	Support
Class imbalance	Yes
Decision threshold	Yes
Misclassification costs	Yes
Individual transaction costs	No

Table 4-6: Matching evaluation criteria for F-measure.

4-7 Loss function

The main objective is to design an optimal[1] route decision system. Realistic loss functions combine different erroneous classifications with different cost functions[10]. Selecting the wrong route (caused by a bad prediction) comes with a price, just like false positives do. The route costs can be explained best by showing a decision tree, of transaction routing.

The decision tree is shown in Fig. 4-5. First a prediction decides the route to take, what charges the costs for this specific route. After a route is taken (and some variable delay) we know if it will be refunded or not. A refund can add additional costs, if the route without refund support is chosen. A transaction is overpaid if it was not a refund, but was send to the route with support for refunds. The costs associated with the outcomes are extracted from decision tree and are shown in Table 5-4.

The decision tree in Fig. 4-5 is different compared to fraud prediction. In fraud a decision do refuse a transaction stops the process. After this decision, there is no way to check if this decision was the right one. In the situation for refund prediction, we can always measure the effect of our decision.

These costs associated to different successful and erroneous predictions can be combined in a loss function. The loss is calculated over the prediction result for a specific decision threshold. The optimum loss is the minimum of all losses over all tresholds [28].

[1] Or more specifically, one that minimizes the costs.

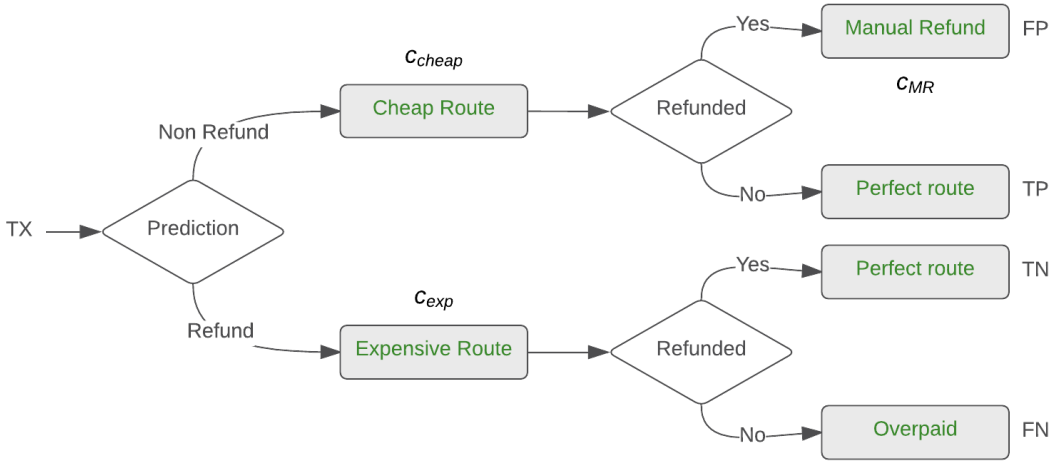


Figure 4-5: Decision tree using the predictor to route transactions. The costs for specific parts in the tree are shown next to the parts in the form c_{xx}.

		Truth	
		Non Refund	Refund
Prediction	Non Refund	$c_{TP} = c_{cheap}$	$c_{FP} = c_{cheap} + c_{MR}$
	Refund	$c_{FN} = c_{exp}$	$c_{TN} = c_{exp}$

Table 4-7: Costs associated to the outcomes of prediction

Definition 4.9. *A loss function is the total cost for all predicted instances, associated with a specific score, s.*

$$C(s) = c_{TP} \cdot n_{TP}(s) + c_{FP} \cdot n_{FP}(s) + c_{FN} \cdot n_{FN}(s) + c_{TN} \cdot n_{TN}(s) \tag{4-12}$$

$$= c_{cheap} \cdot (n_{TP}(s) + n_{FP}(s)) + c_{MR} \cdot n_{FP}(s) + c_{exp} \cdot (n_{FN}(s) + n_{TN}(s)) \tag{4-13}$$

Definition 4.10. *The optimum loss (O_L), is the minimum of all $C(s)$, calculated over all possible scores, s.*

$$O_L = \min_{s \in [0,1]} C(s) \tag{4-14}$$

The optimum loss function is a practical oriented evaluation measure. The possibility to minimize the costs or losses does not require a decision on a threshold. A class imbalance is incorporated into the costs. The costs of the minority class will be much higher if more important.

However, the loss function does not take into account the variable transactional costs of every individual transaction. It works with static costs for different prediction outcomes. However, the real cheapest route is a combination of both routes, depending on instance costs. The optimum loss function works with constant costs, and is not designed for variable transaction cost functions.

Criteria	Support
Class imbalance	Yes
Decision threshold	Yes
Misclassification costs	Yes
Individual transaction costs	No

Table 4-8: Matching evaluation criteria for loss function.

4-8 Conclusion

This chapter described the most important evaluation methods from literature used to evaluate classification algorithms. These evaluation methods are tested to the criteria in the introduction of this chapter. In Table 4-9 we summarized the main capabilities of the discussed evaluation methods.

	Class imbalance	Decision threshold	Misclass. costs	Individual txn costs
OSR	-	+	-	-
Marginal Rates	-	+	-	-
AUC	+	-	-	-
AP	+	-	-	-
F_β measure	+	+	+	-
Loss function	+	+	+	-

Table 4-9: Overview of applicability evaluation methods

Using the constructed overview of evaluation methods, we are able to answer the following research question:

RQ10: Which existing evaluation methods measure performance considering the refund specific characteristics?

According to the introduced evaluation requirements, it is shown that existing classification evaluation techniques are not sufficient for evaluation of a classifier in transaction routing context. Therefore in the next chapter we will introduce an evaluation method which is able to evaluate a classifier using costs functions dependent on instance variables.

Chapter 5

Current Optimal Instance Score

5-1 Introduction

The prospective classifier must be evaluated taking into account *all* the evaluation criteria, as defined in Chapter 4. None of the discussed evaluation methods are designed to support variable transaction costs. Some methods can deal with static misclassification costs. However, transaction routing involves dynamic misclassification costs, depending on the transaction amount. This relates back to the following research question:

RQ11: How would we ideally measure the classifier performance which expresses the winnings in relation to payment routing?

In this chapter we introduce Current Optimal Instance Score, an evaluation method designed to support different instance costs by extending the loss function.

In this chapter we elaborate the individual transaction costs, and show how these can be optimized. These instance costs are incorporated in the loss function from literature. After defining the instance loss function, it will be transformed into an interpretable score. This score will be used as evaluation method during the rest of the research.

5-2 Instance costs

Transaction costs associated to routes are functions depending on the transaction amount. The transaction amount is also called transaction value, and is denoted by tv. Some routes charge fixed costs while others increase the fee as the amount of the transaction increases. Interesting combinations of the two types can be found in practice.

This section is build upon the cost function introduced in Section 4-7. Transaction costs for a route are not static any more, the cheap route can consist of a combination of multiple

routes, with different refund support. We describe routes in terms of support for refunds, because cheap and expensive are misleading. This changes the naming of cost functions of both routes. The costs of the route that supports refunds is defined as $c_R(tv)$, while the other, non-refund route is $c_{NR}(tv)$. Manual refund costs are defined as c_{MR}, and represents a constant cost.

The costs corresponding to individual transactions are shown in Table 5-1. The most important difference is that the costs of the original loss function are static, while the costs of the instance loss function depend on instance variables. *Note* the introduced input argument for the cost functions.

		Truth	
		Non Refund	Refund
Prediction	Non Refund	$c_{NR}(tv)$	$c_{NR}(tv) + c_{MR}$
	Refund	$c_R(tv)$	$c_R(tv)$

Table 5-1: Costs associated to the outcomes of prediction

The extra dimension in cost functions for individual transactions require a different optimization. The costs and prediction should be optimized together.

5-3 Cost optimization

Combining costs and prediction show how a predictor can influence optimization of transaction routing. This section elaborate on the following research question:

RQ1: How does the intervention of a classifier influence optimization of transaction routing?

In transaction routing a transaction predicted as refund can still be routed to the non-refund route, and vice versa. This is because the costs of the other route, independent of the prediction is cheaper, also in the case of a manual refund. In this section the cost differences between the routes, and its optimization, is explained.

Two example cost functions for the introduced routes, one with refund support, are shown in Fig. 5-1. An extra function is introduced which is derived from the non-refund route. The function consists of the non-refund route costs plus the manual refund costs, $c_{NR}(tv) + c_{MR}$. This is the cost function for FP, as showed in Table 5-1.

In practice, systems are configured to send all transactions to the refund route, the route that supports refunds. A simple cost optimization (without predictions) incorporating the cost differences, introduces two simple routing rules;

1. Send transactions to the refund route, *if* the costs of the refund route are lower than the non-refund route. This corresponds to $c_R(tv) \leq c_{NR}(tv)$, the left part in Fig. 5-1.

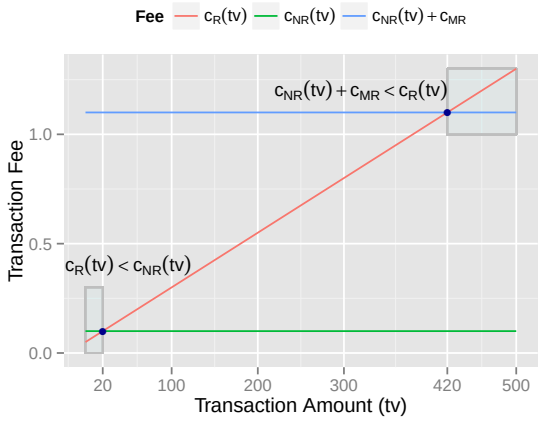


Figure 5-1: Example of two different cost functions where a third represents non-refund route costs plus the manual refund. Two different cost optimizations are highlighted.

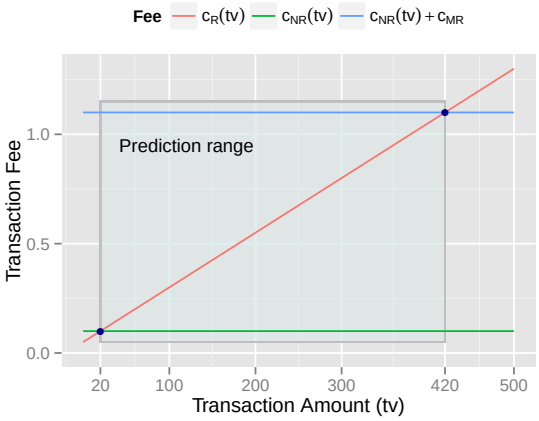


Figure 5-2: The area where predictions lead the least-cost routing to cheapest routes.

2. Send transactions to the non-refund route, *if* the costs of the non-refund route plus a manual refund are lower than the refund route. In this case every rerouted transaction is profitable. This corresponds to $c_{NR}(tv) + c_{MR} \leq c_R(tv)$, the right part in Fig. 5-1.

Both rules are added to the route decision tree, introduced in Section 4-7. The three represents the decision for a route and its accompanied costs. The decision tree with routing rules is visualized in Fig. 5-3.

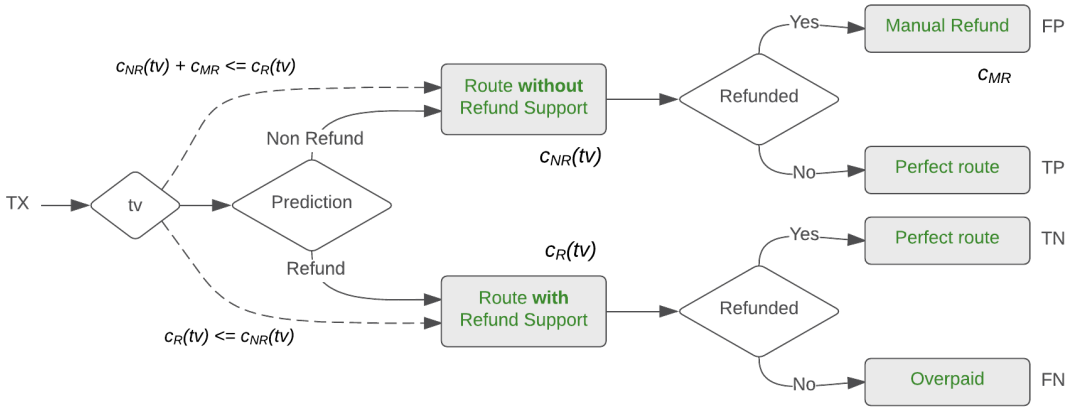


Figure 5-3: Decision tree using the predictor to route transactions depending on the individual transaction costs. The costs for specific parts in the tree are shown next to the parts in the form c_{xx}. The two dashed lines are conditional lines.

These rules should be incorporated into the evaluation method. Fig. 5-4 shows the high-level structure of the evaluation method in routing context. Decision logic exists before and after the predictor. The 'predictive power' of the classifier is measured in terms of its cost-

optimization. The logic before the predictor can holdout transactions from the prediction, so these are unimportant for evaluating the predictor. The logic after the predictor, deciding on the route, infer the individual costs for misclassification, see Fig. 5-2. Therefore both pre- and post logic should be incorporated in the evaluation method.

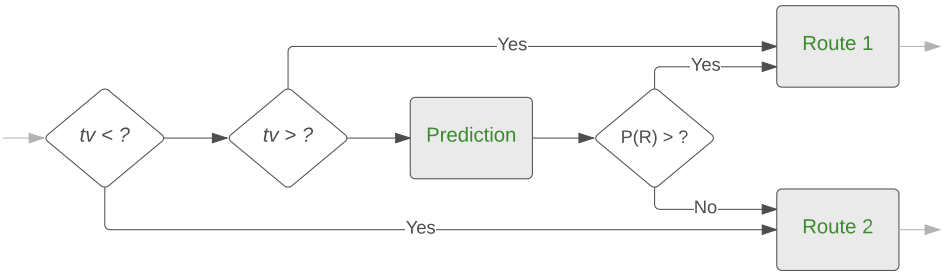


Figure 5-4: Evaluation in routing context.

The first routing rule is triggered if the costs of the refund route is cheaper, then this is the cheapest route for both refund and non-refund transactions. Sending refund and non-refund transactions over a refund route do not have any further consequences. This rule is independent of the prediction.

The second rule is triggered if the non-refund route costs including a manual refund are cheaper, in contrast to the refund route. The costs for refund and non-refund transactions differ, but are both cheaper than the refund route. The prediction does not change the decision for this route, what makes this rule also independent of the predictor. Therefore both rules belong to the logic applied before the prediction.

The introduced route rules change the calculation of the total route costs. Together with the instance costs we can define an instance loss function.

5-4 Instance Loss Function

The costs associated with the route decision outcomes are extracted from the decision tree and are shown in Table 5-2. First the cost conditions are checked, if they match, the corresponding route is decided. If no cost condition matches, the route decision depends on the prediction outcome.

			Truth	
Costs	Prediction	Route	Non Refund	Refund
1: $c_R(tv) \leq c_{NR}(tv)$		R	$c_R(tv)$	$c_R(tv)$
2: $c_{NR}(tv) + c_{MR} \leq c_R(tv)$		NR	$c_{NR}(tv)$	$c_{NR}(tv) + c_{MR}$
	3: Non Refund	NR	$c_{NR}(tv)$	$c_{NR}(tv) + c_{MR}$
	4: Refund	R	$c_R(tv)$	$c_R(tv)$

Table 5-2: Costs associated to route decisions generated from the outcomes of prediction together with transaction cost conditions. The numbers represent the sequence for the according decisions.

These costs can be reduced to three different cost routes. The cost are combined into a short cost overview, shown in Table 5-3.

			Truth	
Costs	Prediction	Route	Non Refund	Refund
1: $c_{NR}(tv) + c_{MR} \leq c_R(tv)$	**3:** Non Refund	NR	$c_{NR}(tv)$	$c_{NR}(tv) + c_{MR}$
2: $c_R(tv) \leq c_{NR}(tv)$	**4:** Refund	R	$c_R(tv)$	$c_R(tv)$

Table 5-3: Shorthand overview of Table 5-2, costs associated to route decisions. The numbers represent the sequence for the according decisions.

With these conditions and cost functions we defined a loss function. This function is called instance loss function and is described in Definition 5.1 and Definition 5.2.

Definition 5.1. *We define the instance loss function $C_{IL}(X, s)$ as the sum of all instance costs, associated with a specific score, s. Where $X = \{tv_i \ ; \ i = 1 \dots N\}$ and tv_i is the transaction value for transaction i. TP, FP, FN and TN can be determined given score s.*

$$C_{IL}(X, s) = \sum_{i=1}^{N} \begin{cases} c_{TP}(tv_i) & \textit{if TP} \\ c_{FP}(tv_i) & \textit{if FP} \\ c_{FN}(tv_i) & \textit{if FN} \\ c_{TN}(tv_i) & \textit{if TN} \end{cases} \tag{5-1}$$

$$= \sum_{i=1}^{N} \begin{cases} c_{NR}(tv_i) & \textit{if TP} \\ c_{NR}(tv_i) + c_{MR} & \textit{if FP} \\ c_R(tv_i) & \textit{if FN or TN} \end{cases} \tag{5-2}$$

The instance loss function is dependent on a certain threshold decision. The optimum loss is calculated by minimizing the costs for all available thresholds.

Definition 5.2. *We define the optimal instance loss as the minimum of all $C_{IL}(s)$, calculated over all possible scores, s.*

$$O_{IL}(X) = \min_{s \in [0,1]} C_{IL}(X, s) \tag{5-3}$$

The optimal loss function represents the costs of all transactions. Minimization of the costs result in a certain decrease in costs, which show how much money is saved. The costs are in practice easy to understand, but for evaluation of a classifier hard to position. This is because the optimal- and worse costs are not known. It is not clear what the aimed costs, say the ultimate goal is. The costs cannot be minimized to zero because success routes also have costs associated, see Table 5-3. Therefore, we define a score between 0 and 1, representing the normalized costs between worst and optimal situation. The score is called Optimum Instance Score (OIS) and is defined in Definition 5.5.

Definition 5.3. *We define the highest costs, C_{MAX}, as the sum of all transaction costs, when all predictions are the opposite of the truth.*

Definition 5.4. *We define the optimal costs, C_{MIN}, as the sum of all transaction costs, when all predictions are equal to the truth.*

Definition 5.5. *We define the Optimum Instance Score (OIS) as the complement of the normalized costs between C_{MAX} and C_{MIN}. The score ranges between 0 and 1. The OIS is defined as follows:*

$$OIS(X) = 1 - \frac{O_{IL}(X) - C_{MIN}}{C_{MAX} - C_{MIN}} \tag{5-4}$$

A classifier designed for routing can be evaluated with the OIS. A certain score shows the performance in relation to minimal and maximal performance, with a scale that depends on the route costs. From a prediction perspective this linear scaling is an interpretable measure. However, from a business perspective, we can enhance the method so that it is interpretable for the business as well.

5-5 Back to Business

The new evaluation score is introduced mainly to facilitate evaluation of the classifier in the given context. However, OIS is still a bit difficult to interpret in practice. This is due to the fact that it is normalized between 0 and 1, representing the prediction power in terms of costs winnings. In this section OIS is expanded to **bridge the gap between theory and practice**.

The OIS can be shown in relation to the practical transaction route costs. This relation is shown in Fig. 5-5. OIS (reversed) spans between the worst and optimal transaction route costs. In this perspective the OIS is reversed (0 is top and 1 is bottom).

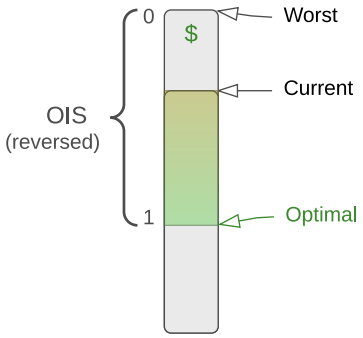


Figure 5-5: OIS in context of transaction routing costs.

Current practice is that merchants solve the routing issue by sending all their traffic to a route that supports refunds. These costs, or current OIS are the baseline to compare the new OIS scores with. Scores lower than this value are not interesting. This makes the current costs comparable the *random score*, the baseline in a theoretical measure, for example Area Under ROC (AUC). The colored part in Fig. 5-5 shows the part in which scores are optimized in practice.

In practice cost reduction would be measured by the percentage of gain. This is compared to the current situation. This is the most interpretable way of describing the performance of the system in practice.

Definition 5.6. *We define the current costs, C_{CURR} , as the sum of all transaction costs, when routing all transactions to the route which supports refunds.*

Definition 5.7. *We define the cost reduction as the percentage of costs reduced from the current situation. The cost reduction is measured in units of percentage. Cost reduction is calculated as follows:*

$$\frac{C_{CURR} - O_{IL}(X)}{C_{CURR}} \cdot 100\% \tag{5-5}$$

This measure is used to show the practical impact of the cost optimization. For interpretability, the current costs should be the baseline for the measure. The 'worst case' is when the system performs equal or worse than the current situation.

We can enhance the OIS by changing the worst costs C_{MAX} by the current costs C_{CURR}. The current costs is used as lower bound in Current Optimal Instance Score (cOIS), which apart from the lower bound has the same functionality as OIS.

Definition 5.8. *We define Current Optimal Instance Score (cOIS) as the complement of the normalized costs between C_{MAX} and C_{CURR}. The score ranges between 0 and 1. cOIS is defined as follows:*

$$cOIS(X) = 1 - \frac{O_{IL}(X) - C_{MIN}}{C_{CURR} - C_{MIN}} \tag{5-6}$$

The measure cOIS scales over the colored part in Fig. 5-6. Theoretically the score can go below zero, when the costs are higher than the current costs. cOIS is a combination of practical- and theoretical context by scaling from the current situation (practice) to the max reachable situation (theory).

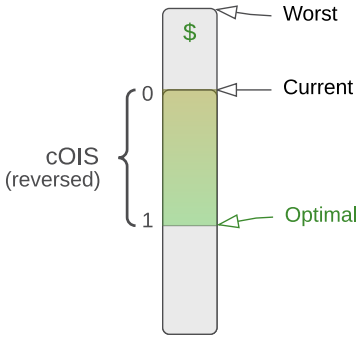


Figure 5-6: cOIS in context of transaction routing costs.

5-6 Evaluation of cOIS

cOIS is evaluated by comparing the performance to other evaluation methods. The following two methods will be compared:

- Loss function
- Current Optimal Instance Score (cOIS)

The loss function is built upon static misclassification costs, recall Section 4-7. In order to apply the loss function and evaluate a trained classifier we need to estimate the misclassification costs.

Estimating static costs

A loss function validates a predictor based on a real cost function. The different errors will be weighted differently. However, our cost functions are continuous and differ per transaction, while misclassification costs are static. Recall the transaction decision tree in Fig. 4-5. From this decision tree we can extract the two different misclassification cost.

- *Costs of False Positive* A transaction resulting in a refund, send to the non-supportive route, results in manually processing the refund. The costs of a manual refund is charged, $c_{MR} = 1$.
- *Costs of False Negative* A transaction sent to the supportive route that does not results in a refund is overpaid. 'Overpaid' is the difference between the two continuous cost functions. A static misclassification value should be estimated from these continuous functions.

The costs of overpaying a transaction is the difference between the two alternative routes. The two routes have been introduced in Chapter 1 and shown in Fig. 1-4a. We calculate the difference for all non-refund transactions and visualize the distribution in Fig. 5-7. Only transactions from the non-refund class can be overpaid and thus incorporated in the cost estimation.

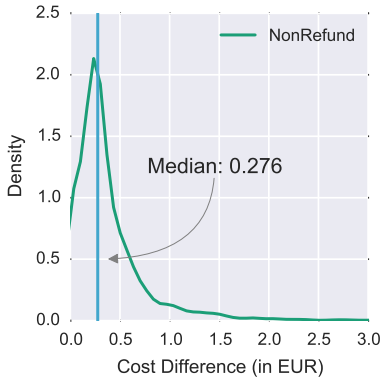


Figure 5-7: Distribution of route fee differences between two alternative routes for all non-refund transactions in the train set.

To estimate the static misclassification cost for false negatives, we take the median (€0.276) of all route differences in the training set. The cost of a false positive equals manual refund costs, set on €1. An overview of the misclassification costs for the loss function can be found in Table 5-4.

		Truth	
		Non Refund	Refund
Prediction	Non Refund	$c_{TP} = 0$	$c_{FP} = 1$
	Refund	$c_{FN} = 0.276$	$c_{TN} = 0$

Table 5-4: Estimated costs associated to the outcomes of prediction

Experiment In this section we will describe the experiment of searching optimal parameters for two equal classifiers using two different evaluation methods, Current Optimal Instance Score (cOIS) and the loss function. The final classifiers are compared using a 'neutral' measure, costs.

In Table 5-5 a summary of the results can be found. The costs are visualized in Fig. 5-8. The experimental setup can be found in Appendix C-1 and the details of the experiment in Appendix C-2.

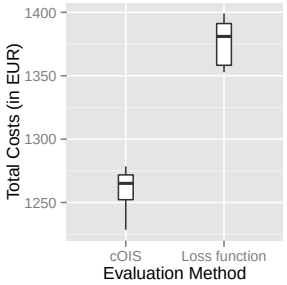


Figure 5-8: Results of two estimators optimized with different evaluation methods.

	Selected Parameters			Total Costs	
Evaluation Method	Tree Depth	# Trees	Threshold	Mean	Sd
Loss function	16	1000	0.81	1376.11	17.68
cOIS	1	32	0.71	1260.76	15.22

Table 5-5: Results for training estimators with different evaluation methods.

cOIS decided on a more simple model, using 32 trees with just one split, called decision stumps. We can see that a classifier, optimized using cOIS, minimizes the costs significantly better compared to the usage of the loss function. cOIS scores on average €115 lower, which corresponds to 8% of the costs resulted from the classifier optimized using the loss function.

This result is the cross-validation score over the outer train set. 10 folds are used and thus $1/10^{\text{th}}$ of these transactions are involved in calculating the test score for each fold. To get grip on the difference (what is €115?), this score is multiplied for the folds, and scaled to the number of transactions per year. *The difference between using cOIS or the Loss function for this merchant is over €8.600,- per year, in favor of cOIS.*

This is just a simple example in optimizing 2 parameters together with the decision threshold. The two tested classifier variables are both influencing the prediction power, and indirectly the transaction costs.

5-7 Conclusion

According to the introduced evaluation requirements, it is shown that existing classification evaluation techniques are not sufficient for evaluation of a classifier in transaction routing context. We defined an ideal evaluation method that fulfills all criteria from Chapter 4. We introduced two methods:

1. Instance loss, an extended loss function which is able to evaluate a classifier using costs functions dependent on instance variables.
2. Current Optimal Instance Score (cOIS), a linear transformation of the optimum instance loss function.

By creating the ideal evaluation method, the following research question is answered:

RQ2: What is the minimal classifier performance to enable a profitable situation for transaction routing?

The evaluation method is the perfect mix between theory and practice. cOIS scales from the current practical cost situation to the theoretical optimum. Due to the range of the measure, a negative score will not be profitable, and a positive score will be profitable.

cOIS is able to evaluate a classifier using costs functions dependent on instance variables. In the rest of the research we will calculate the performance of a classification method using cOIS.

Chapter 6

Designing the Predictor

6-1 Introduction

Designing a predictor requires several design stages. Following Theodoridis and Koutroumbas[29], these stages include *a*) feature generation and *b*) selection, *c*) classifier design and *d*) system evaluation. The system is evaluated using Current Optimal Instance Score (cOIS), the evaluation method defined in Chapter 5. This chapter will elaborate the remaining stages; feature generation and selection, and classifier design.

Feature generation and selection are elaborated in a section together. We will explain which features are derived from payment data and which features are derived from additional sources. The actual feature selection, for usage in the classifier, will take place in the feature selection experiment, Section 7-4.

The classifier design stage is where the type and algorithm of a estimator is chosen. The estimator is the algorithm that is able to draw a figurative line between all transactions, where each side of the line represents a class, refund or non-refund. Most estimators have parameters to change the behavior of the estimator, these are tested during the experiments, Chapter 7. In this section an estimator is chosen by describing and evaluating different widely used classifiers.

6-2 Feature Sets

This section will describe the feature sets used in this research. A feature set is defined as a set of features, simple as that. These features are represented as sets to group features which are retrieved from the same data source. Features are the data variables which are given to the classifier in which it searches for patterns.

These explanatory variables can be direct properties of the transactions, but also derived or computed and still associated to the transaction. This section will explain the following four feature vectors in their appropriate subsections.

1. Transactional Attributes
2. Risk Checks
3. User History
4. Regional Statistics

6-2-1 Transactional Attributes

Transactional Attributes are the set of direct properties of a transaction. A lot of the available properties depend on where to the transaction is send to, which payment method is used, what kind of extra services the merchant is using etcetera. The routing of transactions takes place before it is send to an acquirer. This is shown in Fig. 6-1. Therefore only a subset of all information can be used for classification. The 'less information' restrictions will make it harder to predict refunds. This paragraph will elaborate on the available information, defined by the question:

RQ4: What information can be retrieved at the time of the refund prediction?

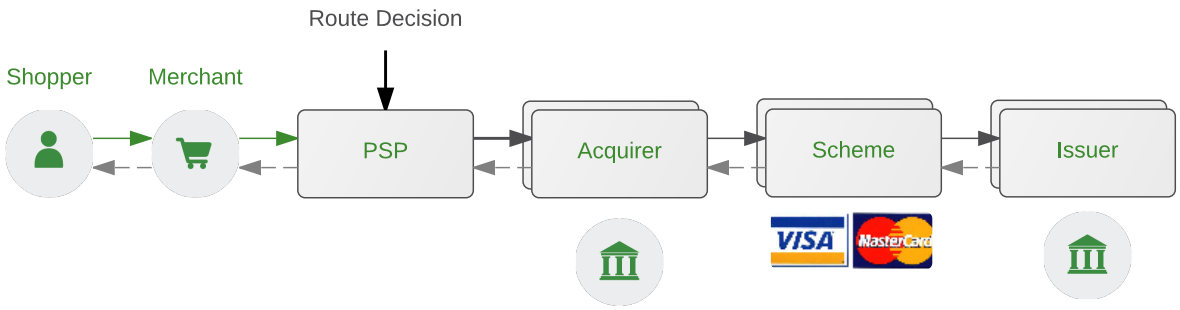


Figure 6-1: Route decision in context of full transaction process. Information flow is shown between stakeholders. The green arrows show the available information for the route decision.

The green arrows show that only the information from the merchant and thus shopper can be used as source for the decision. These properties are the information that is send with a payment, such as shopper information, like the name, IP and address. A part of the currently available information is collected in realtime, such as risk scores, which we will discuss in Section 6-2-2.

The attributes of a transaction can be used as basic feature vector. An overview of the available attributes can be found in Table 6-1. Some attributes are explained in short in this section.

Merchant Account It's an internal account structure used at the industrial partner. Within a company multiple merchant accounts can be created to separate configurations and reporting. Every company can determine their own merchant account configuration following their own strategy. In practice, the configuration of merchant accounts mostly follow the separation between countries and online/offline type of transactions.

3D Secure responses The result codes of the 3D Secure (3Ds) communication protocol. The protocol is executed when the transaction arrives at the Payment Service Provider (PSP), and

Attribute name	Format	Short description
Amount	Numeric	Amount of transaction in Euro's
Merchant account	Categorical	Sub account of merchant
Issuer country	Categorical	NL, BE, DE ...
Shopper country	Categorical	NL, BE, DE ...
Directory response	Categorical	3D Secure enrollment: [Y, N, U, -]
Authentication response	Categorical	3D Secure authentication: [Y, N, U, A, -]
System risk score	Numeric	Risk score default configuration
Account risk score	Numeric	Risk score custom configuration

Table 6-1: Transactional attributes

before it is send to an acquirer. 3Ds is designed to authenticate the shopper at the playform of the issuer. Authentication at the issuer makes the issuer liable for fraud if the shopper is authenticated successfully. A more extensive description of the 3Ds protocol can be found in Appendix B.

The directory response shows if the shopper is enrolled (Y) in the program or not (N). The authentication response shows if the shopper is successfully authenticated (Y) or not (N). An extra value '-' dash is used for erroneous or missing data, if the 3Ds process was not executed. More information about the 3Ds responses can be found in Table B-1.

Risk scores There are two risk scores available at transaction routing; a system and account score. The system score is calculated with default configuration and the account score with custom configuration. Both values start at zero and the higher the value the more suspicious it is, where negative values represent trusted transactions. Usually the values are between 0 and 100, where the transaction is refused if the score is above 100. Missing values (risk is not calculated) are corrected with a zero.

Generated Features

Next to the direct properties we also generated features from properties. Some features are derived from properties not listed in Table 6-1. These 'raw attributes' were not useful as features directly. The generated features are listed in Table 6-2 and explained shortly in this section.

Feature name	Format	Short description
Time	Cos/Sin	The time the transaction was initiated
Day of week	Cos/Sin	The day of the week the transaction was initiated
Day of month	Cos/Sin	The day of the month the transaction was initiated
Email domain	Categorical	The domain of the shopper's email address
Different country	Boolean	Location of shopper differs from location issuer

Table 6-2: Extracted Attribute Features

Different country The shopper is performing an international transaction. The location of the issuer bank is different from the location of the shopper.

Time and Day From the transaction arrival date, we extracted specific information about the time of arrival and the day of arrival (in relation to the week or month). To handle the cutoff between the end- and beginning of the timespans we format them as pair of two features encoded as Cos/Sin. We refer to this encoding as the *cosinus/sinus encoding* (Cos/Sin in Table 6-2). We show an example how the time of the day is encoded in Cos/Sin;

$$Time = (f_1, f_2) \tag{6-1}$$
$$f_1 = cos(2\pi \cdot t/T) \tag{6-2}$$
$$f_2 = sin(2\pi \cdot t/T) \tag{6-3}$$

Where t is the time of the day and T the total time in a day (both in minutes).

Remarks

The following two notable things can be observed from the listed transaction attributes;

1. The number of features are fairly low, and
2. Most of the attributes are categorical

As discussed in Section 2-6 all categorical features must be converted to numerical data. Boolean format can be represented by 0 and 1. Categorical features require special preprocessing to be translated into a numeric representation. In Section 2-6 different types of categorical encoding and their performance are explained.

Due to the low number of transactional attributes we extracted more information form several sources. These sets are described in the next sections.

6-2-2 Risk Checks

One of the extra services the industrial partner offers to merchants is one to prevent fraudsters. The risk module consists of 80 different checks, together determining the risk of a transaction. In practice these checks are accompanied with a configuration that weights every check and sums the triggered checks into a score, the account/system risk score introduced in transactional attributes.

All checks are formatted as boolean, stating if the check triggered. A risk check can be based on different information. Sometimes this is expressed as integer accompanied with a threshold. It would be best to incorporate these numeric sources as features, however this would acquire too much time for this research.

In Table 6-3 a few risk checks are shown[1]. Most risk checks find their roots in fraud classification. Due to the low fraud rate these are triggered only often. The variability in the individual checks is very low. Therefore only a subset of these features can be used.

[1]Not all risk checks could be stated explicitly due to secrecy of the industrial partners risk system.

Risk check	Format	Short description
Check 1	Boolean	Cross border payment
Check 2	Boolean	Payment from high risk country
...	...	...
Check 79	Boolean	High amount
Check 80	Boolean	Velocity of succeeding transactions

Table 6-3: Examples of risk checks

6-2-3 User History

Feature vectors are the finite set of information on which the classifier will find its patterns. We want to show that refund patterns can be shown in user behavior. These patterns could be known on forehand, or identified by the learning algorithm. The features of the learning algorithms should include the information to identify these patterns. Features for user history are generated through transaction aggregation.

Transaction aggregation

Transaction aggregation is a strategy to generate features from a succession of transactions. Multiple transactions are linked to each other following some criteria. Properties from transactions can be aggregated into a numerical feature. Even categorical properties fulfill. An example feature could be the number of transactions in a certain category. Feature generation by transaction aggregation is shown successful for fraud detection [10, 11, 7].

We will link transactions based on card or shopper information, such that it represents a user history. Transactions are linked if one of the following properties matches.

- Card Information
  - Card Number
  - Bank Account
  - IBAN
- E-mail address
- Shopper reference[2]

The matching properties above form the based for defining 'users'. The linked transactions represent a user history. Using transaction aggregation features can be generated for this user. The features based on user history can be found in Table 6-4.

Take the first feature as example, the number of transactions over a given time period. This feature calculated over 1 month can positively correlate to the refund class while the same feature calculated over 9 months could result in no correlation or even a negative correlation. It is shown that different aggregation periods can correlate differently in fraud research [10]. Therefore we calculate the user history features from Table 6-4 over three different timespans;

[2] A reference to the user as it is in the system of the merchant.

Feature name	Format	Short description
Txn number	Numeric	The number of received transactions in the time period.
Average amount	Float	The average transaction amount (in Euro's) over the given time period.
Refund ratio	Float	The percentage of refunded transactions over the given time period.

Table 6-4: Features extracted from user transaction history. All given history features are calculated over three different time periods; last month, three months, and ten months.

- last month
- last 3 months
- last 10 months

This results in a total of 9 additional user history features.

6-2-4 Regional Statistics

One transactional property is not utilized yet, the delivery address of the shopper. An address is no use for a machine learning algorithm directly. In order to be useful, it must be transformed into numerical data.

Statistics Netherlands (CBS) provides all kind of statistical information for the Netherlands. We downloaded[3] statistics about the Dutch population per region from CBS. It is grouped by the 4-numbered zip code (originally there are four numbers and two letters). The retrieved statistics contain the following information[4];

- Population by gender, age and zip code
- Immigrants by ethnic and zip code
- Private households by size and zip code

The delivery address of a transaction can be connected to the population and household statistics of its region. The statistics, as retrieved from CBS, should be preprocessed and transformed to ratios per region. For example, we transform the counts of population by gender to ratios to prevent influence by the size of the region. The size of a region is also used as feature. The extracted features are the following;

- Ratio of total people

[3]To find these statistics go to the (Dutch) 'Open data Statline' and click sequentially on 'Kies thema', 'Bevolking', 'Bevolking en huishoudens', 'Bevolking per postcode', and 'Bevolking; postcode; 2013'.

[4]Due to statistical secrecy the counts per postcode region is randomly rounded to multiples of 5.

- Difference according to mean total people
- Ratio of men
- Ratio of women
- Ratio of immigrant
- Ratio of western immigrant
- Ratio of non western immigrant
- Ratio of one-person household
- Ratio of more-person household without kids
- Ratio of more-person household with kids
- Ratio of household size
- Difference according to mean household size

A limitation for these features is that it only applies to transactions with Dutch delivery addresses. Experiments which incorporate these features should be done using only Dutch transactions.

6-2-5 Overview

We have retrieved the following four feature sets;

1. Transactional Attributes
2. Risk Checks
3. User History
4. Regional Statistics

The first two sets are retrieved directly from the system, where additional features are extracted from raw values. The user history is extracted by applying transaction aggregation on the transactional history of the user. These features are calculated over three different time periods. At last we have retrieved regional statistics from CBS.

While the risk check are all booleans and user history and regional statistics are all integers, the formats of the transaction attributes differ. Some features are formatted as categories which require additional analysis to be useful for a classifier.

6-3 Classifiers

In this section we will explain and test three common classifiers; *a*) Random Forest, *b*) AdaBoost, and *c*) Support Vactor Machines. These classifiers perform best in studied fraud research. We will study these classifiers to answer the following research question:

RQ8: Which classification algorithms perform best for refund classification in context of routing decisions?

Before describing the classifiers we will shortly describe a classification tree, which is used internally by the former listed classifiers.

6-3-1 Classification Tree

A classification tree, by Breiman et al.[30], follows the tree structure where each node represents a decision in the form of *"is feature* $x_i \geq \alpha$*?"*. Leaf nodes represent the predicted class. Such a tree is sometimes also called decision tree. An example of a classification tree is given in Fig. 6-2. In a classification tree, a rule is defined as the combined if-else conditions from one path in the tree, from the root to a leaf.

Tree based models are easy to interpret as a leaf defines a specific outcome and the path to a leaf is how the model came to this outcome. A drawback of tree classifiers are high variances due to the nature of tree hierarchy. A few small changes in the training data can generate a total different tree.

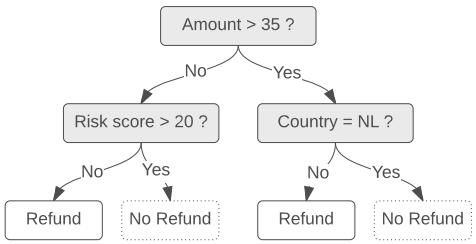


Figure 6-2: Example of classification tree. Each node/split represents a decision based on a feature and threshold. Leaf nodes represent a class.

How to create a classification tree? A node t splits up into two subnodes t_Y and t_N respectively for the 'Yes' and 'No' answer to the node question. Alike, the training set X_t of node t, is split into two subsets X_{tY} and X_{tN}. Let $P(w_i|t)$ be the probability that an instance of X_t belongs to class w_i. In this example decrease of impurity is used as measure to split nodes. To create nodes and their criteria, do for each node (recursively, start with root node):

1. Compute node impurity $I(t)$ for each candidate question. This forms the combinations of features x_i and their possible values.

$$I(t) = -\sum_{i=1}^{M} P(w_i|t) log_2 P(w_i|t) \tag{6-4}$$

2. Decide on question (feature, value combination) with the highest decrease of impurity $\Delta I(t)$.

$$\Delta I(t) = I(t) - \frac{N_{tY}}{N_t} I(t_Y) - \frac{N_{tN}}{N_t} I(t_N) \tag{6-5}$$

3. Stop creating nodes when stop criteria are met.

4. Define class w_j in leaflet nodes according to majority rule.

$$j = arg \max_i P(w_i|t) \tag{6-6}$$

6-3-2 Random Forest

Random Forest is an ensemble classifier, which combines several other classifiers with the idea that combined learners achieve a better performance. *Random Forest (RF)* uses the idea to combine multiple weak learners into one strong classifier. A weak learner is a simple model trained on less data or features, and performs slightly better than random guessing. The combined outcome of all learners together define the outcome of the Random Forest classifier.

RF creates a 'forest' of multiple classification trees[31]. Random forests reduce the variance of trees by combining multiple and therefore improves generalization error performance.

The decision trees are generated using a random sample of F features at each node, using the same training data for each tree. F is a user defined value with $F \leq l$, with l number of features. All trees could be generated in parallel due to the independence of the trees, which makes the Random Forest classifier computational efficient.

6-3-3 AdaBoost

AdaBoost uses a boosting approach to improve its performance in iterations. Boosting can be seen as combining multiple weak learners (like RF), however a different strategy is used to train the weak learners. A series of (weak) classifiers is trained iteratively, using a different subset of the data. Each iteration samples are weighted according to misclasifications, emphasizing the 'hardest to classify'. The final classifier is a weighted average of all trained learners. AdaBoost is such a classifier using decision trees as weak learners.

The boosting approach can reduce the error on the training set to a arbitrarily low rate. A pleasant property of AdaBoost is that it does not overfit, independent of the high number of parameters and the low error score on the training set. For enough number of rounds, the test error keeps decreasing while the training error already converged to zero.

6-3-4 Support Vector Machine

In simple two-class classification task, a classification method will try to find a function $g(\boldsymbol{x})$ which separates the indices between both classes. Any new instance will be easy to classify according to its position regarding the hyperplane. The two different hyperplanes in Fig. 6-3 both separate the classes well, but might classify future points different. The hyperplanes have

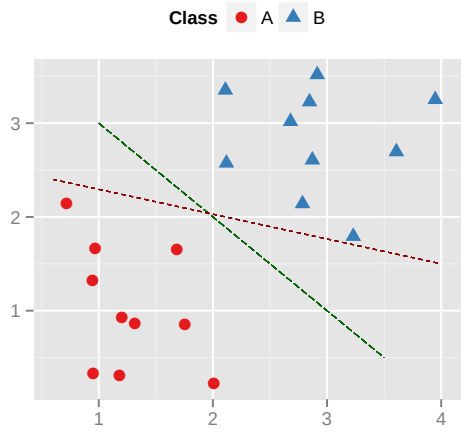


Figure 6-3: Example of separating two classes with two different hyperplanes. The green separator would probably work best for new points.

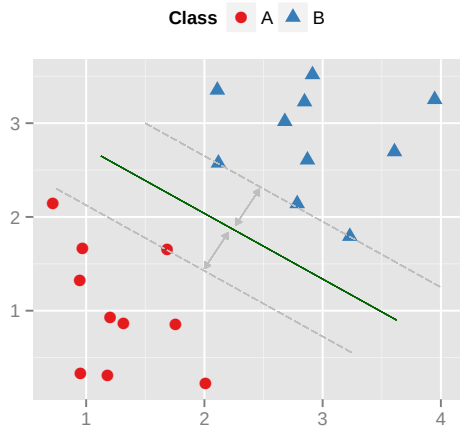


Figure 6-4: Example of separator maximizing the margin between the classes.

been optimized with regards to its error rate, however a lot of possible candidate functions could be applicable with the same or without error rate.

A Support Vector Machine (SVM) optimizes the function $g(\boldsymbol{x})$ which separates the two classes. The optimization maximizes the margin between the closest instances from different classes, see Fig. 6-4. Lots of candidate functions could separate the two classes, however only one $g(\boldsymbol{x})$ separates the classes best. This function is known as the *maximum-margin hyperplane*[29].

The classes of the given example are separable, and thus a hyperplane with maximum-margin hyperplane can be found. Now consider a hyperplane somewhere in between non-separable classes. The instances fall into one of the following three categories;

1. Correctly classified instances, not within the margin of the selected hyperplane.
2. Correctly classified instances, whithin the margin of the selected hyperplane
3. Misclassified instances

SVM still maximizes the margin but with a different restriction, to keep the instances that fall into the second and third category as low as possible.

6-3-5 Overview & Performance

RF is computationally efficient because it can train the internal trees in parallel. AdaBoost and Support Vector Machines have a longer computation time. However, as training time is important during the research for exploration and experiments, it is of less importance for the use in practice. The model can be trained periodically, as long as the prediction is efficient.

A small *experiment* is done defining the best performing classifier. The performance of the classifiers is measured in terms of cOIS, introduced in Chapter 5. In short, it ranges from the current route costs to the optimum, resp. 0 and 1. The scores in between are scaled costs based on the prediction outcome.

The comparable 10F-CV scores are shown in Fig. 6-5. The linear SVM, called Support Vector Classifier (SVC) in the results, performs stable, but ends up last in performance. It is shown that AdaBoost can reach the highest scores, however not generalized very well. The average score of AdaBoost is lower than RF. RF performs stable across different folds, and on average performs best. This experiment showed that RF is the best choice to use in future experiments.

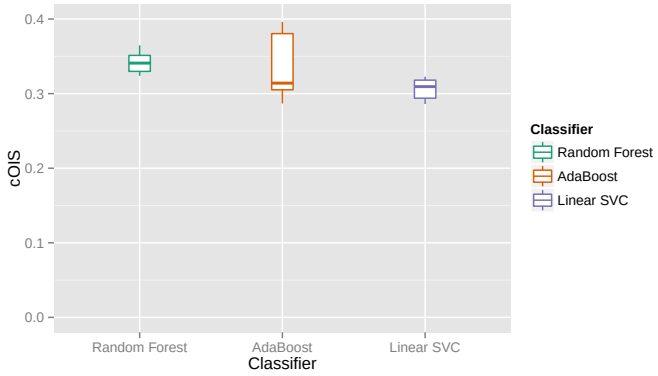


Figure 6-5: Results for training three different classifiers on a subset of data.

6-4 Conclusion

In the first part of this chapter, the features are defined, collected in particular sets. A subset of transaction attributes can be used due to the early prediction in the transaction process. Categorical features require additional analysis to be transformed into numerical features.

Additional feature sets are retrieved or generated, including risk checks, user history and regional data. Additional research can show the usefulness of these retrieved or generated sets of features. The feature selection stage, as introduced in the introduction of this chapter, includes selecting the optimal subset of features from these sets and is elaborated in Section 7-4.

The second part of this chapter described three common classifiers from fraud research; RandomForest, AdaBoost and SVM. A quick experiment shows that AdaBoost performs well, but does not generalize very well. RandomForest performs quick, scores good and has the lowest generalization error. Therefore we choose to elaborate experiments using RandomForest.

Chapter 7

Experimental Results

7-1 Introduction

As seen in the previous chapter, the Random Forest (RF) classifier performs best and therefore is used for all experiments in this research. This chapter is structured as follows:

- Baseline
- Experiment 1: Categorical Encoding Methods
- Experiment 2: Feature Sets
- Experiment 3: Sample Weight Strategy
- Best design

Experiments are elaborated independent, but based on a shared baseline. This is visualized in Fig. 7-1. From left to right: a baseline is created, several experiments are executed, based on the baseline. The best model includes knowledge and design decisions from the experiments. The structure of the experiments, used tools and selected data is explained in Appendix C-1.

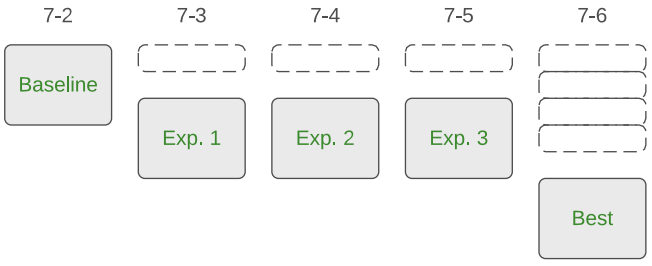


Figure 7-1: Structure of experiments. From left to right the 5 core steps of the experiments.

7-2 Baseline

To the best of our knowledge, refund prediction is not yet executed and described in literature. A baseline for refund prediction is necessary to show how we can improve the transaction routing. This section is divided into three parts, each with following objective;

- Show that *refund classification* works
- Create a *baseline for refund prediction* in routing context, and thus enabling comparison to this baseline
- Show that using refund predictions we can *reduce costs by route optimization*

7-2-1 Refund Classification

The main goal of this section is to show that refund classification works. Recall the research question about refund classification:

RQ3: Is refund prediction possible with transactional data, and how does it perform in general?

This involves the accuracy of the classifier, however due to class imbalance we choose Area Under ROC (AUC) as evaluation measure. Using AUC does not require a decision threshold, because it gives a good overall performance over all thresholds.

The classifier is designed using numerical transaction attributes. The best parameter configuration will be searched for the baseline classifier. The parameters are varied in such a way that the classifier changes from simple to complex models. Details about this approach and parameters can be found in resp. Appendices C-1-4 and C-3.

The classifier performs best with 2000 trees of depth 12. This results in a cross-validation AUC score of 0.731, with standard deviation of 0.0087. In general this score is not that good and thus using such a classifier for optimization will be a challenging job. In the rest of this section we will show that it is good enough to route transactions to the right route.

7-2-2 Baseline for Transaction Routing

In this section a baseline will be created. The baseline is created so that other experiments can be compared to this baseline. This enables us to show the increase (or decrease) of performance, compared to the baseline.

A baseline is created using a small selection of the features. The numerical features of the transactional attributes are used as feature vector. The categorical attributes are not part of the baseline due to additional analysis, which is done in Section 7-3.

Baseline score The cross-validated performance of a predictor trained with the selected best parameters on the outer training set is 0.34144. The used parameters and result is shown in Table 7-1. In the rest of this chapter we will refer to this result as the *baseline score*.

7-2-3 Reduce Costs by Route Optimization

Routes can be optimized using refund prediction in terms of costs savings. Due to the construction of Current Optimal Instance Score (cOIS), and the baseline score of ~ 0.34 from the previous section, the conclusion is obvious; Yes, we can save costs when optimizing transaction routing. This section gives a little bit more detail into how the costs are structured and how big the cost reduction is.

Types of optimization

Not every transaction requires a refund prediction in the routing process. Recall Fig. 5-3, where transactions are sent to a route based on their transaction value, and corresponding fees of both routes. To explain the cost savings we divide the transactions into two domains, corresponding to two types of optimization;

- *Least-cost Domain* These transactions are routed to their least-cost route if they do not require prediction knowledge. The transactions are selected by the routing rules introduced in Section 5-3.
- *Prediction Domain* The remaining transactions (not in least-cost domain) are routed according to their prediction.

Transactions correspond to one of the optimizations based on their transaction value. In Fig. 7-2 we visualized split between these domains over the transaction value. The blue region corresponds to the least-cost transactions. The cost reduction is analyzed for both domains.

Cost reduction

The cost reduction can be explained by application of each type of optimization. Starting with the current situation, this results in the following 3 situations;

Selected Parameters			cOIS	
# Estimators	Max Depth	Threshold	Mean	Sd
32	1	0.71	0.34144	0.01345

Table 7-1: Baseline score for best performing parameters on 10F-CV with the outer training set.

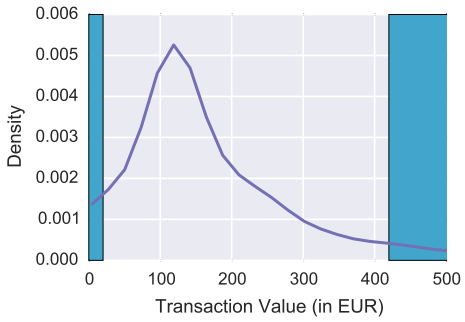


Figure 7-2: Transaction distribution over its value. Transactions in highlighted (blue) domain are routed independent of the predictor (least-cost optimization).

- *All Refund* The first situation, all refund, is where we started before this thesis, all transactions to the refund route.
- *Apply least-cost optimization* The second situation differs by sending all transactions in the least-cost domain to the least-cost route. With this optimization the costs of the least-cost domain are decreased by 68%.
- *Apply prediction optimization* The third situation differs (from the second) by routing the transactions, in the prediction domain, according to the predictions. With this optimization, using the baseline classifier, the costs of the prediction domain are decreased by 16%.

The costs for these three situations are shown in Fig. 7-3. Applying both optimizations result in a total **cost reduction of almost 30%** compared to before this thesis.

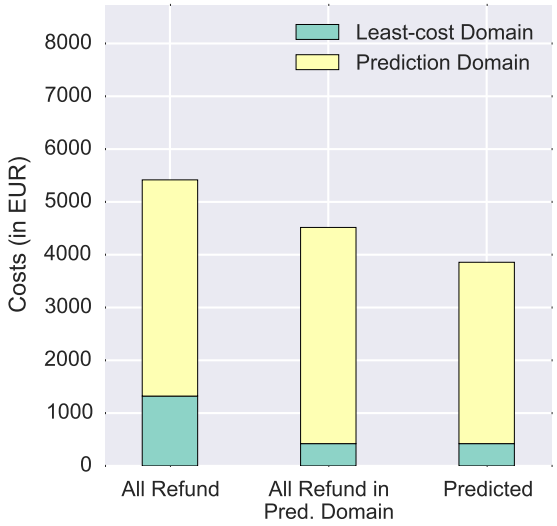


Figure 7-3: Shows cost reduction per domain for the old- and new situation using the baseline predictor.

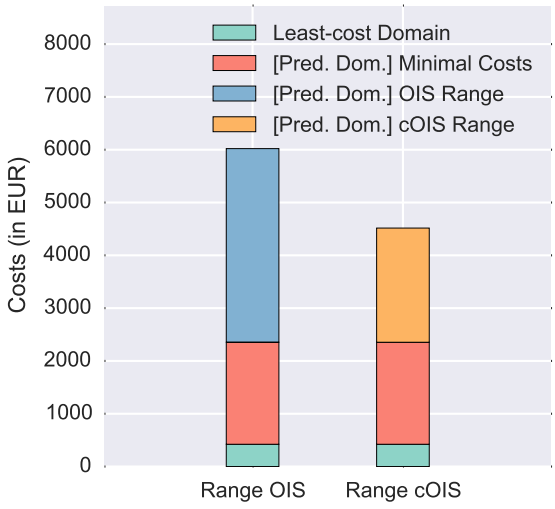


Figure 7-4: The theoretical share of different costs; static, per domain or evaluation methods.

Opportunity

A cost reduction of almost 30% implicates another 70% to go, for real fanatics. However, this is not possible due to the transaction costs for success routes. This will become clear when explaining the different parts of the total costs.

In Fig. 7-3 we have split the costs by type of optimization. The least-cost domain is routed optimally, from the second situation, so we cannot influence these costs with our predictions. The costs within the prediction domain cannot become zero because the success routes are also associated to specific costs. The minimal prediction costs, for an optimal prediction is also static.

A theoretical detailed costs specification of the prediction domain is shown in Fig. 7-4. The Optimum Instance Score (OIS) range, are the costs that can be caused by misrouting through misclassification. Due to the static expenses, the total costs cannot go below $\sim$ €2400, when optimizing this subset of transactions. The static expenses equal to 37% of the original costs, which implies that, with 30% already reduced, *another 33% could be reduced*.

In contrast to OIS, the Current Optimal Instance Score (cOIS) range starts at the current situation, when all transactions are send to the refund route. This equals the second situation in Fig. 7-3, when only the least-cost optimization is applied. The cOIS range is shown in Fig. 7-4. For the rest of this chapter, take into account that cOIS ranges from the current to the optimal situation, and only incorporates the prediction domain.

7-3 Experiment 1: Categorical Encoding Methods

In this section the categorical attributes are incorporated into the predictor. This requires data transformation as the categorical data is not supported. In this experiment encoding methods described in Section 2-6 will be compared in context of refund prediction to answer the following question:

RQ6: Which encoding methods work best
to transform categorical attributes into numerical features?

The problem In practice, working with categorical data involves a lot of practical problems. For example the scikit-learn toolkit used in this research is not able to process categorical data. It requires numerical data. Some toolkits are able to process categorical data, for example the randomForest package in R. In these toolkits, calculating the best split for K categories, involves a selection of $2^K - 2$ combinations. Due to this exponential increasing computation time this package is limited to a (hardcoded) maximal amount of 32 categories. the used payment data has categories with over 100 categories.

The scikit-learn package used in this research is not able to process categorical data, other than in ordinal or binary format[1]. It can process boolean, floats and numerical formats. In this experiment the 6 categorical transaction attributes are encoded into numerical features.

Categorical data An overview of the number of categories per categorical attribute is shown in Table 7-2. The last three features have a lot of categories.

The occurrences of the top 9 categories are shown for each feature in Fig. 7-5. The categories in the merchant account are well distributed over the transactions. The first 5 email domains are present in almost 50% of the transactions. The tail of this distribution is huge. For both shopper- and issuer countries we can say that over 75 countries in total occur at only 13% of the transactions.

The 3D-Secure process id not used for all transactions, making the occurrence of dash so enormous. The authentication response is dependent on the directory response, ie. an authentication response is only possible for the Y cases in the directory response.

Feature	Merchant Account	Directory Response	Authentication Response	Email Domain	Shopper Country	Issuer Country
Number of Categories	18	4	5	2946	103	86

Table 7-2: Number of categories in categorical attributes.

[1]See scikit-learn documentation; http://scikit-learn.org/stable/modules/preprocessing.html#encoding-categorical-features

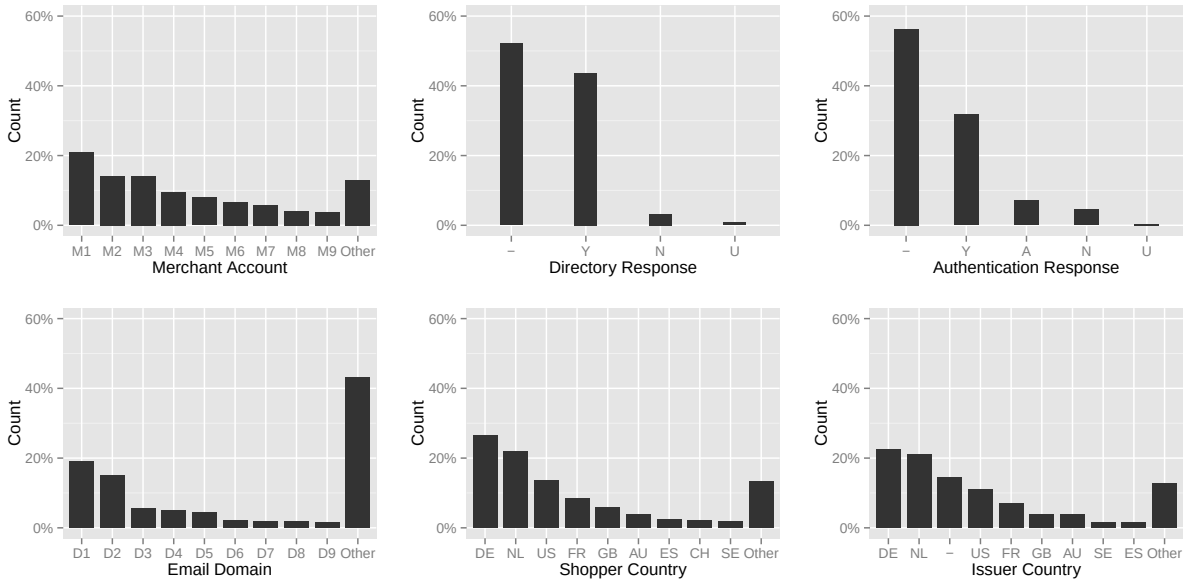


Figure 7-5: Distribution of categories in categorical attributes. If over 10 categories, the first 9 plus 'other' will be showed.

Encoding Methods Section 2-6 described ordinal, binary and scalar approaches, with their variants, to encode categorical data. By definition ordinal features are structured in a certain order, where the distance matters. Think about the domain names or countries, additional statistics need to be extracted to order these. The described categorical features do not easily enable themselves for ordering and therefore ordinal and temperature (- 1) encoding will not be used.

With binary encoding, every category will be given its own feature. Due to the large number of categories in some attributes (over 2000 for email domains), binary (- 1) encoding is expected to increase the computation time exponentially. The other two described scalar encoding methods are implemented for this experiment. The used encoding methods are described in detail in Section 2-6, a short explanation as follows;

- *Nishisato* Substitute category by the probability of the non-refund class for this category. In reality this probability unknown and therefore estimated by observing the known class distribution.
- *Fahrmeir* Substitute category by a scalar value based on the difference between the probability of two classes for this category. These probabilities are also estimated as in Nishisato.

The two scalar encoding methods are applied to the categories and compared against each other and the baseline. The baseline does not incorporate categorical features.

Results The results are listed in Table 7-3. Both encoding methods score significantly higher than the baseline score, without categories. However, the variance generated by Nishisato far more than baseline, which makes the prediction unstable. The average increase of performance

	cOIS		Diff. with Baseline	
	Mean	Sd	Mean diff.	Sd diff.
Baseline	0.34144	0.01345	-	-
Nishisato	0.42299	0.01727	+23.88%	+28.41%
Fahrmeir	0.43145	0.01118	**+26.36%**	**-16.83%**

Table 7-3: Results for categorical features encoding.

for Fahrmeir encoding is 0.09%, which is a lot for adding 6 features. We will inspect features more in the next experiment, Section 7-4.

The scores are visualized in Fig. 7-6. Two things can be concluded from these results;

1. Incorporating the categorical features significantly improve the performance of the classifier.
2. Fahrmeir encoding is preferred because it is more stable and scores a higher performance.

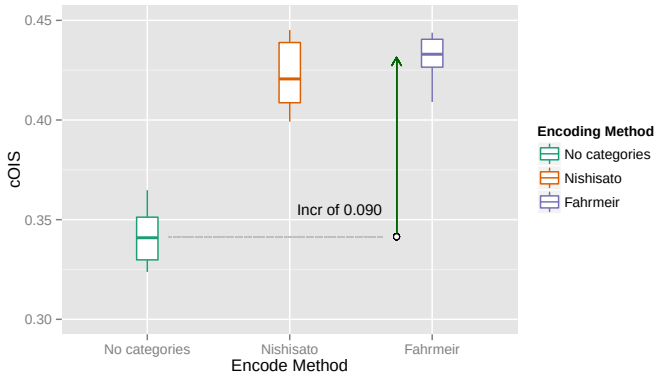


Figure 7-6: Results for incorporating 6 categorical features with different encoding methods.

7-4 Experiment 2: Feature Selection

In this section, we will study the influence of the different feature vectors on the performance. The main objective of this section is to decide on an optimal feature vector. First we will show the main performance of the different general feature sets.

7-4-1 Feature Sets

In section Section 6-2 four different feature sets have been explained. Some sets contain very sparse features, for example the risk checks. For this reason the sets are combined with the baseline features. The following feature vectors, derived from feature sets introduced in Section 6-2, are compared.

- *Baseline (B)* The numerical attributes of the transaction attributes.
- *Baseline + Risk Checks* The baseline features plus the risk checks.
- *Baseline + User History* The baseline features plus the user history.
- *Baseline + Regional Stats* The baseline features plus the features extracted from regional statistics.

For each of these feature sets, the best performing parameters and threshold are estimated on the inner- and applied to the outer training set, resulting in the scores in Table 7-4, visualized in Fig. 7-7.

Both risk checks and user history perform slightly better than the baseline. However, the added regional statistics performs significantly lower. We think this is due to the fact that adding the same information to large groups of samples, lowers the distinctiveness of the individual samples. With these results of set-based performance we can answer the following research question:

RQ7: Which features from fraud detection also apply to refund prediction?

The performance of all the risk features together perform marginally better than the baseline. In Fig. 7-8, later in this section, we will see that none of the individual features significantly influence the refund prediction.

The individual features from different feature sets can be combined into a new set. In the next section an optimal subset of features is selected.

	Selected Parameters			cOIS	
Feature Vector	Max Depth	# Estimators	Threshold	Mean (▽)	Sd
B + User History	1	1000	0.71	0.3885	0.0137
B + Risk Checks	4	2	0.68	0.3439	0.0103
Baseline (B)	1	32	0.71	0.3414	0.0134
B + Regional Stats	20	125	0.81	0.0507	0.0291

Table 7-4: Results per feature vector on outer training set, ordered by the mean cOIS.

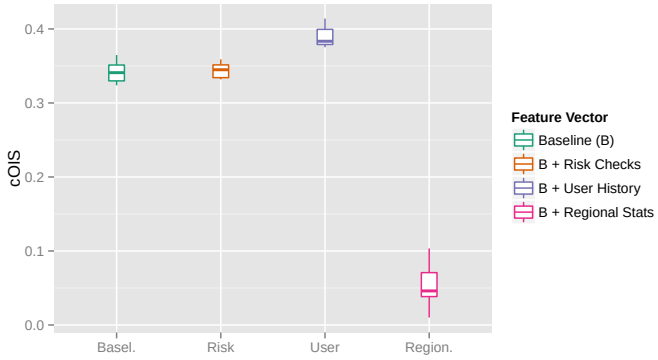


Figure 7-7: Results for different featuresets.

7-4-2 Feature importance

In the previous section the performance of different feature sets are compared against each other. Individual features can differ in terms of informativeness, and therefore in their contribution to the total performance. In addition to the full feature sets, we compose a subset of features which performs optimal.

Tree-based Feature Selection (TFS) TFS uses the feature importance, extracted from a trained classifier, to select the most useful features. In this experiment the feature importance is 'gini importance' due to the use of gini as impurity criterion. It is the total decrease of node impurity averaged over all trees. The decrease of node impurity is weighted by the probability to reach the corresponding node, which is approximated by the number of samples that reach the node[30].

A RF is trained on a combination of the three best performing feature sets from the previous section; *a*) (numerical) transaction attributes, *b*) risk checks, and *c*) user history . The optimal parameters, selected with the inner train set, are 2000 estimators, with a maximal tree depth of 6. The decision threshold is set on 0.67.

The total set of 56 features, consists of 18 attributes, 29 risk checks, and 9 user history features. The gini importance is extracted from the fit with all features. An overview of the scores, summed per feature set is shown in Table 7-5. The Gini importance of all individual

Features	#	Gini imp. (sum)
Transaction attributes	18	0.398
Risk checks	29	0.129
User history	9	0.473

Table 7-5: Overview of summed gini importance for all features per feature set.

Rank	Feature	Gini imp. (▽)	Cum.
1.	Amount	0.188	0.188
2.	Refund ratio (10 months)	0.170	0.358
3.	Refund ratio (3 months)	0.101	0.460
4.	Refund ratio per Amount bins	0.100	0.560
5.	Refund Ratio (1 month)	0.041	0.600
6.	Average Amount (1 month)	0.037	0.637
7.	Number of Txns (1 month)	0.037	0.674
8.	Account risk score	0.032	0.706
9.	Average Amount (3 months)	0.027	0.732
10.	Number of Txns (3 months)	0.023	0.755

Table 7-6: Top 10 features, selected and sorted by their Gini importance.

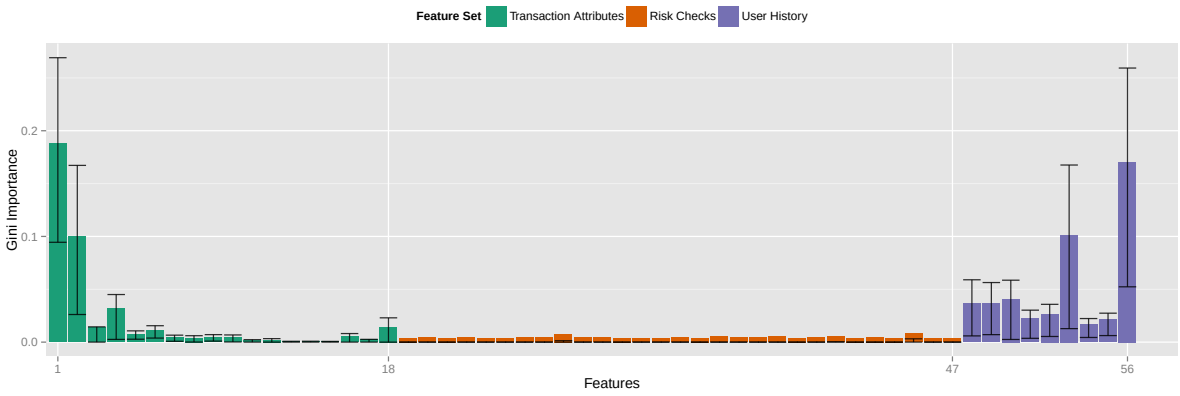


Figure 7-8: Gini importance (mean over all trees) per feature. The error bar shows the 1st and 3rd quartile.

features are shown in Fig. 7-8. The features are colored by their corresponding featureset. None of the risk checks stands out, but still all together their cumulative gini importance is 0.129. This can be due to the type of data. Boolean formatted features are just less informative than numerical features, even if they are with more.

The Top 10 important features are shown in Table 7-6. It is striking to see that **7 out of 10** most important features are originated at the **user history**. The other three features are coming from the transaction attributes. The cumulative scores show that the Top 3 together scores 46%, while the Top 10 scores over 75%, in terms of relative Gini importance.

More Features We have shown that the transaction amount and features regarding the user history are important pieces of information for recognizing refund patterns. A search for more information to strengthen predictor might be a good next step.

RQ5: Which information should have been available for refund prediction?

In this research we show that a refund predictor on transactional data can reduce transaction costs of a merchant. It is in the interest of the merchant to improve such a classifier. The next step is to get data from the merchant representing the shopping behavior of the user at the webshop. This could include all kinds of information from fields of interests to order information, both current and historical.

In addition to different data retrieval we should expand features by generating them with the user's transaction history. According to the results of the feature importance, features generated from user history are the most promising. The long term (ten months) refund ratio of the user is important, as well as the short term (one month) number of transactions and their value. These values might be obvious as a refund can be correlated to historical refund ratio, and historical transaction value is also important seen transaction value as the most important feature.

We have shown which features are important, now can we decide on which features we select and which we eliminate? How much features should be selected, and how does this decision impacts the performance? This is not straight forward, and therefore we use Recursive Feature Elimination to select the best performing subset.

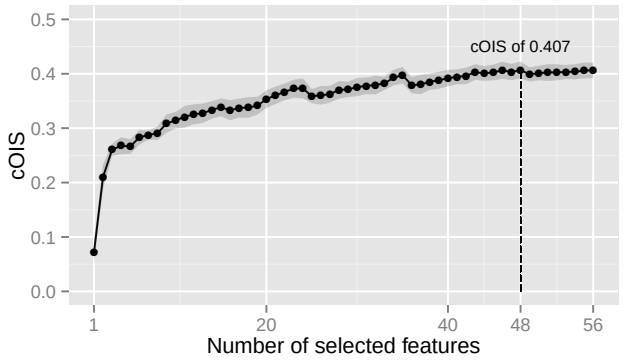


Figure 7-9: Results of RFE, performance in terms of cOIS for different number of recursively selected features. The ribbon represents standard deviation.

Order of elimination	Feature
1st	Month (Sinus)
2nd	Month (Cosinus)
3rd	Month (Raw)
4th	Day of week (Cosinus)
5th	Day of week (Sinus)
6th	Shopper name all lowercase
7th	Risk check 1
8th	Risk check 21

Table 7-7: Eliminated features, first eliminated features listed on top.

Recursive Feature Elimination (RFE) As the name suggests, RFE recursively eliminates features. This method uses the Gini importance of all features. RFE has l rounds for l number of features. The classifier starts using all features, and each round it eliminates the feature with the lowest Gini importance. At each round the classifier is trained with the given features and a cross validation score is calculated. The ideal number of features can be selected comparing the corresponding scores.

The results of RFE are shown in Fig. 7-9. For each number of selected features a cross validation score is shown. The highest cOIS, of 0.407, is scored with 48 selected features.

8 features have been 'eliminated' using RFE. The eliminated features are shown in Table 7-7. All the month related features are eliminated first. This makes sense because the training data only contains two months of data. However this might change when the predictor is trained with data spread over multiple months or years.

The day of the week is of no interest when talking about finding refund patterns. It scored so bad that it was eliminated from the predictor. Apparently there are no pattern between orders in- or around the weekend, or business days with sending in refunds.

7-5 Experiment 3: Sample Weight Strategy

As research question we asked the following general and broad question. In this section we will incorporate a strategy that improves the performance significantly.

RQ9: How can we improve the performance of the classifier in context of routing decisions?

This section describes a strategy to weight individual samples during training. The main objective is to show the influence of this weighting strategy. First we will shortly describe the origin and contents of the strategy.

In the research approach section we introduced a modular approach to solve the problem, consisting of *a*) the prediction, and *b*) the cost minimization logic. The logic is incorporated into the created evaluation method (cOIS) to measure and optimize the full system performance. In this experiment, we show that incorporating costs into the training of the classifier significantly improves the overall performance, which is as expected.

To incorporate the costs we explore the differences in costs between route alternatives. These cost differences (for individual transactions) are shown in Fig. 5-1, introduced in Section 5-3, where optimizing costs as part of the performance is described. The figure (cost differences) show the cost reduction chances for different transaction amounts. The opportunity to optimize cost is high for high-valued non-refund transactions, and the other way around for refunds.

7-5-1 Strategy

A high diff transaction (one with a high cost difference between its route fees) can be seen as a high risk transaction. These transactions have the highest misclassification costs. Therefore these transactions should be more important in the model. We use a *sample weight strategy* where transactions are weighted according their cost difference between their route fees.

The refund and non refund transactions both have other costs differences due to their different misclassification costs. The density of the cost difference are shown per class in Fig. 7-10.

The general distribution of transactions over the transaction value is shown in Fig. 7-2. The peak in low-valued transactions cause the large amount of low-valued non-refund transactions in Fig. 7-10. However, misclassification risks for refund transactions are higher for low-valued transactions, and therefore the refund transactions are more distributed around higher cost differences.

7-5-2 Results

This experiment will show the difference in incorporating the defined sample weight strategy. The samples are weighted according to the cost difference between the its individual fees of the alternative routes.

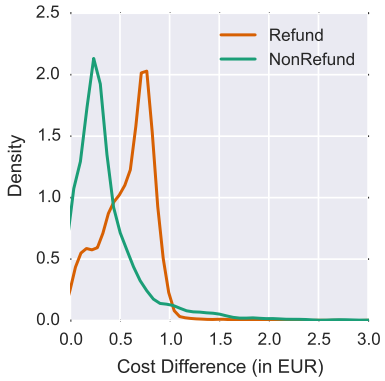


Figure 7-10: Density function of cost difference between alternative routes, per class, for the prediction domain. The density function continues till around €12.

	cOIS		Diff. with Baseline	
	Mean	Sd	Mean diff.	Sd diff.
Baseline	0.34144	0.01345	-	-
Sample Weight Strategy	0.44888	0.01876	**+31.47%**	**+39.48%**

Table 7-8: Results for sample weight strategy.

The optimal selected parameters for RF is a rather complex one. 2000 trees with depth of 10 are selected. The results are shown in Table 7-8. The cross-validated score with the testdata included is significantly bigger (0.107) than baseline score, this is shown in Fig. 7-11.

The complex selected parameters seem to be more sensitive for overfitting, dispite of the cross-validated parameter search. The standard deviation is increasing by 0.0053 (39% of baseline sd) in comparison to the cross-validated baseline scores. This shows the model is slightly more over-fitted than the simple baseline model. A reasonable explanation for this is that the model is over-fitted to the high-cost-diff samples, which are weighted higher during the training.

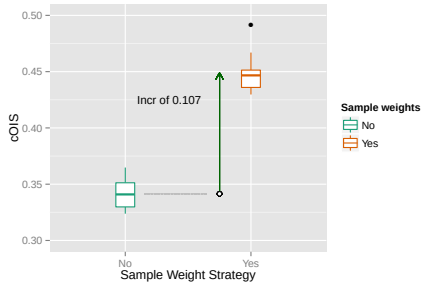


Figure 7-11: Results of 10F-CV split by decision to incorporate sample weights.

7-6 Best Predictor

In this section a final model will be designed, which is tested against the baseline to see the total gain in performance. The final predictor is validated against unseen data and the expected costs reduction is calculated.

First the final design of the predictor will be explained. Second results will be presented for the predictor's cross-validation performance on the train set. At last the predictor is validated against the unseen data, which has been set aside at the start of this research.

7-6-1 Design

The main part of the design is established by the previous experiments. The conclusions of the experiments indicate how the ideal predictor should be designed. The main conclusions are shortly summarized;

- The **Fahrmeir (numerical) encoding** applied to all categorical transaction attributes contribute significantly to cOIS performance. The performance is 0.09 better than baseline of 0.34 (over 26% increase), while decreasing the standard deviation with 0.002 (almost 17% decrease).
- Experiments incorporating different subsets of retrieved and extracted features showed that features from regional data should not be included. RFE on the other three sets eliminated 8 invaluable features, mostly date and risk related. The **selected 48 features** beats the baseline cOIS performance with a score of 0.407.
- The proposed **sample weight strategy**, where cost differences are emphasized in the learner, contributes to cOIS performance. Compared to baseline the performance is almost 0.11 better (over 31% increase), however the standard deviation also increases by 0.005 (over 39% increase). The system is slightly over-fitted.

The conclusions from previous experiments are combined and elaborated into the final model of the best predictor. This classifier is trained on the inner data set to find the best performing parameters in terms of cOIS.

The best performing parameters chosen to elaborate in the final model are 1000 number of trees with a maximal depth of 8 splits. These parameters create a more complex model compared to the simplistic baseline parameters, a reason for this is that the number of features is almost three times as much. The detailed results, which can be found in Appendix C-4, show that these parameters are the optimal choice.

Results

The final model is constructed from selected features by RFE, Fahmeir encoded categories, and instance weighting is applied during training. The performance of this model are measured by applying a double cross validation, to first select the optimal parameters. Details of this selection can be found in Appendix C-4. The selected parameters for the final model are 1000

	cOIS		Diff. with Baseline	
	Mean	Sd	Mean diff.	Sd diff.
Baseline	0.34144	0.01345	-	-
Best Predictor	0.49306	0.01107	**+44.41%**	**-17.69%**

Table 7-9: Results for best predictor on train set.

trees with a max depth of 8 and a decision threshold of 0.51, nicely in the middle. This model is applied to the outer training set and the cross-validation score is shown in Table 7-9.

The best predictor results in an cOIS of 0.493, an increase of 44% compared to the baseline result. Simultaneously the standard deviation is decreased by almost 18%. This is a nice improvement and takes the result to almost 0.5, halfway to the optimal situation. This score shows that the classifier is able to create a profitable situation, but certainly has a big way to go. The next step is to validate the designed and tested predictor against unforeseen data.

Validation

The last test is to apply the final model to unforeseen data, the validation dataset. The predictor is trained on all train data and applied to the validation dataset. The final model is validated with a score of 0.48721 cOIS performance. This is just a little bit lower than the results on the outer train set, but satisfying.

Cost Reduction

In Section 7-2-3, where the baseline is established, the cost reduction was 30%. The previous sections have improved the classifier its performance from cOIS of 0.341 to 0.493. With a classifier performance of 0.493 the total costs are reduced with 36%. The improvement of the classifier improved the cost reduction with an extra 6% cost reduction from the 33% that we were able to reduce. Theoretically there is still 27% left to optimize by improving the refund classifier.

To show the impact in terms of costs we calculate the savings for the sample merchant. This merchant processes around 22k transactions a month. Assuming that every transaction can be routed, the merchant can save just over €4K a month. This makes his yearly cost reduction almost €50K.

To give some context to this amount we show the relative cost savings over the merchant profits. The 22k transactions in total process 4 million euro. Of this amount 26.8% is refunded and therefore this merchant has a revenue of almost 3 mln euro per month what corresponds to 35.6 mln euro per year. The €50K cost reduction corresponds to 0.14% of their total revenue. Assuming the retail merchant has a margin of around 10% the reduction corresponds to 1.4% of their profits.

7-7 Conclusion

In this chapter a baseline is established for refund prediction. It is shown that refund prediction scores 0.731 AUC. Where a random predictor scores 0.5 this predictor scores good enough to be useful for transaction routing. With this baseline classifier we are able to reduce costs by 30%.

In three different experiments we have shown that fahrmeir encoding, applied RFE and the instance weight strategy all three contribute to the performance of the classifier, see Table 7-10. The results of the three experiments are combined in a final model which scores 0.493 cOIS, 44% better than the baseline score.

The final model is able to reduce transaction costs with 36%, which can be almost €50K for a merchant processing 22k transactions a month. The cost reductions represent around 1.4% of the merchant its profits.

		cOIS performance	
Experiment	Outcome	Mean	Sd
Baseline		**0.34144**	0.01345
Encoding	Fahrmeir	0.43145	0.01118
Features	Transaction Attributes, Risk Checks and User History, and eliminated 8 features	0.40692	0.01552
Instance Weights	Use the strategy	0.44888	0.01876
Combined	*All preceding*	**0.49306**	0.01107
Validation	*All preceding*	**0.48721**	*n/a*

Table 7-10: Overview of experiments, their decisions and results, and the final combined outcome and validation.

Chapter 8

Conclusions and Future Work

8-1 Introduction

In this thesis we have successfully designed system that is capable of optimizing transaction routing leveraging refund predictors trained on transactional data. We have designed a classifier and evaluation method specifically for transaction routing. With these techniques a retail merchant can reduce 36% of its transaction costs what equals to 1.4% of their profits.

This chapter summarizes the contributions and conclusions of this research. Before we move on to the conclusions we shortly summarize the problem and answer the defined research questions from Chapter 3.

8-2 Research Questions

Transaction routing is not a new concept. In practice routing is applied to route transactions to acquirers with different strategies. However, routing transactions to different payment schemes is not yet shown in both practice and literature. The first step is to understand how to apply transaction routing between schemes.

8-2-1 Transaction Routing

The differences between payment schemes make routing between them challenging. One of these changes is support for refunds and defines the problem of this research. A classifier should then be responsible for predicting if a transaction will result in a refund or not.

RQ1: How does the intervention of a classifier influence optimization of transaction routing?

The transaction fees are dependent on individual transactions. The construction of the different scheme fees splits the number of transactions into two groups, one group where the prediction is necessary and one which is routed without prediction. The latter group can be routed because the difference in costs between the routes is higher than the potential error. The transactions in the first group are routed according their prediction value.

RQ2: What is the minimal classifier performance to enable a profitable situation for transaction routing?

The performance of the classifier reflects how many and what type of errors it makes during prediction. We have introduced Current Optimal Instance Score (cOIS), an evaluation method where a negative score equals a non-profitable situation and a positive score a profitable situation. The maximum score is 1 when all predictions are correct and transaction routing is optimized. The idea of using a classifier to route transactions is established, now how do we predict refunds?

8-2-2 Refund Prediction

Just like routing between schemes, refund prediction is not explicitly researched in literature. Therefore we need to find out if refund prediction is possible using transactional data.

RQ3: Is refund prediction possible with transactional data, and how does it perform in general?

We have shown that refund prediction is possible. With a basic classifier a basic score of 0.731 Area Under ROC (AUC) is reached. The score is better than random (0.5) and good enough to open up possibilities for transaction routing.

Evaluation Before we research the prediction of refunds, we need to select a standard to measure the performance which represents the important characteristics specific to this situation.

RQ10: Which existing evaluation methods measure performance considering the refund specific characteristics?

We established criteria specifically for this routing situation. These criteria have been used to select an evaluation method that fits this situation best. We have shown that none of the evaluation methods under study support all presented criteria. Evaluation methods which are common practice in classification research such as AUC and Average Precision (AP) are shown not useful for individual transaction costs. These methods evaluate the on average performance of the classifier and do not evaluate a practical setting where a decision threshold is chosen. Measuring overall performance is not the right strategy to reach the optimum. Therefore additional research must define how we should measure performance.

RQ11: How would we ideally measure the classifier performance which expresses the winnings in relation to payment routing?

We introduced a new evaluation method which is able to measure the performance of the classifier in terms of transaction routing. The realistic loss function is extended to support dynamic individual misclassification costs. The classifiers power is expressed in terms of minimizing the individual transaction costs.

The evaluation method comes in two flavors; Optimum Instance Score (OIS) and Current Optimal Instance Score (cOIS). Both methods measure the performance of the classifier in terms of minimizing the sum of all individual transaction costs, but change in how to interpret. OIS is more theoretical where the scale reaches from the worst to the best prediction. cOIS is slightly different where the scale is a subset of the OIS scale, reaching from the current situation to the optimum where the current situation equals routing all transactions to the scheme that supports refunds.

In comparison to AUC or AP, cOIS does not measure overall classifier performance, but measures performance of the classifier in a specific practical situation in terms of total costs minimization. This change the way the classifier can be improved. Instead of focusing on improving the number of correctly identified instances we focus on improving the total value created by the classifier. OIS and cOIS allows the classifier to be tweaked for correctly predicting the most important instances.

With the introduced evaluation method cOIS the details for the classifier can be designed.

Data Retrieval Information should be retrieved to improve the identification of refund patterns. The prediction is necessary at the time when the routing takes place, this is before the incoming transaction is sent to any other party.

RQ4: What information can be retrieved at the time of the refund prediction?

We have identified all data fields that are available at the time of prediction. due to this timing constraint we have dropped information During this process that might impact the performance. This information however can be used when the refund prediction is used at another time in the process for example after authorization of the transaction.

The basic information that arrives with a transaction when it enters the system can be used for classification. This also includes all risk data generated by internal services. Knowing these limitations caused some thinking about data that might or should be there at the moment.

RQ5: Which information should have been available for refund prediction?

Research to the available information shown that information about the user and its history are very informative for predicting refunds. Additional user information is essential to improve the performance of the classifier. Examples of this information can be about the cart, products, lifetime or history and might be retrieved in the future from the merchant.

Features Of the available information some transactional attributes were formatted as categorical values, like the country (Netherlands, England, France etc). We needed to research how we could incorporate these values the best.

RQ6: Which encoding methods work best to transform categorical attributes into numerical features?

To answer this question we searched for literature about categorical encoding. Not a lot of encoding methods were available and some were dating from the 80's. However we tested the usable ones and showed a significant improvement on the prediction.

Risk data could also be retrieved from a transaction. With this data we could indicate how valuable these features are in context of refund prediction.

RQ7: Which features from fraud detection also apply to refund prediction?

None of the available risk checks individually contributed a lot to the refund prediction. However, the whole set of features did marginally contribute to refund classification.

Classification Algorithms Given all this information, which algorithm can recognize refund patterns best? This research is applied in a different context, i.e. optimizing transaction routing context, and therefore different algorithms need to be tested and compared.

RQ8: Which classification algorithms perform best for refund classification in context of routing decisions?

To answer this question three commonly good performing classifiers from fraud research are tested for refund classification. We have shown that Random Forest (RF) is the best performing and most stable of the three. Using this classifier we search for ways to improve the performance of the classifier.

RQ9: How can we improve the performance of the classifier in context of routing decisions?

To improve the performance of the classifier we have performed three independent experiments.

1. First we have tested the best performing encoder for categorical data. We have selected multiple different encoders from literature and tested the applicable encoders against the baseline.
2. Second we identified the good and bad performing features, and eliminated the bad ones. We have shown that the amount of a transaction is in terms of Gini importance responsible for almost 20% of all features. The historical refund ratio over different timespans comes second and the top 5 important features are responsible for 60% of the relative feature importance.

3. Third we constructed a sample weighting strategy to weight individual transactions in the training phase of the classifier. The introduced evaluation method allows us to tweak the classifier in such a way that it emphasizes the valuable instances. This strategy elaborates this concept.

These three improvements significantly improve the classifier performance. With the knowledge and answers from all sub questions for each aspect we are able to answer the main research question;

Main RQ: How can we use classification methods to predict refund behavior per transaction in order to optimize payment routing?

We designed a modular approach to use a predictor and simple route optimization logic. It is shown that existing evaluation methods are not optimal to measure the performance of the classifier in this context. We constructed an evaluation method for the classifier which is capable of expressing performance gains in terms of costs associated to optimizing transaction routing.

The final classification model is validated and applied to show the cost reduction for one particular merchant. This merchant processes 22k transactions a month and can reduce 36% of transaction costs on all routable transactions, saving almost €50K per year. This cost savings is equal to 1.4% of its profits under the assumption that the merchant makes 10% margin over its revenue.

8-3 Scientific Contribution

1. We performed an evaluation of evaluation methods with established criteria in routing context. We have selected various evaluation methods from literature including AUC and AP which are common practice in classification. With theoretical examples we showed that none of the evaluated methods is capable of dealing with the individual transactions costs.

2. In addition, we have introduced and validated an evaluation method that is capable of showing the performance in terms of transaction route costs. Two representations of the evaluation method are introduced, *a*) Optimum Instance Score (OIS), to measure between worst and optimal routing situation, and *b*) Current Optimal Instance Score (cOIS), to measure between the current practical routing and the theoretical optimum.

   cOIS changes the way we can improve the classifier. Instead of improving the overall performance such as the number of correct predictions, we are now able to improve the number of valuable predictions.

3. This exploratory research sets a baseline for more research in refund classification. The baseline performs 0.34 on a cOIS scale. This shows that even a simple predictor can reduce costs. Specific contributions to prediction in routing context are;

- We compared different encoding methods from literature, and showed that different scaling approaches perform equally and significantly better than without the encoding methods. Farhmeir encoding is shown to perform the best with an improvement of 26% compared to the baseline cOIS. Due to the high amount of categories in categorical features for the payment data, scaled encoding is shown a successful replacement.
- We studied the importance of individual features for refund classification. The transaction value and user history together are the most important features of refund prediction. The refund ratio from user history exists in 3 different timespans; the ratio of the last month, the last three months and the last ten months. These features together with the transaction amount are responsible for almost 60% of the relative feature importance.
- We showed that adding global information (not instance specific, but spanning a big group) ends up in worse performance (see regional statistics)
- We introduced a sample weighting strategy that weights samples according to their importance in the transaction cost optimization. The samples are weighted with the costs differences between the alternative routes. We have shown that the proposed strategy has a significant positive impact on the classifier and thus route performance.

8-4 Industrial Contribution

1. The industrial usage and strategies of transaction routing is described, including the limitations of current used acquirer routing. The concept of transaction routing between schemes by leveraging dual branded cards is identified and described.

2. In this thesis a new approach to optimize payment routes is proposed. A modular approach is designed to route transactions with additional knowledge about transactions. This additional knowledge is the prediction if the transaction results in a refund. Refunds are only one specific feature (or property) of a route. Payment network routes have lots more properties, which all possibly can be predicted. This shows that this exploratory research took transaction route optimization one step further leveraging machine learning tools.

3. We have successfully shown that refund prediction is possible with transactional data. Other applications could benefit of this new refund prediction. Merchants can advise or inform a shopper once it knows the probability of a refund. Think about if the predictor will be better, you can even refuse transactions highly likable to be a refund or apply targeted marketing.

4. We successfully optimized route costs in payment network. A best classifier is built by assembling the best performing parts or configuration selected from literature and tested in this thesis. With the best classifier the optimization results in a total cost reduction of 36%. The merchant studied in this thesis is saving almost €50K per year in paying less transaction fees. The cost savings equals to 1.4% of the merchant its profits under the assumption that the merchant makes 10% margin over its revenue.

8-5 Limitations

- Only one merchant is used for the experiments. We have shown that a merchant with 26.8% refund ratio can save 36% on transaction costs. The results may differ for other merchants with various refund rates. For lower refund rates the cost reduction is expected to be higher because the predictor can send more valuable transactions to the cheaper route.
- Two fixed cost functions are used in this research. Both are real cost functions used in the industry. These are specific to the Belgium market, the domain of this research. The nature of these functions, one static and one linear, allow for transaction routing. Other cost functions will change the route logic and influence the results.
- We assumed that a manual refund can be done and costs €1,- per transaction. This is done by calling the shopper, asking his/her details and transfer the money. There might be other possibilities or costs related to the manual refunds. The manual refunds costs is directly related to the results and a different manual refund costs implies different results. Higher costs will result in less reduction because more transactions are dependent on the prediction, which is sensitive for classification errors. In contrast, lower costs will improve the cost reduction because more transactions can be routed to the cheaper scheme without using the refund prediction.
- This thesis focused on minimizing the costs. However there are more objectives which change the optimal situation. Take for example the conversion rate of the routes for a scheme. These differ per scheme and might differ for individual transactions or groups of transactions.

8-6 Future Research

- One of the most interesting ideas that rose during this thesis is to adapt the loss function of the classifier. The scale of OIS, based on cost differences, suggests to change or optimize the loss function in an estimator. Do not separate the classes just on information gain, but on value gain. The evaluation method created during this thesis is essential to measure the usefulness and performance gain of such an approach.
- Instead of designing a completely new loss method a short research to existing loss functions can be also useful to understand their influence of the loss functions.
- A more general way to improve the classification in this thesis is to encorporate online learning techniques, in contrast to batch learning. Transactions come in sequentially, meaning the learning algorithms can adapt once a new transaction enters the system.
- The introduced evaluation methods OIS and possibly cOIS can also be applied to classification problems from other fields. The first method can be used when the misclassification costs of individual instances differ in a supervised learning problem. The second method can be used when the total costs of the classification can be related to current costs in practice. We will list a few fields where OIS could be applied;

- *Fraud Detection* Imagine an airline company where flights are booked in busy and quiet times. A classification system tries to detect and stop fraudulent tickets. If it accepts a fraudulent ticket for a quiet plane, one seat will be reserved and only costs some kerosine (plane would have empty seats anyhow). If it accepts a fraudulent ticket in a busy plane, the seat could have been bought by another customer and possibly costs hundreds of euros. The proposed evaluation method can deal with these individual costs in fraud detection.
- *Computer Vision* Marine ships are equipped these days with highly complex radar systems. Some of these systems are built to recognize other ships or objects. The classification of different types of ships are followed by different consequences. Examples of different ships are allied or enemy ships, but also ships for fishing or transport can be detected. It is obvious that the costs of misclassification differ in these situations.

Additional research is necessary to show the usefulness of OIS in fraud detection, computer vision, and possible other fields.

8-7 Remarks

It was definitely challenging to explore the possibilities of refund prediction with transactional data. The overall performance of the predictor was not too good, while the resulting optimization is promising. But the results are much broader then just refund prediction.

During this thesis we defined a solid and modular approach to predict and route according to this prediction. This approach can be used for more than just predicting refunds, optimizing costs or routing between schemes. All these parts can be exchanged with other parts, making this approach useful for other purposes.

We are sure that the results and approach of this thesis contribute to a better understanding of the possibilities, further research to this topic, and incorporation into the live environment of the system.

Appendix A

The Payment Process

The payment process - from the beginning of a shopper's buying initiative till the fund transfer between the involving banks - is quite a large and complex chain to describe. In this chapter the payment process is explained in more detail. To start explaining the payment process we first need an understanding of the stakeholders and an high level overview of the relations between those. Second the payment process is described from the shopper's perspective, and third from a system's perspective. At the end of this section refunds are described.

A-1 Stakeholder Overview

In this section different stakeholders will be described by presenting an high-level overview of the stakeholders and the relations between them. First the high-level overview will be presented, where the overview is followed by a description for each stakeholder.

The overview can be found in Fig. A-1. The stakeholders can be best described by explaining a scenario where a shopper buys a product from a Merchant via its website. The payment of the goods can be done via either a Payment Service Provider (PSP) or an acquirer directly, in this scenario only one acquirer is involved. The shopper initiates a payment session after he added goods to his shopping cart. When the shopper confirms its shopping cart at the website the Merchant will notify his acquiring bank of this payment. The acquiring bank will collect the money by connecting to the issuing bank (which is the shopper's bank). The connection between the issuing and acquiring bank is done using standardized payment schemes.

We will quickly describe the different stakeholders in the payment process and their relation to each other.

Shopper A person which buys goods (on- and offline) from a company (merchant) is called a shopper. Shoppers can be either consumers or other companies. The shopper is the customer of the shop concerned, however customer is preferably not used in this document because there are more entities involved which can have customers.

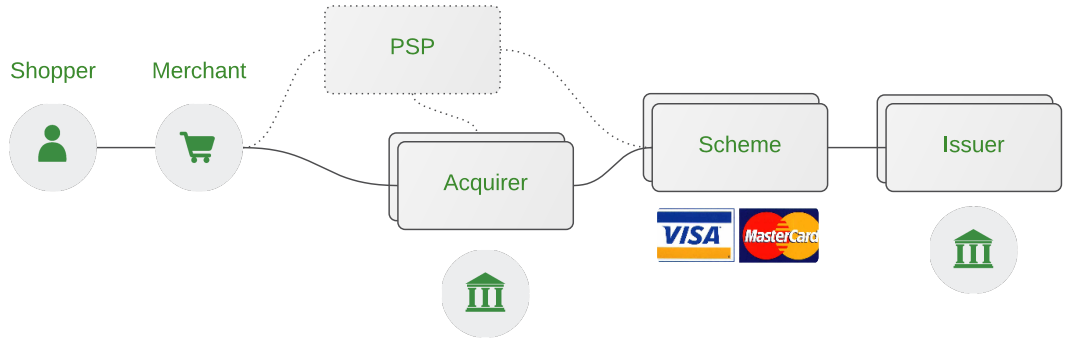


Figure A-1: Overview of Stakeholders (and their relations) in Payment Process

Merchant A person or company which sells products or goods to customers. In this report the customers of a merchant are called shoppers. The merchant is connected to a PSP or acquirer to process transactions from his (web)shop.

Issuer The bank (or financial institution) which provides a payment card (or other payment goods) to a shopper is called an issuer (or issuing bank). In case of a payment the issuer is the bank where the money is collected from after a payment initiative.

Acquirer The bank (or financial institution) which processes the payments of a merchant is called an acquirer (or acquiring bank). In case of a payment the acquirer captures and processes the transaction. The acquirer will collect the money via the scheme from the Issuer.

Scheme A card scheme describes the rules of the network, which enables (standardized) communication between an acquirer and issuer. Examples of card schemes are VISA and MasterCard (also the owners of the 'card scheme').

Payment Service Provider Offers various payment related services for (online) shops. The main service is an easy gateway connection for a Merchant to different payment methods. An example of another service is risk management (including fraud prevention). A PSP contains multiple connections to acquirers and payment schemes and by maintaining this connections and relations they take away complexity from Merchants.

A-2 Shopper's Perspective

In this section the payment process is described from the perspective of a shopper. This perspective is easily understandable and serves as bridge from buying activities by consumers to the complex payment process. The Shopper's perspective will only involve the issuer and merchant from Fig. A-1 and is explained by describing the process step by step. The buying process of a shopper at a shop from a merchant is visualized in Fig. A-2.

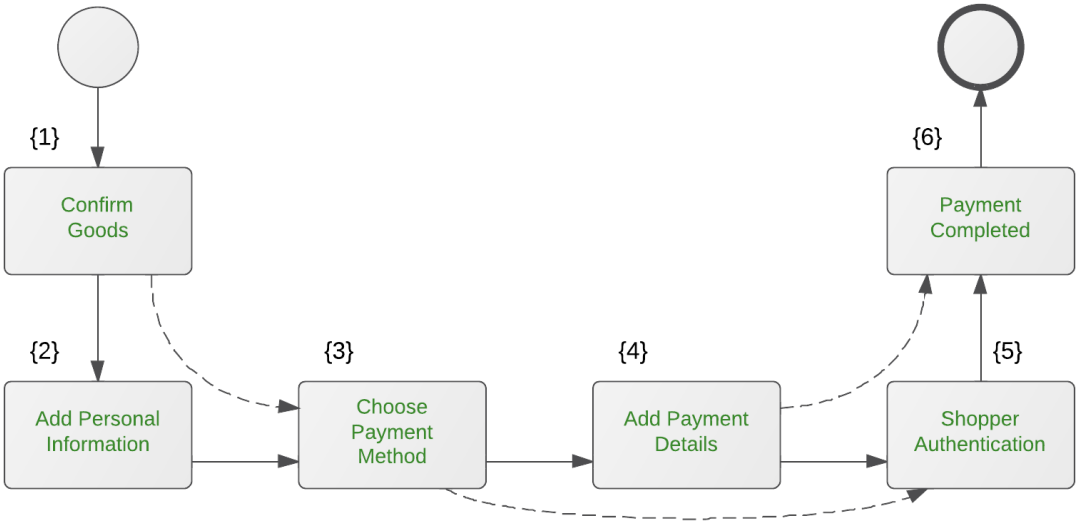


Figure A-2: Online payment process from the perspective of a shopper

1. The process is initiated when a shopper enters an online shop and wants to buy (a selection of) goods. The shopper confirms the goods by confirming his shopping cart and proceeding to the payment of the goods.

2. A new shopper has to fill in his personal details where a returning shopper can skip this step or has to log in at the shop.

3. In this step the available payment methods (defined by the merchant) are presented to the Shopper. The shopper decides which payment method he wants to use.

4. Additional payment details involving the chosen payment method can be entered in this step. This step is not required for returning shoppers.

5. The authentication of chosen payment method. For example entering personal card-holder password related to the card information given in step 4. This step is not always required, as explained in Appendix B.

6. When the authentication is completed the transaction and order is completed, at least for the shopper's belief. A split second before finishing the transaction it was send via the acquirer and scheme to the issuer to authorize the payment.

Note that the aforementioned process only describes shopping at an online platform. The differences for offline payments is that step 1 is done implicitly when bringing your goods to the cash desk and step 2 is skipped. Step 3 involves non-cash payment options via a Point of sale (POS) device. A card or mobile is scanned by a POS device in step 4, followed by PIN (Personal Identification Number) in step 5a.

A-3 System's Perspective

In this section the Payment Process is described from a technical perspective. This section focuses on the part of the system where the payment is processed in real-time while the shopper is paying at the Merchant's shop. The parts of the system are visualized in Figure A-3.

1. The shopper interacts with the HPP or via the merchant's system to the PAL.

2. Hosted Payment Pages (HPP) are the webpages where the shopper can choose the payment method and fill in additional payment information. The HPP is the user interface to the PAL. HPP is the simplest way of integrating the payment into a webshop. The webshop redirects the shopper (with an internal reference) to the HPP and when the payment is completed, the shopper will be redirected back to the merchant's webshop.

3. The Payment Acceptance Layer (PAL) is the API (Application Programming Interface) for processing real-time payments. The Merchant can connect his own system to the PAL to process the payments in its own environment. The PAL communicates to internal en external components and acts as the controller of the real-time process.

4. The recharge service is called by the PAL to retrieve information about recurring payments.

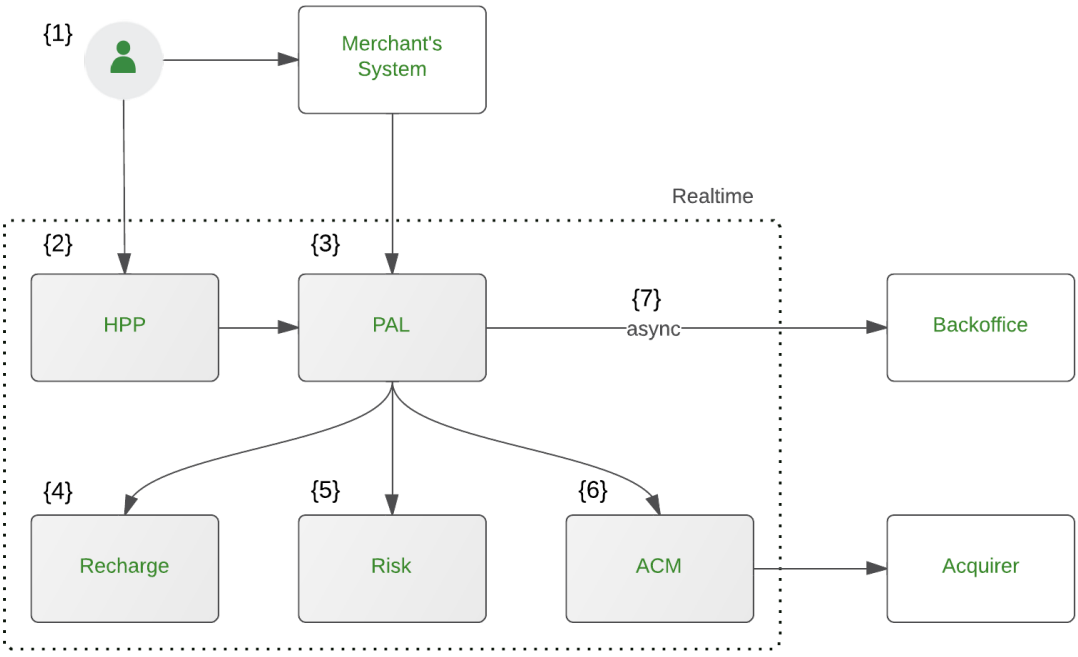


Figure A-3: Real-time system parts in Payment Process

5. The Risk service classifies the level of risk for a transaction. The risk service tries to identify fraudulent transactions while minimizing classifying good transactions as high-risk to avoid refusal of unnecessary transactions.
6. When a transaction is valid it is send to the involved acquirer through the Acquirer Connection Module (ACM). The acquirer collects the money and completes the payment, this part is out of the scope of this section.
7. All the processed transactions are send in batches to the Back-Office (internal booking and reporting software). The back-office tracks the payments and generates all kinds of reports.

Some system parts can be directly related to the Shopper's Process from Section A-2. The relation is shown in Figure A-4.

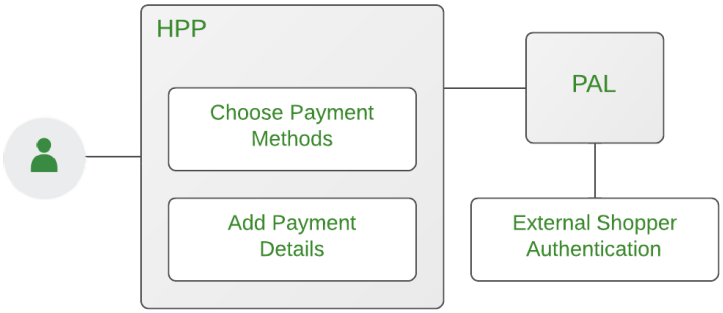


Figure A-4: Relation between shopper process and system components

The 'choose payment method' and 'add payment details' phases from the shopper's perspective (steps 3 and 4) are processed by the HPP or merchant's system. The merchant system is not shown in the figure to keep it simple. The HPP contacts the PAL which initiates the shopper authentication process if necessary (step 5 from shopper's perspective). This authentication step could be mandatory or optional depending on the payment method and shopper information. The authentication process is executed by the issuer to guarantee that the shopper corresponds with the given cardholder information. When the authentication is skipped or completed, the user will continue at the merchant's website to finalize the buying process.

A-4 Refund Process

When a shopper orders some goods on the internet he or she generally has some time to rethink its decision and has the ability to return it. The shopper then returns the goods and the money is returned to the shopper. But wait, how does this money ends up back at the shopper? This fund transfer is called a refund and could be supported by a payment scheme.

The technical process of a refund can be best explained by visualizing the possible states of a transaction. When a transaction is received by the system it has the received state and ends

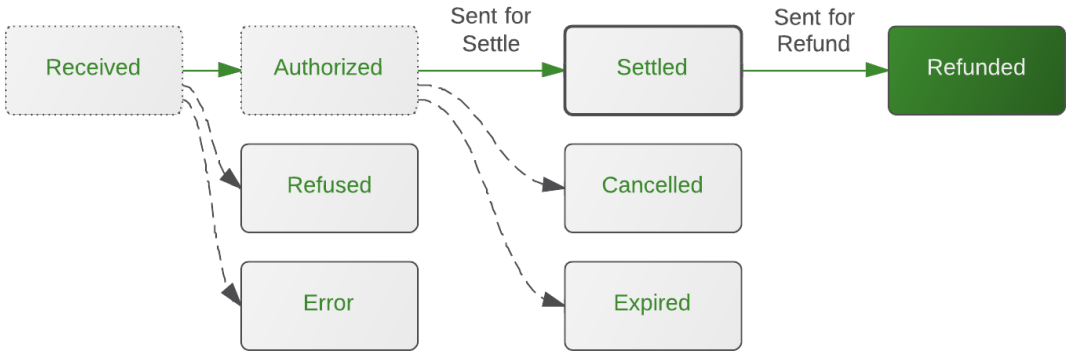


Figure A-5: Possible transaction states. Green path is the path to a refund. Dotted lined states are not end states.

up in another state which is shown in Fig. A-5. The received and authorized state are no end-states and therefore marked with dotted lines. The green path is the path to the refund state, where the other arrows show other possible paths between the states.

After the transaction is authorized by the issuer and thus is completed from the perspective of the shopper, the funds transfer still need to happen. Settlement is the process of showing the transaction to the issuer and collecting the funds. In practice a transaction is settled in a few days, but this can differ per scheme.

A similar structure exists to refund a transaction. First the refund is requested at the financial institution. When the funds are transferred, the transaction is refunded. The 'send for refund' status can be requested by a merchant if the used payment scheme of the transaction supports refunding transactions. The merchant will for example request a refund after a customer have send back his goods, this might take up to 30 days before a refund is requested.

Appendix B

3D Secure

B-1 Introduction

3D Secure (3Ds) is a protocol first developed by VISA under the name of 'verified by VISA' to prevent fraudsters from using cards from legitimate shoppers to process transactions. The idea is to authenticate the cardholder at their own bank and thus moving the liability of the transactions from the merchant to the issuing bank.

This section will shortly elaborate on 3Ds. First the technical process is explained with the messages involved between different parties. Second the most important limitations will be discussed.

B-2 Process

3Ds protocol facilitate authentication of the shopper at the issuing bank. This is implemented by redirecting the shopper to a webpage or application owned by the issuer bank. At this page the shopper enters a secret PIN or password, and the issuer authenticates the shopper in real-time. After authentication the shopper is redirected back to the webshop and can finish his/her transaction.

This process involves two steps;

1. **Verification Enrollment** Is the user enabled for 3D-Secure? During this step the merchant will ask the issuer if the user is enabled for 3Ds. A user is enabled for 3Ds if it has provided the bank its secret, required for authentication. In general, the user will be asked to setup its 3Ds secret the first time it is used, and before this is setup is completed the issuer will take liability.

2. **Payer Authentication** Is the shopper authenticated? If the user is enabled for 3Ds, the merchant asks the issuer to authenticate the user with his known secret.

The messages between the merchant and issuer bank retrieve some information about the status of the protocol. The issuer can respond the first and second question respectively with the following possible responses; Y, N and U and Y, N, U and A. These values are explained in Table B-1. Note that the messages of the 3Ds protocol uses more variables which are out of the scope for this research.

Message	Response	Description
Verification Enrollment	Y	Authentication is available
	N	Cardholder is not (yet) participating
	U	Unable to authenticate due to technical or business reasons.
Payer Authentication	Y	Authentication was successful
	N	Authentication failed
	A	Attempts to authenticate are performed. With this status the issuer is in general liable, but this can differ per issuer.
	U	Unable to authenticate due to technical or business reasons.
Both	-	This is an internal status code of the industrial partner of this research. This status is representing an internal technical problem.

Table B-1: Description of 3Ds status responses per type of message.

The 3Ds process is executed just after the transaction is initiated by the merchant. The raw responses of 3Ds messages contain information about the shopper, and therefore can be used as input data for a payment related predictor and due to the early timing of the process even for transaction routing.

B-3 Limitations

At the introduction of 3Ds big card schemes thought they had invented the perfect tool to reduce fraud for online card transactions. However it soon became clear that there were serious limitations of the protocol which caused businesses to decide on not using 3Ds.

Usage of the 3Ds protocol has several limitations. This section does not intend to create an extensive list of limitations. The most important ones in context of this thesis are described in this section.

B-3-1 Limited to single e-commerce transactions

3Ds is designed only for one time e-commerce transactions. The protocol did not solve this issue for transactions other than e-commerce. Examples of other type of transactions are recurring or CVC-only transactions. With *recurring transactions* the shopper enters his payment details, authenticates and agrees to a subscription model in which he/she is billed

after a fixed recurring period. *CVC-only transactions* are follow-up transactions of an existing user in a webshop. The first time the shopper enters its details when the merchant stores the card information without CVC[1]. The shopper now only has to fill in its CVC for any follow-up transactions at the webshop. Both transaction types are intended to make the payment flow as easy as possible for the shopper.

The e-commerce restriction limits this protocol to a subset of all card transactions, and therefore will not fully reduce fraudulent transactions. Merchants are still vulnerable for fraudsters using (stolen) card information of legitimate shoppers.

B-3-2 Conversion drop

As stated in the previous sections, merchants and payment providers want to make the payment flow as easy as possible for the shopper. It is well known in the payment industry that every extra step in the payment flow will give shoppers an extra ability to stop the process.

The conversion rate is defined as the ratio of shoppers that complete the payment process out of the shoppers that enter it. Merchants want their conversion as high as possible as this increases their revenues.

The 3Ds process causes more steps in the payment process.

1. The most common extra step is entering the 3Ds secret. This can be a simple extra step but gives the shopper time to rethink its purchase.
2. If the shopper has not used 3Ds before it should set this up *at the moment!* The shopper is likely unfamiliar with the process and due to the unexpected flow it could stop the purchase. This is highly likely if the shopper ironically thinks he/she is or might be victim to fraud when proceeding.
3. During the step where the shopper is authenticated he or she might have forgotten its secret. This could be either direct or indirect (after the extra step of thinking or searching the secret) cause the shopper to cancel the purchase.

Conversion optimization is a common activity for a merchant or Payment Service Provider (PSP). The conversion rate in general drops for transactions with 3Ds enabled due to the extra steps of 3Ds. Merchants decide not to use 3Ds to maximize the conversion rate and thus their revenues.

There is one popular example where conversion does not drop with the usage of 3Ds and this is in India. Over there almost all shoppers expect 3Ds authentication, and if they are not authenticated they exit the flow because they do not trust the situation.

[1] This is by law (PCI standards) restricted

Appendix C

Detailed Results

C-1 Experimental setup

C-1-1 The Environment

Choosing the language and tools for this research require a bit of thinking about the requirements. First, it should support the use of (various) classification methods. Second, it should be freely available[1]. Third, it must be proven technology, which we define as widely used with an active user community and properly documented features. Next to those it would be nice (pre) if it runs fast and can be integrated with the technology at the industrial partner (using Java).

Weka[2], a machine learning tool build in Java, with a Java API integration, seems like a good fit. However, experts at the industrial partner experienced regular crashes using Weka for big datasets, and therefore Weka seems unreliable. Together with the fact that there are more machine learning tools available, we choose not to use Weka.

In this thesis *R* is chosen as language to perform data analysis by exploration, manipulation and visualization, while *Python* is used for classification. In R it is easy to interactively play around with tools and functions (especially your data), which is a pre as researcher. Next to that, there are a few data scientists in the company, specialized in using R for data analysis and statistics. In python we use scikit-learn[32] for classification. The toolkit is open source (hosted on Github) and therefore easily extendible.

All experiments are executed on a 64-bits Ubuntu machine with a quad-core processor; Intel® Core™ i5-2540M CPU @ 2.60GHz (x4).

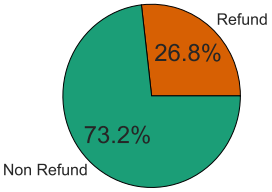


Figure C-1: Class distribution of selected merchant.

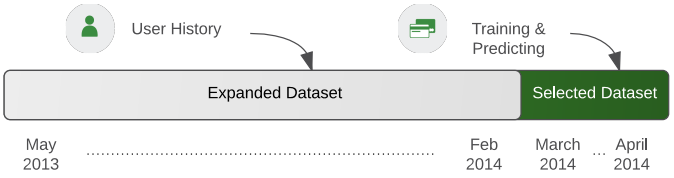


Figure C-2: Visual representation of the selected data and its expansion for User History

C-1-2 The data

This thesis is aimed to create value for both science as industry. The data should be chosen carefully, in such a way that it can be used as practical business case. Due to the exploratory nature of this research to refund classification, we decided to focus on one particular merchant. It is assumed that refund prediction will work different for different merchants, also spanning merchants, however, this is considered to be out of scope.

Deciding on the 'right' dataset require thinking about different criteria which are important for the aimed purpose.

- The research is focused on *retail merchants*, because refunds are a key asses of these merchants. Performing a refund automatically can be seen as extra service for the merchant its customer as well.
- A refund predictor will be highly valuable for merchants with a relatively *high refund rate*. Merchants with a high refund rate cannot decide to 'just route all transactions to a cheap route'. Imagine a merchant with a refund rate of 5% might think about this, while one under 0.5% can decide directly to do this. Note that really high refund rates (these do not exists in practice) would have less opportunity to re-route the non-refund transactions and therefore less interesting.
- The retail merchant should do business in *Belgium*, due to the potential business case of rerouting transactions of dual branded Bancontact/Mister Cash transactions.

To make the refund predictor useful in practice we should consider the aforementioned criteria. The selected merchant should be a retail merchant, doing its business (partially) in belgium, with a relatively high refund rate (between 10-30%) and preferably processing a big amount of transactions.

The selected merchant is a retail merchant selling overseas, but mainly processes transactions in (west) Europe. The merchant processes around 20 thousand transactions per month. This merchant is representative for the retail branche with a refund rate of 26.8%, see Fig. C-1.

The selected data contains two months of transactions, between March and April, 2014. This set contains almost 44 thousand transactions. In order to generate features for user history,

[1]Preferably open source as this is default policy at the industrial partner
[2]For detailed information we refer to: http://www.cs.waikato.ac.nz/ml/weka/

introduced in Section 6-2-3, we select ten additional months previous to the selected dataset. In Fig. C-2 the selected transactions and expanded selection for computation of user history are shown.

C-1-3 Split for training

The selected two months of transactions should be separated to form a proper train/test structure. Stratified sampling is used to split the groups. This sampling method samples independently from both classes. This way the resulting splits have the same statistical properties, ie refund rate.

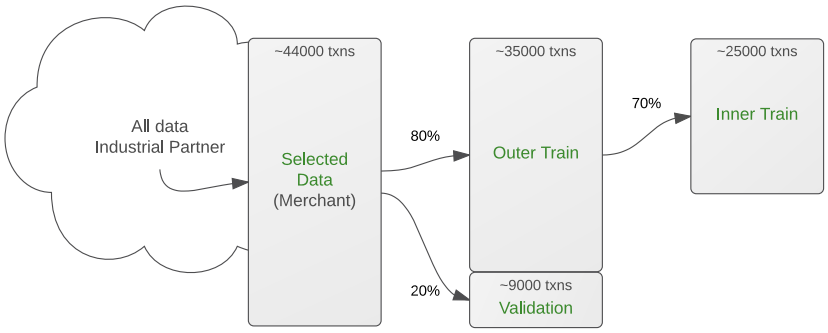


Figure C-3: Division of data in terms of train, (test) and validation. Both inner- and outer train set are trained using 10-fold cross-validation.

As shown in Fig. C-3 the data is divided into two sets, 80% for training and 20% for validating the final model. The validation set represent 'unseen data', and is not used before the final validation.

Due to the experiment structure, the training data is also called outer train. A subset, 70% of the outer train dataset, is called the inner train set. The model is first trained on the inner dataset, and the best parameters are used for training the outer training set. This structure will be explained in more detail in the next section.

C-1-4 Experiment design

Cross Validation (CV) is a validation technique that generalizes the results to an independent set. 10-Fold CV (10F-CV) divides the set randomly into 10 subsets, called folds. Nine folds represent the train set and the remaining one is the test set. Using ten iterations every fold represents the test set once. 10F-CV tests on different data then on which data is trained. The average result of the 10 iterations represent the CV score of the algorithm.

Structure of experiments

In all experiments different methodologies will be tested and compared against each other or the baseline. However, all those different methodologies may perform best under different circumstances, ie. parameters of the algorithms. Therefore, the best parameters for each

methodology will be selected first on a subset, which then will be used at a bigger set, with added unforeseen data.

Lets explain the train strategy by an example, comparing performance of two classifiers. The train strategy looks as follows;

1. In the *first CV*, the best parameter configuration and decision threshold are selected for each classifier, calculated over the inner train set. 10F-CV is executed for every parameter configuration and threshold. Thresholds are varied in the range $0, 0.01, 0.02, ..., 0.99, 1$. The parameter-threshold combination which on average scores best is selected as best configuration.
2. In the *second CV*, the average performance for every classifier is calculated over the outer train set, using the best parameters from the first CV. Extra data is added to this data set. 10F-CV is executed for every classifier using their best parameters and associated decision threshold for evaluation.

This method is called *double cross-validation* and will be used in all experiments. All cross-validation experiments are done using 10 folds, if not mentioned otherwise.

Parameters The first CV will search for the best performing parameters. For the Random Forest (RF) classifier we vary two parameters which together influence the complexity of the model. These parameters are the number of trees and the maximal depth of these trees. The variables are shown in Table C-1. Other parameters are left static. Gini impurity is used as internal loss function and the number of random features per weak learner is the square root of the number of features available.

The parameter search also includes a search for the best threshold given this parameter combination. The result of the first cross-validation is a combination of classifier parameters and decision threshold. Thresholds are varied in the range $0, 0.01, 0.02, ..., 0.99, 1$.

Parameter	Values
Max tree depth	[1, 2, 3, 4, 6, 8, 10, 12, 16, 20]
Number of trees	[1, 2, 4, 8, 16, 32, 64, 125, 250, 500, 1000, 2000]

Table C-1: Parameter variation for searching optimal combination.

C-2 Evaluation of Current Optimal Instance Score (cOIS)

Experiment In this experiment we will analyze the difference between using Current Optimal Instance Score (cOIS) and the loss function. cOIS is an extension on the loss function and takes into account continuous cost functions. The structure of the experiments, used tools and selected data is explained in Appendix C-1.

The decision threshold for the optimum loss and cOIS can be extracted from minimizing the underlying cost functions. The parameter search also includes a search for the best threshold given this parameter combination. The result of the first cross-validation is a combination of classifier parameters and decision threshold.

Results The optimal parameters of two equal estimators are selected using two different evaluation methods. A neutral metric to compare the methods in this context is in raw terms of costs. This includes basic fees for each route (so not only misclassification costs). In Table C-2 the results of the experiment can be found. The costs are visualized in Fig. C-4.

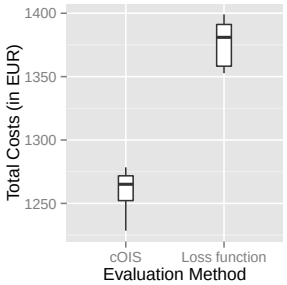


Figure C-4: Results of two estimators optimized with different evaluation methods.

	Selected Parameters			Total Costs	
Evaluation Method	Tree Depth	# Estimators	Threshold	Mean	Sd
Loss function	16	1000	0.81	1376.11	17.68
cOIS	1	32	0.71	1260.76	15.22

Table C-2: Results for training estimators with different evaluation methods.

cOIS decided on a simple model, using 32 trees with just one split. We call these trees decision stumps. We can see that an estimator, optimized using cOIS, minimizes the costs significantly better compared to the usage of the loss function. cOIS scores on average €115 lower, which corresponds to 8% lower costs.

C-3 Baseline

The structure of the experiments, used tools and selected data is explained in Appendix C-1.

Results The top 3 performing parameters in the inner training set are shown in Table C-3. The best performing configuration exists of 32 trees with a depth of only one. Such trees are called decision stumps. The follow-up configurations also use decision stumps. The second best performs almost equal, while it consists of only two trees.

The maximal average cross-validation score is measured for very simple configurations. The top 3 scoring configurations all consists of 2 to 32 decision stumps. This presumes that only a few number of features, perhaps one or two, are really important for the classification.

Selected Parameters			cOIS	
# Estimators	Max Depth	Threshold(s)	Mean (▽)	Sd
32	1	0.71	0.34631	0.01858
2	1	0.68-0.74	0.34624	0.01850
4	1	0.69	0.34624	0.01850
Tested selected parameters				
32	1	0.71	0.34144	0.01345

Table C-3: Top 3 results for parameter search on inner training set for baseline predictor. The score with the selected parameters in the second cross-validation is shown separately.

Parameter influence In Fig. C-5 the impact of the number of trees is shown for a subset of maximal tree depths. The complexer trees (depth of 16) is converging around 0.25, due to overfitting on the train set. The less complexer trees perform significantly better, especially for small number of fitted trees.

In Fig. C-6 the impact of the maximal tree depth is shown for a subset number of trees. For all number of trees, the performance is gradually increasing when increasing the depth of the tree. However, the performance for lower complexity trees, with just a few trees is performing significantly higher for the cross-validation score.

Threshold influence The impact of maximal tree depth and the number of trees for deciding on the threshold is shown respectively in Figs. C-7 and C-8. The best performing parameters both have a peak at threshold 0.71 and then a performance drop for greater threshold values. This can be due to the fact that the probabilities are not that distributed due to the choice for the simplest weak learners.

The peaks in both figures at threshold 0.71 decreases quickly for neighboring threshold values. This configuration might be risky. A safer configuration might be one tree with a single split, deciding on a threshold of 0.66, see the constant high performing (but variable) line in Fig. C-8.

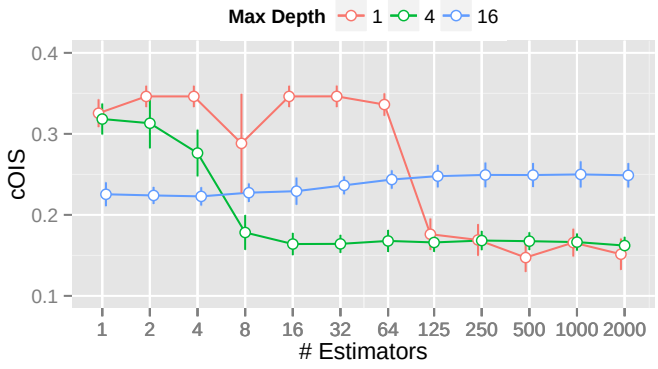


Figure C-5: Results of different parameters in 10F-CV. Threshold is constant at 0.71. Error bars show 95% confidence interval.

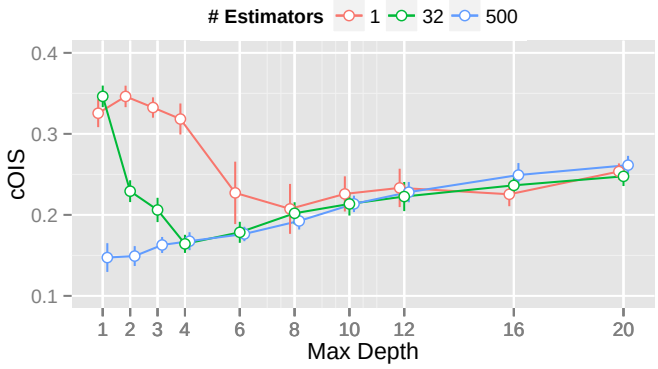


Figure C-6: Results of different parameters in 10F-CV. Threshold is constant at 0.71. Error bars show 95% confidence interval.

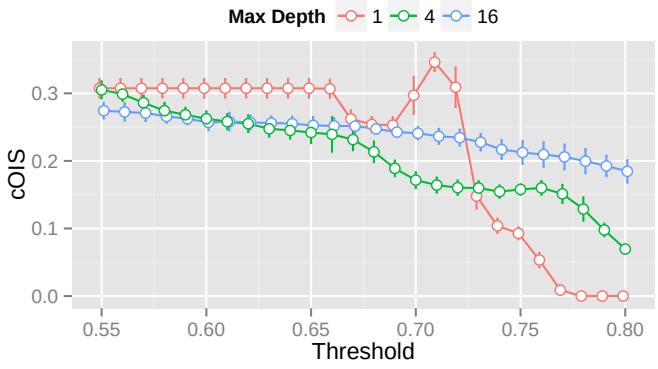


Figure C-7: Results of different threshold values for different maximal tree depths in 10F-CV. Number of estimators is constant at 32. Error bars show 95% confidence interval.

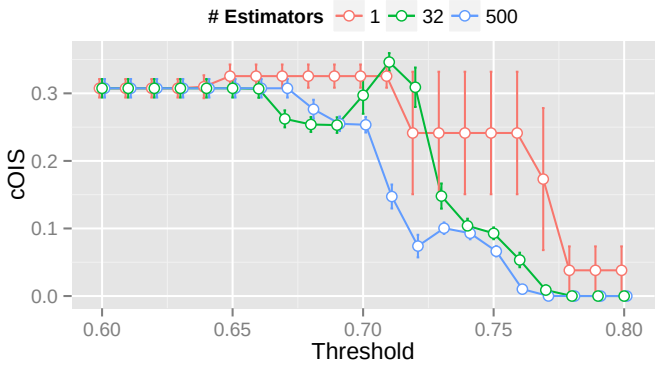


Figure C-8: Results of different threshold values for different number of estimators in 10F-CV. Max depth is constant at 1. Error bars show 95% confidence interval.

C-4 Best Predictor

The structure of the experiments, used tools and selected data is explained in Appendix C-1.

Results Parameters are being selected in the first loop of the double cross-validation. The top 3 best performing selections of parameters are shown in Table C-4. The best set of parameters are rather are complexer than the selected parameters for the baseline. A reason for this increase in complexity is the addition of features. In the baseline there was one groundbreaking feature, the transaction value. In the best predictor more features are together important which create a more complex model. In the next paragraphs we will see this complexity stabilizes at an acceptable level.

Top 3 Tested Parameters			cOIS	
# Estimators	Max Depth	Threshold(s)	Mean (▽)	Sd
1000	8	0.51	0.49289	0.01849
250	10	0.51	0.49274	0.01916
2000	8	0.51	0.49190	0.01862
Tested selected parameters				
1000	8	0.51	0.49306	0.01107

Table C-4: Top 3 results for parameter search on inner training set for best predictor. The score with the selected parameters in the second cross-validation is shown separately.

Parameter influence In Fig. C-9 the impact of the number of trees is shown for a subset of maximal tree depths. It is shown that the performance eventually stabilizes for more weak learners in the model. For different depths the performance becomes stable around 250 learners. There is a little peak at 1000 learners for trees with depth 8. This configuration is chosen as best configuration.

The decision stumps which performed quite well on the baseline set no longer perform, and cannot reach a score of 0.4, visualized in Fig. C-10. Due to over fitting, too complex models (with a tree depth of 16) are performing lower than less complex models. Using weak learners with a tree depth between 6 and 12 performs best, where 8 is selected as optimal three depth. In addition is is shown that a few classifiers do not perform very well, even not for a few complex trees.

Threshold influence The impact of maximal tree depth and the number of trees for deciding on the threshold is shown respectively in Figs. C-7 and C-8. The region between 0.4 and 0.7 is shown because these lead to the highest scores. A nice curve is shown with a peak around threshold 0.51. In contrast of the baseline, this threshold is much more robust (remember the sudden drop 0.02 after the peak). A threshold of 0.51 has been chosen safely.

As shown the curves do not change a lot for different number of trees or depth of trees, as long as we omit to decide on the decision stumps.

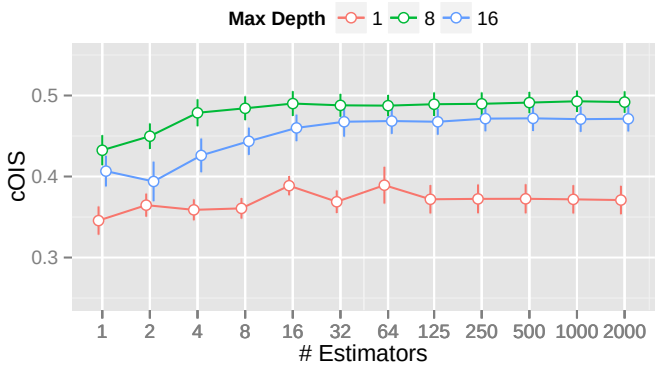


Figure C-9: Results of different parameters in 10F-CV. Threshold is constant at 0.51. Error bars show 95% confidence interval.

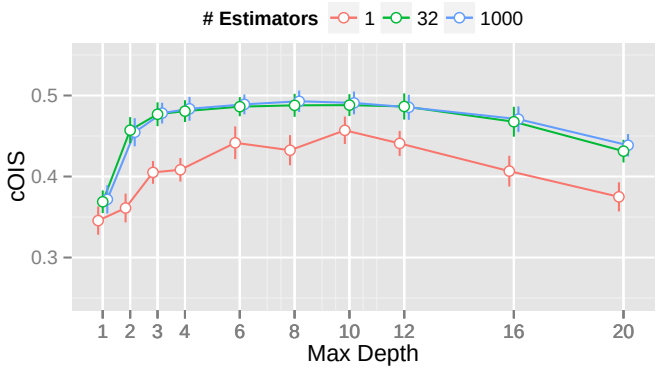


Figure C-10: Results of different parameters in 10F-CV. Threshold is constant at 0.51. Error bars show 95% confidence interval.

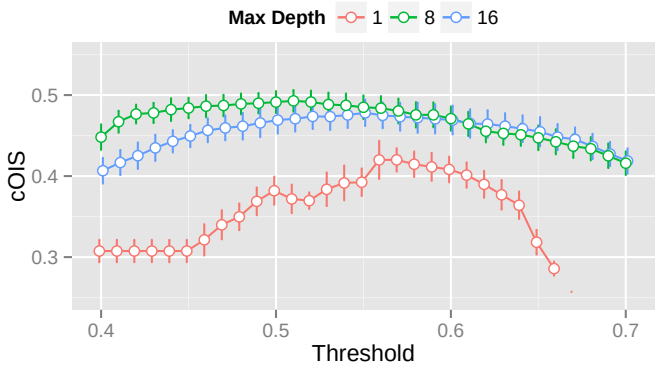


Figure C-11: Results of different threshold values for different maximal tree depths in 10F-CV. Number of estimators is constant at 1000. Error bars show 95% confidence interval.

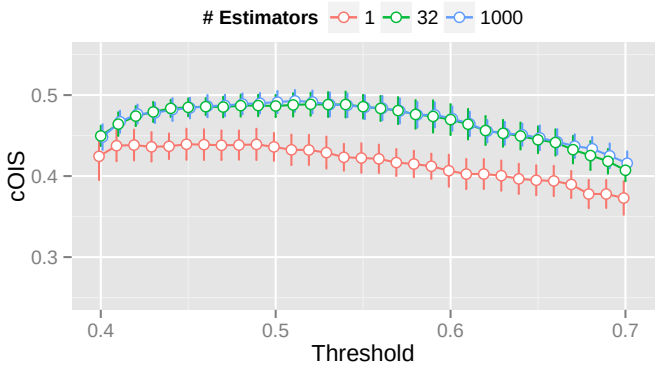


Figure C-12: Results of different threshold values for different number of estimators in 10F-CV. Max depth is constant at 8. Error bars show 95% confidence interval.

Bibliography

[1] A. Hummel and H. Kern, "An Agent-Based Simulation of Payment Behavior in E-Commerce," in *Multiagent System Technologies*, pp. 41–52, Springer Berlin Heidelberg, 2011.

[2] NewNet, "The Keys to Becoming a Successful Acquirer of Transactions in a Changing Payment Processing Environment," tech. rep., NewNet Communication Technologies, 2012.

[3] Solid Payments, "Routing Transactions." http://www.solidpayments.com/Routing-Transactions.htm. Accessed on: 26/03/14.

[4] N. Theuriet, "Product Review: Advanced Smart Routing," tech. rep., Bank of America, 2010.

[5] PaySourcing, "Intelligent transaction routing," tech. rep., PaySourcing.

[6] PayNetEasy, "Capabilities of Multi-Acquiring." https://payneteasy.com/payneteasy-multi-acquiring.html, 2013. Accessed on: 25/03/14.

[7] M. Krivko, "A hybrid model for plastic card fraud detection systems," *Expert Systems with Applications*, vol. 37, pp. 6070–6076, Aug. 2010.

[8] C. Phua, V. Lee, K. Smith, and R. Gayler, "A comprehensive survey of data mining-based fraud detection research," *arXiv preprint arXiv:1009.6119*, 2010.

[9] J. Gao, B. Ding, W. Fan, J. Han, and P. S. Yu, "Classifying Data Streams with Skewed Class Distributions and Concept Drifts," *IEEE Internet Computing*, vol. 12, pp. 37–49, Nov. 2008.

[10] C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, and N. M. Adams, "Transaction aggregation as a strategy for credit card fraud detection," *Data Mining and Knowledge Discovery*, vol. 18, pp. 30–55, July 2008.

[11] S. Jha, M. Guillen, and J. Christopher Westland, "Employing transaction aggregation strategy to detect credit card fraud," *Expert Systems with Applications*, vol. 39, pp. 12650–12657, Nov. 2012.

[12] N. V. Chawla, N. Japkowicz, and P. Drive, "Editorial : Special Issue on Learning from Imbalanced Data Sets Aleksander Ko l cz," vol. 6, no. 1, 2004.

[13] C. Phua, D. Alahakoon, and V. Lee, "Minority report in fraud detection: classification of skewed data," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 50–59, 2004.

[14] G. E. a. P. a. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD Explorations Newsletter*, vol. 6, p. 20, June 2004.

[15] T. Landgrebe, P. Paclík, and D. Tax, "Cost-based classifier evaluation for imbalanced problems," *Structural, Syntactic, and ...*, 2004.

[16] S. Nishisato and S. Nishisato, "Analysis of categorical data: Dual scaling and its applications," *Toronto: University of Toronto Press*, 1980.

[17] S. Nishisato, *Elements of Dual Scaling: an Introduction to Practical Data Analysis*. 1994.

[18] L. Fahrmeir and A. Hamerle, *Multivariate statistische Verfahren*. 1984.

[19] E. Fitkov-norris, S. Vahid, C. Hand, and K. Hill, "Evaluating the Impact of Categorical Data Encoding and Scaling on Neural Network Classification Performance : The Case of Repeat Consumption of Identical Cultural Goods," pp. 343–352, 2012.

[20] H. Kauderer and H. Mucha, "Supervised learning with qualitative and mixed attributes," *Classification, data analysis, and data highways*, 1998.

[21] V. Labatut and H. Cherifi, "Accuracy measures for the comparison of classifiers," *arXiv preprint arXiv:1207.3790*, 2012.

[22] A. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[23] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, June 2006.

[24] D. J. Hand, "Measuring classifier performance: a coherent alternative to the area under the ROC curve," *Machine Learning*, vol. 77, pp. 103–123, June 2009.

[25] R. D. Clark and D. J. Webster-Clark, "Managing bias in ROC curves.," *Journal of computer-aided molecular design*, vol. 22, no. 3-4, pp. 141–6, 2008.

[26] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," *Proceedings of the 23rd international conference on Machine learning - ICML '06*, pp. 233–240, 2006.

[27] K. Boyd, K. H. Eng, and C. D. Page, "Area under the Precision-Recall Curve : Point Estimates and Confidence Intervals," pp. 451–466, 2013.

[28] D. J. Hand, C. Whitrow, N. M. Adams, P. Juszczak, and D. Weston, "Performance criteria for plastic card fraud detection tools," *Journal of the Operational Research Society*, vol. 59, pp. 956–962, May 2007.

[29] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. 2008.

[30] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. New York: Chapman and Hall, 1984.

[31] L. Breiman, "Random forests," *Machine learning*, pp. 1–33, 2001.

[32] F. Pedregosa and G. Varoquaux, "Scikit-learn: Machine learning in Python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

Glossary

List of Acronyms

BC/MC	Bancontact/Mister Cash
PSP	Payment Service Provider
POS	Point of sale
3Ds	3D Secure
HPP	Hosted Payment Pages
PAL	Payment Acceptance Layer
ACM	Acquirer Connection Module
SVM	Support Vector Machine
SVC	Support Vector Classifier
RF	Random Forest
RFE	Recursive Feature Elimination
TFS	Tree-based Feature Selection
OSR	Overall Success Rate
TPr	True Positive rate
FPr	False Positive rate
ROC	Receiver Operating Characteristic
PR	Precision-Recall
AUC	Area Under ROC
AP	Average Precision

pAUC partial Area Under ROC

logAUC logarithmic Area Under ROC

OIS Optimum Instance Score

cOIS Current Optimal Instance Score

CBS Statistics Netherlands

Index