Master's Thesis Economics & Informatics

Identifying and Predicting Economic Regimes in TAC SCM

Using Sales and Procurement Information

Author:
Frederik Hogenboom, BSc
287986fh@student.eur.nl

Supervisors:
Prof. dr. ir. Uzay Kaymak
kaymak@ese.eur.nl

Wolfgang Ketter, Ph.D.
wketter@rsm.nl

Dr. Jan van Dalen
jdalen@rsm.nl

Computational Economics programme
Econometric Institute
Erasmus School of Economics
Erasmus University Rotterdam
February 14, 2009

Abstract

In this thesis, the effects of adding procurement information to a sales-based regime model, which is used for predicting price trends in a simulated supply chain, are researched. This supply chain is simulated in the TAC SCM game, which is an annual international competition held for several years, where researchers from around the world submit their artificial trading agents. The regime model extended in this thesis is used by the MinneTAC agent of the University of Minnesota.

We find that component offer prices can be used to extend the regime model, which is currently based on a one-dimensional Gaussian Mixture Model where probabilities are clustered. The resulting clusters hold as regimes. Extending the model with a new dimension results in newly defined regime clusters. Implementing the new regime model, MinneTAC increases its customer orders significantly. However, because the agent configuration shows a structural error in predicting future price trends – possibly due to an insufficient pricing mechanism – we have strong indications that our new approach leads to lower profits, although the decrease of the amount of cash at the end of a game is not significant. We believe that this decrease of profits can be tackled in the future by research into price trend prediction in the newly defined regime model.

Acknowledgements

Here, I would like to thank the many people who helped me finish this thesis. First of all, a special thanks to each of my supervisors – who devoted a lot of their time to this thesis project – for their support and supervision throughout the process of writing this thesis and for all the interesting meetings we had. I would like to thank my main supervisor Uzay Kaymak for his insights and general project supervision. Moreover, I have very much appreciated Wolf Ketter for his continuous flow of new ideas and for motivating me, as well as for his help with the regime model, and Jan van Dalen for his mathematical and statistical support.

Also, I would like to express my gratitude to John Collins of the University of Minnesota for all his suggestions and support for MinneTAC specific problems, as well as for enabling me to use servers and tools at the University of Minnesota, without which I would have never been able to finish my experiments.

Furthermore, I would like to thank the IT staff of RSM, which have put (and still put) a lot of effort in setting up an experimental environment, so that future projects regarding the MinneTAC agent can be done at the Erasmus University using powerful servers. Also, they have always responded quickly to my ongoing requests.

Finally, a special thanks to family, friends and loved ones, for their patience and understanding, but most importantly, for their support and confidence in me.

Contents

List of Figures

List of Tables

List of Symbols

This section contains a summary of symbols used in equations introduced in this thesis. In table 1, a definition is given for each symbol.

Symbol	Definition
α, β, γ	Smoothing coefficients
d	Current day
M	Number of regimes
N	Number of Gaussians in the Gaussian Mixture Model
np	Normalized sales price
$\widetilde{\text{np}}$	Double exponentially smoothed mean normalized sales price (approximated with mid-range)
$\widetilde{\text{np}}^{\max}$	Double exponentially smoothed maximum normalized sales price
$\widetilde{\text{np}}^{\min}$	Double exponentially smoothed minimum normalized sales price
op	Procurement offer price
$\widetilde{\text{op}}$	Exponentially smoothed mean procurement offer price
$P(\zeta_i)$	Prior probability of the i-th Gaussian of the Gaussian Mixture Model
$P(\zeta_i \vert \text{np})$	Posterior probability of the i-th Gaussian of the Gaussian Mixture Model, dependent on the normalized sales price
$P(\zeta_i \vert \text{np} \cap \text{op})$	Posterior probability of the i-th Gaussian of the Gaussian Mixture Model, dependent on the normalized sales price and procurement offer price
$P(\zeta \vert r)$	N by M matrix containing conditional probabilities resulting from `K-Means` clustering
$P(R_k \vert \text{np})$	Posterior probability of regime k dependent on the normalized sales price
$P(R_k \vert \text{np} \cap \text{op})$	Posterior probability of regime k dependent on the normalized sales price and procurement offer price
$p(\text{np})$	Density of the normalized sales price

Table 1: Summary of notation.

Symbol	Definition
$p\left(\mathrm{np}\middle\vert\zeta_i\right)$	Density of the normalized sales price, given the i-th Gaussian of the Gaussian Mixture Model
$p\left(\mathrm{np}\middle\vert R_k\right)$	Density of the normalized sales price, given the k-th regime
$p\left(\mathrm{np}\cap\mathrm{op}\right)$	Density of the normalized sales price and procurement offer price
$p\left(\mathrm{np}\cap\mathrm{op}\middle\vert\zeta_i\right)$	Density of the normalized sales price and procurement offer price, given the i-th Gaussian of the Gaussian Mixture Model
$p\left(\mathrm{np}\cap\mathrm{op}\middle\vert R_k\right)$	Density of the normalized sales price and procurement offer price, given the k-th regime
R_k	k-th regime, $k = 1, 2, \ldots, M$
$T\left(r_{d+n}\middle\vert r_d\right)$	Markov transition matrix for n days into the future
$\widetilde{\mathrm{tr}}^{\max}$	Predicted trend for future maximum sales prices
$\widetilde{\mathrm{tr}}^{\min}$	Predicted trend for future minimum sales prices
$\widetilde{\mathrm{tr}}^{\mathrm{np}}$	Predicted trend for future mean sales prices
$\widetilde{\mathrm{tr}}^{\mathrm{op}}$	Predicted trend for future mean procurement offer prices
x, y, z	Procurement variable types 1 through 3
$\widetilde{x}, \widetilde{y}, \widetilde{z}$	Exponentially smoothed procurement variable types 1 through 3

Table 1: Summary of notation, continued.

Chapter 1

Introduction

Because of the extremely competitive character of today's markets, it is valuable to gain insight in the dynamics of supply chains and to research supply chain optimization possibilities, both for individual elements in the chain, as well as for the chain as a whole. Every company is in a market and is also part of a supply chain, which is, according to Ghiani et al. [1], a complex logistics system in which raw materials are converted into finished products and then distributed to the final users (consumers or companies). A supply chain includes suppliers, manufacturing centers, warehouses, distribution centers and retail outlets. Within a chain, every element adds some sort of value to the final product (or service) and also fulfills a function within the chain.

A simplified supply chain consists of suppliers, traders, and customers. Such a chain is simulated in the Trading Agent Competition for Supply Chain Management (TAC SCM), which is an international competition for designing trading agents for an imaginary simulated personal computer supply chain. In the TAC SCM game, the individual agent's profits have to be maximized. Because of the economic relevance and the software engineering challenges, a lot of research has already been done on both the TAC SCM game and its participating agents. One of these agents is the MinneTAC agent, an agent created by the University of Minnesota (UMN). Despite all the research that has already been done, the performance of MinneTAC's numerous activities (like sales and procurement activities) can still be improved in many ways.

1.1 Background

In 2003, the first Trading Agent Competition for Supply Chain Management (TAC SCM) game was held. The TAC SCM game, as introduced by the Carnegie Mellon University (CMU) and the Swedish Institute of Computer Science (SICS), is an annual international competition which is designed to

promote and encourage high quality research into trading agents. TAC SCM is part of the International Trading Agent Competition, which also encapsulates the TAC Classic competition (since 2002). The TAC SCM game offers a complex simulated trading environment for a personal computer (PC) supply chain, in which teams from around the world can compete using their artificial trading agents. In each TAC SCM game, trading agents compete with each other in the sales market for customers and in a procurement market for computer components, trying to maximize their profits.

The competition attracts researchers from all over the world, because of its characteristics. The TAC SCM game environment is designed in such a way that it contains many characteristics that can be found in real-life supply chains as well, such as unpredictable opponents and interdependent chain entities. The supply chain simulated in the TAC SCM game offers many research opportunities into various subjects, such as price setting strategies, as well as prediction strategies for competitor behavior or market characteristics and developments. Because of its (increasing) complexity and size, new sub-competitions of the TAC SCM game have emerged over the past few years, which focus on specific parts of supply chain management, such as optimizing price predictions, which is the main activity in the TAC SCM Prediction Challenge.

As stated in the introduction of this chapter, the University of Minnesota (UMN) is one of the competitors in the TAC SCM game. Their MinneTAC trading agent [2] is a multi-component trading agent which bases its sales decisions on identified and predicted economic regimes, given the estimated normalized mean (sales) price. Until now, regime identification and prediction are solely based on sales information, which gives room for improvement of the agent's overall performance.

Besides MinneTAC, a large number of other agents also compete in the TAC SCM game, of which a few have good performance on a regular basis. Similar to the MinneTAC trading agent, these competitors apply prediction techniques as a basis of their decision making. The applied techniques vary for each agent. Also, each agent has its own specializations, so that one agent performs well in procurement-related tasks, whereas another agent performs well in sales-related tasks.

Well-known and well-performing trading agents of past TAC SCM games are TacTex [3], University of Michigan's DeepMaize [4], PhantAgent [5], University of Thessaloniki's Mertacor [6], and the CMieux agent [7], CMU's own submission to the TAC SCM game. The TacTex agent is based on machine learning algorithms to learn from historical (market) data, whereas the DeepMaize agent – winner of the TAC SCM 2008 Finals [8] – mainly relies on empirical game-theoretic analyses to be able to make predictions. The module-based Mertacor trading agent employs a combination of operations research techniques, heuristics, adaptive algorithms, and statistical modeling techniques. The PhantAgent is also characterized by the fact that it

does not use complicated algorithms throughout the agent, but that it uses simple heuristics and assumptions instead. One thing that distinguishes the CMieux agent from other agents, is that this trading agent continuously re-evaluates its (low-level and high-level) strategies, whereas a lot of other participating trading agents do not perform certain actions continuously, but only once in a while. Apparently, this approach yields good results, since the CMieux agent placed third in the TAC SCM 2008 Finals.

The MinneTAC trading agent has proven to be successful as well in past TAC SCM competitions by placing fifth and sixth in the TAC SCM 2005 and 2006 Finals respectively [9, 10]. MinneTAC's performance deteriorated in the games of 2007, where the agent did not make it past the TAC SCM 2007 Quarter-Finals [11], but improved again in 2008, when MinneTAC reached the TAC SCM 2008 Semi-Finals and placed fourth [12]. UMN's trading agent has proven to be able to compete with other strong competitors in the past, and therefore, improving the core of the agent might bring MinneTAC back in future TAC SCM finals.

1.2 Goal and Research Question

Because the performance of the MinneTAC agent in TAC SCM games falls behind with other competitors, the goal of this master's thesis is to investigate the possibilities of improving MinneTAC's existing regime identification and prediction process. These are the core features upon which most of the sales decisions made by the MinneTAC trading agent in a TAC SCM game are based. Thus, improving these processes could have a great impact on the agent's performance and is likely to result in a good ranking for MinneTAC in the TAC SCM games of 2009.

Conducting research in the fields of identifying and predicting economic regimes, as well as the MinneTAC agent and the TAC SCM game, is also relevant from an economics and informatics point of view. Combining techniques from informatics with economic theory to solve problems in economic environments contributes to novel approaches to existing problems. Furthermore, understanding the dynamics of supply chains and investigating strategies (e.g., for pricing or procurement) is important for players in today's highly competitive markets, because competitive advantages can be obtained when taking results of research in our fields into account. For instance, not being able to correctly identify and predict regime changes, which are important events in time series, causes (sales) decisions to be biased, which could result in a decrease of profit (or in bad rankings in case of a TAC SCM game). Identifying and predicting regimes more accurately enables players in a supply chain to make better decisions.

Since MinneTAC's regime identification and prediction both are currently solely based on sales information, we need to alter the processes in

such a way that other valuable information such as procurement information – which is also available to the agent – is used. Currently, procurement information is not used in MinneTAC's internal regime model, but is likely to influence pricing strategies and is (partially) available to the agent during a game, making it a good candidate for usage in an extended regime model. This information can be extracted from the market reports which are distributed to all agents in a TAC SCM game every twenty days. Also, information can be estimated or gathered during a game, since not all (procurement) information is available to an agent in every detail. Resulting from these observations, the main research question to be answered in this master's thesis is:

What is the effect of adding procurement information to MinneTAC's regime identification and prediction model?

1.3 Methodology

In order to be able to give an answer to the research question posed in section 1.2 and to achieve the objective mentioned in the latter section, the methodology as discussed here is used.

First of all, a literature survey is done, focussing on the exact specifications of the TAC SCM game and the characteristics of the MinneTAC trading agent and its competitors. This should give some proper understanding of the constraints posed by the environment MinneTAC is designed for, as well as the processes which are to be altered or improved. Furthermore, the literature survey enables us to elaborate on how regime identification and prediction are currently done.

Subsequently, ways of improving the regime identification and prediction are investigated, for instance how and when to utilize available procurement information. This results in an extended version of the current regime model, implementing procurement information. For that, we need a data set containing – among others – procurement information. This data set contains historical game data from TAC SCM games of 2007 and 2008 and is described in more detail in chapter 3.

In order to be able to extend the current regime model, we preselect one variable from the data set, which is the procurement information to be implemented in the agent's regime model. We determine the procurement variable with the highest likelihood of improving MinneTAC's performance by applying feature selection.

After extending the regime model with some procurement information and training new regime models using the data set, it is required to test the performance of the suggested improvements. First of all, performance of the improvements is assessed offline using all available historical data. Evaluation measures include entropies and correlations. Validation is done

by means of comparison to the current regime model and by evaluating the course of identified regime probabilities.

Finally, an optimal configuration is determined, which is subsequently implemented in the agent, after which several test games are run against other competitors. The results of these online experiments are used as verification of the regime model, and are evaluated by considering the changes in MinneTAC's bank account balance and its number of competitor orders at the end of a TAC SCM game.

1.4 Structure

The structure of the thesis is as follows. First, a literature review is presented in chapter 2, which discusses topics related to TAC SCM and MinneTAC, as well as other relevant areas. Subsequently, the data set to be used for some of our experiments is introduced in chapter 3. Also, we try to identify a good performing procurement variable in this chapter, which is used in chapter 4 to alter MinneTAC's regime identification and prediction processes. A newly defined regime model based on sales and procurement information is experimented with in chapter 5. Finally, experimental results are discussed in chapter 6 and conclusions are drawn in chapter 7.

Chapter 2

TAC SCM and MinneTAC: Introducing Regimes

This literature survey discusses the main specifications of the TAC SCM game, since we need to know the constraints posed by the environment when investigating the possibilities to extend MinneTAC's existing regime model. Furthermore, the MinneTAC trading agent is discussed in detail. This includes relevant economic models and the regime model as it is implemented now, in order to outline the model that is to be extended. Finally, we suggest some general improvements and we give an overview of the state of the art of a few competitor agents, to illustrate other approaches for trading in the TAC SCM game.

2.1 The TAC SCM Game

In 2003, the first Trading Agent Competition for Supply Chain Management (TAC SCM) game [13] was held. The TAC SCM game is a yearly international competition which is designed to promote and encourage high quality research into trading agents. The game simulates a supply chain for personal computers (PCs), which consists of customers, six traders, and eight suppliers. In total, sixteen types of PCs are available, which can be classified into three market segments: the low-, mid-, and high-range products. Each game simulates a total number of 220 trading days, on which human-created artificial trading agents place bids, buy products, and sell products, while trying to optimize their profits. Even though trading agents are human-created, human intervention is not allowed during a game. Generating supply and demand, generating market reports every twenty days with information about shipments and orders, and providing banking, production, and warehousing services are taken care of by the TAC SCM game.

Figure 2.1 illustrates the basic concepts of the TAC SCM game. The introduction of this section introduced the three main supply chain players:

customers, traders and suppliers. Since the implemented supply chain is customer driven, the customers are displayed on the left in the figure, whereas the suppliers are displayed on the right. The human-created trading agents are also referred to as manufacturers and these traders own factories, in which they assemble personal computers out of components procured from suppliers. Suppliers are controlled by the game and each have an own name, whereas customers are anonymous. Figure 2.1 shows the main interactions between the chain entities. Requests for quotation (RFQs) are translated into offers, of which the accepted ones turn into orders, which lead to shipments. Customers only buy PCs from traders, traders sell PCs to customers and buy PC components from suppliers, and suppliers sell PC components to traders. Neither trading, nor messaging takes place within groups of customers, traders, or suppliers. Also, the market reports mentioned in the introduction are periodically sent to traders, providing them information which could be useful for decision making.

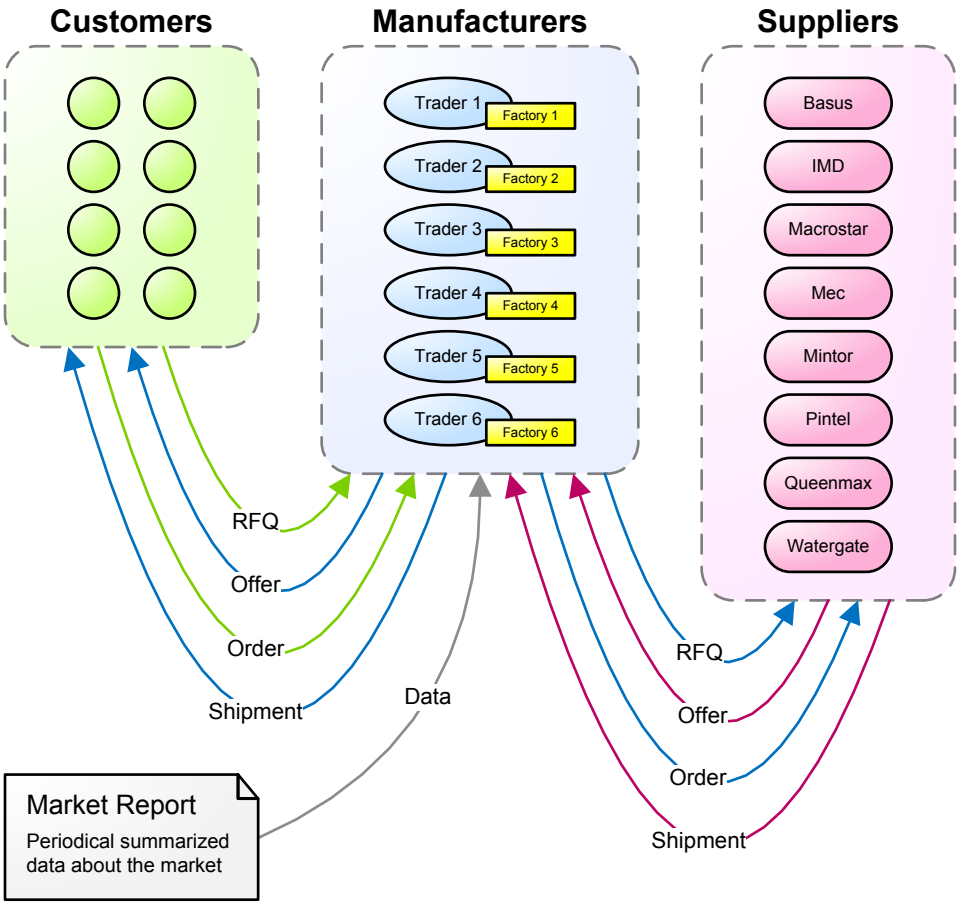


Figure 2.1: Schematic overview of the basic concepts of the TAC SCM game.

This being said, we can take a closer look at the characteristics of the TAC SCM game. As previously stated, a game consists of 220 simulated days. These TAC days have a duration of fifteen real seconds each. Every simulated day, requests for quotation for randomly chosen types of finished computers are issued by customers, which are controlled by the game itself and should be considered as given. The demand per market segment is drawn from a Poisson distribution (using a reverting random walk). Trading agents are designed to place bids on the RFQs. Customers select from the trader submitted quotes and buy products, generating profit for the trading agent.

Traders not only bid on customer RFQs, but they also assemble PCs, using four components, which can be procured from eight game-controlled suppliers, which all provide two products on a make-to-order basis. Sending RFQs to suppliers is constrained to a maximum of five RFQs per product per supplier, thus yielding a maximum of ten RFQs per supplier. Also, ordering from suppliers is accompanied with immediate billing for a portion of the ordering costs, also referred to as down payments. Every trader is able to handle all components and thus is able to assemble any PC type. However, the trader factory assembly capacity is constrained. Because of its bidding and assembly activities, a typical trading agent has to make some daily decisions about bidding on customer RFQs, about which supplier offers to accept, and about production and delivery scheduling. Also, trading agents have to take into account that suppliers may not always be able to supply the ordered quantities by the due date, resulting in partial or earliest complete offers. Deliveries always take up at least one day.

In each TAC SCM game, traders compete with each other in a sales market for customers and in a procurement market for computer components, trying to maximize their profits, which is measured using the agents' bank account balance at the end of the TAC SCM game. The agents generate income by selling products and by receiving daily savings interest in case of a positive account balance. Also, different costs are to be taken into account, like the costs for banking (interest costs in case of a negative account balance) and costs for production and warehousing. The latter costs are fixed throughout the game and are randomly chosen every time a new game starts. Each trading agent starts with no inventory and an empty bank account.

2.2 The MinneTAC Trading Agent

The University of Minnesota competes in the TAC SCM game with their MinneTAC trading agent [2]. This is a multi-component trading agent, which is built by a team of university students, staff, and partners.

The MinneTAC trading agent is founded on two pieces of software: the Apache `Excalibur` component framework [14] and the `AgentWare` package. The framework enables components to be constructed for and used in configurations of working agents, resulting in a highly flexible agent along with easier development and maintenance. The `AgentWare` package is distributed by the TAC SCM game organizers and handles all game server interactions.

The agent is divided into seven main components, which each are important and have their own roles to fulfill (see figure 2.2). The *Oracle* component embodies a large number of sub-components and is used for market and inventory models. It also performs analysis and prediction tasks. The *Oracle* basically provides the MinneTAC agent with all sorts of information and predictions, useful in decision making. The *Repository* component of the agent is used for facilitating data sharing among the agent's components. In order for the agent to work on a TAC SCM server, interaction with the game server should be dealt with, which is done by the *Communications* component. The other four components, i.e., the *Sales*, *Procurement*, *Production*, and *Shipping* components, are responsible for the major decision processes.

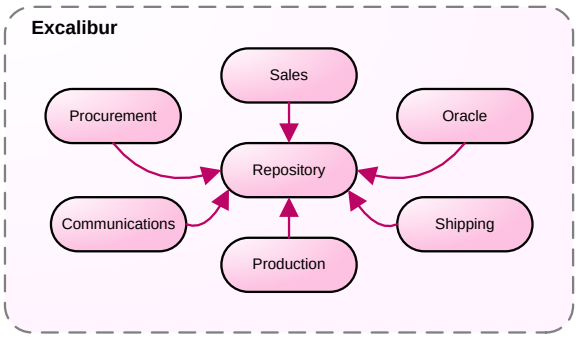


Figure 2.2: Schematic overview of the architecture of MinneTAC.

The trading agent's sales decision making is largely dependent on regime identification and prediction [15]. Every day, for each individual market segment, the current and possible future economic regimes are determined, using – among others – Gaussians and Markov matrices. Both regime identification and regime prediction are important tasks of the MinneTAC agent, because so many decisions are based upon current and expected regimes and their directly associated price-related forecasts, such as future sales price medians and trends. Regimes are used for both tactical and strategic decisions, such as product pricing and production planning. Section 2.3.2 discusses the regime model of the MinneTAC agent in more detail.

2.3 Regime Identification of the MinneTAC Agent

This section elaborates on economic regimes, which are already mentioned briefly in section 2.2. First of all, section 2.3.1 tries to come up with a definition of (economic) regimes and explains the use of regimes in the MinneTAC agent and the TAC SCM game. Subsequently, section 2.3.2 discusses the current application of economic regimes in the MinneTAC agent, while section 2.3.3 elaborates on a few possible improvements to MinneTAC's regime model, leaving the addition of procurement information out of consideration.

2.3.1 Introduction to Regimes

As already stated in section 2.2, the MinneTAC agent uses economic regime identification and prediction for taking tactical and strategic decisions, which are mainly related to sales. The regimes in the TAC SCM game can be considered as a set of characteristics which apply to a certain period of days. The identification and prediction of regimes is done so that different behavior can be modeled for different situations, which is also referred to as a switching model. In other words, problems can be solved differently, depending on their (regime) classification, which possibly yields a higher accuracy of the agent's predictions and higher profits.

Regimes are used in multiple contexts, e.g., political regimes and economic regimes. In general, a regime refers to a set of conditions. In economic context, regimes are also referred to as business cycle phases. These phases are commonly used in macro-economic environments, as is the case in [16], but in [17], regimes are applied in the micro-economic environment of the TAC SCM game. This makes sense, since an economic environment is simulated and one can capture (economical) characteristics in economic regimes, enabling an agent to reason (i.e., make tactical and strategic decisions) based on certain market conditions.

Over the past decades, research has been done not only related to identifying and predicting regimes, but also to regime changes. Regime changes are important events in time series, in which one can obtain strategic advantage if they are predicted or identified correctly. For instance in 1989, Hamilton published a paper about an approach to modeling changes in regimes [18]. Hamilton uses Markov matrices to observe these regime shifts, by drawing probabilistic inference about whether and when they may have occurred based on the observed behavior of series.

Finally, in the nineties, a lot of research has been done regarding applying fuzzy techniques in (macro-economic) regime switching models [19]. For instance, Nguyen et al. emphasize the importance of detecting economic regimes and make the observation that regime changes occur gradually [20]. In their research, they claim that regime changes can therefore best be described using fuzzy values. Even with fuzzy values, it remains hard to

identify exact moments where regimes change, and thus Nguyen et al. only succeed partially in their attempt to characterize a gradual regime change by a single moment of change.

The algorithms implemented in MinneTAC's regime model, which identifies and predicts economic regimes based on limited information, are based on economic theory and some research regarding regime switching models as previously discussed, but also incorporate some new or adapted techniques. Furthermore, it is possible to extend the agent's model by using some techniques discussed briefly in this section. We now continue by elaborating on MinneTAC's regime model.

2.3.2 MinneTAC's Regime Model

The regimes R which are identified in the agent are extreme scarcity, scarcity, a balanced situation, oversupply, or extreme oversupply [17] (see (2.1)). As stated by Ketter, five regimes are used instead of three (which are suggested by economic theory), because in this way outlier regimes can be isolated. Not only the regime of an arbitrary simulation day is determined, but also a regime prediction is made every day.

$$R = \{\text{exScar}, \text{scar}, \text{bal}, \text{osup}, \text{exOsup}\}\,. \tag{2.1}$$

Currently, MinneTAC's regimes are identified and predicted solely based on the normalized mean (sales) price [21]. Regime identification is currently done offline and online, in other words both out-of-game and in-game. Offer acceptance probabilities associated with given product prices (approximated using a Gaussian Mixture Model), derived from observable historical and current sales market data, are clustered offline using the `K-Means` algorithm [22], which yields distinguishable statistical patterns (clusters), which are labeled with the proper regimes after statistical research using correlations. Regime probabilities, which are indicative of how market conditions are, are determined by calculating the normalized price density of all clusters, given sales prices. The current regime is determined online by selecting the regime with the highest probability given the estimated normalized mean sales price. The mean price is defined as the mid-range price.

Short-term regime prediction for tactical decision making is done by using a Markov prediction process. This process is based on the last normalized smoothed mid-range price. To this end, Markov transition matrices, which are created offline (i.e., not in the game) by a counting process over past games, are being used. Long-term regime prediction is done by using a Markov correction-prediction process. This process is almost equal to the short-term regime prediction, but is based on all normalized smoothed mid-range prices up and until the previous day, instead of just the last normalized smoothed mid-range price. Alternatively, regimes can be predicted based on

exponentially smoothed price predictions, which is useful for estimating the current regime, as extensively elaborated in [15].

Normalized Mean Price

As both regime identification and prediction are based on normalized sales prices, this section introduces a mathematical formulation of the normalized price np for product g, np_g, on day d. The normalized price is calculated as

$$\text{np}_g = \frac{\text{price}_g}{\text{assemblyCost}_g + \sum_{j=1}^{\text{numParts}_g} \text{nominalPartCost}_{g,j}}, \tag{2.2}$$

using the price and manufacturing costs of g and the number of parts needed to produce g (price_g, assemblyCost_g, and numParts_g, respectively).

For calculation of the estimated normalized mean (mid-range) price, which is used for regime identification and prediction, yesterday's exponentially smoothed normalized minimum and maximum prices, $\widetilde{\text{np}}^{\text{min}}_{d-1}$ and $\widetilde{\text{np}}^{\text{max}}_{d-1}$, are used. The calculation of both prices is done the same way. Equations (2.3) through (2.5) show how the exponentially smoothed normalized minimum prices are calculated, using a Brown linear exponential smoother with an α of 0.5:

$$\widetilde{\text{np}}^{\text{min}'}_{d-1} = \alpha \cdot \text{np}^{\text{min}}_{d-1} + (1-\alpha) \cdot \widetilde{\text{np}}^{\text{min}'}_{d-2}, \tag{2.3}$$

$$\widetilde{\text{np}}^{\text{min}''}_{d-1} = \alpha \cdot \widetilde{\text{np}}^{\text{min}'}_{d-1} + (1-\alpha) \cdot \widetilde{\text{np}}^{\text{min}''}_{d-2}, \tag{2.4}$$

$$\widetilde{\text{np}}^{\text{min}}_{d-1} = 2 \cdot \widetilde{\text{np}}^{\text{min}'}_{d-1} - \widetilde{\text{np}}^{\text{min}''}_{d-1}. \tag{2.5}$$

Using previous equations, yesterday's exponentially smoothed normalized price on an arbitrary day d can be calculated using (2.6):

$$\widetilde{\text{np}}_{d-1} = \frac{\widetilde{\text{np}}^{\text{min}}_{d-1} + \widetilde{\text{np}}^{\text{max}}_{d-1}}{2}. \tag{2.6}$$

Regime Identification

Since this thesis is about altering the regime identification process, this section continues to elaborate on the currently used process to identify regimes in more detail. As discussed earlier, regime identification is done offline and online. Offline regime identification is done by analyzing data from past sales, which is also available during a TAC SCM game and which is representative enough for future market conditions.

As previously stated, Ketter et al. use a Gaussian Mixture Model (GMM) for identifying economic regimes. A GMM is used, since it is able to approximate arbitrary density functions. Also, a GMM is a semi-parametric approach which allows for fast computing and uses less memory than other

approaches [23]. In their model, fixed means, μ_i, which are equally distributed, and variances, σ_i^2, which are chosen so that adjacent Gaussians are two standard deviations apart [15], are used. The density of the normalized price as calculated in (2.2), $p\,(\mathrm{np})$, is defined as

$$p\,(\mathrm{np}) = \sum_{i=1}^{N} p\,(\mathrm{np}|\zeta_i)\,P\,(\zeta_i)\,. \tag{2.7}$$

In this equation, N is the number of Gaussian components. The prior probability of Gaussian component ζ_i, $P\,(\zeta_i)$, is determined by applying the Expectation-Maximization algorithm [24], which finds the maximum likelihood estimates of the parameters in the model, by repeatedly computing an expectation of the likelihood and maximizing this expectation. The i-th Gaussian, $p\,(\mathrm{np}|\zeta_i)$, can be written as shown in (2.8):

$$p\,(\mathrm{np}|\zeta_i) = p\,(\mathrm{np}|\mu_i \cap \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} e^{\left[\frac{-(\mathrm{np}-\mu_i)^2}{2\sigma_i^2}\right]}. \tag{2.8}$$

When applying Bayes' rule to (2.8), the posterior probabilities for each ζ_i, $P\,(\zeta_i|\mathrm{np})$, are obtained by applying

$$P\,(\zeta_i|\mathrm{np}) = \frac{p\,(\mathrm{np}|\zeta_i)\,P\,(\zeta_i)}{\sum_{i=1}^{N} p\,(\mathrm{np}|\zeta_i)\,P\,(\zeta_i)}, \quad \forall i = 1, 2, \ldots, N. \tag{2.9}$$

For each observed normalized price, a vector of posterior probabilities is computed, which is used in the `K-Means` algorithm. This algorithm, as mentioned before, is used for defining regimes. Applying the algorithm results in an N by M matrix with conditional probabilities for each component of the GMM for each regime, where M is the number of regimes. The density of the normalized price np dependent on the regime R_k is calculated as shown in (2.10):

$$p\,(\mathrm{np}|R_k) = \sum_{i=1}^{N} p\,(\mathrm{np}|\zeta_i)\,P\,(\zeta_i|R_k)\,. \tag{2.10}$$

When applying Bayes' rule to (2.10), the probability of R_k dependent on np, $P\,(R_k|\mathrm{np})$, is obtained by applying

$$P\,(R_k|\mathrm{np}) = \frac{p\,(\mathrm{np}|R_k)\,P\,(R_k)}{\sum_{k=1}^{M} p\,(\mathrm{np}|R_k)\,P\,(R_k)}, \quad \forall k = 1, 2, \ldots, M. \tag{2.11}$$

It is shown in [17] that a number of sixteen Gaussians yields good results. Furthermore, using five regimes instead of three regimes as suggested by economic theory yields good results. In other words, $N = 16$ and $M = 5$.

After the agent has characterized the different economic regimes offline, the dominant regime $\widehat{R}_r$ on arbitrary day d can be identified online, by selecting the regime with the highest probability, given the estimated normalized mean (sales) price $\widetilde{\text{np}}_{d-1}$, as shown in (2.12):

$$\widehat{R}_r \; s.t. \; r = \underset{1 \leq k \leq M}{\text{argmax}} \; \vec{P}(\widehat{R}_k | \widetilde{\text{np}}_{d-1}). \tag{2.12}$$

Finally, Ketter states one can measure the confidence in the regime identification by using the entropy [23]. Low entropy values (i.e., close to zero) indicate a high confidence, while higher values indicate a lower confidence.

Regime Prediction

In the introduction of this section, we introduced three techniques for regime prediction, each of which has – according to Ketter et al. [15] – its own characteristics and optimal time span to predict regimes for. Exponential smoothing can be applied to predict today's regime, whereas a Markov prediction process can be used for predicting short-term regimes (e.g., up to ten days in the future). A Markov correction-prediction process is most suitable for predicting long-term regimes. We define long-term predictions as predictions for up to twenty days in the future. The upper bound (or planning horizon h) is set to twenty days, because a new market report becomes available every twenty days, possibly leading to new or more accurate insights in future developments.

It should be noted that current implementations of the regime prediction framework in the agent differ slightly from the framework as it is explained in this chapter due to some tweaking by the MinneTAC team in order to optimize results. However, throughout this thesis, we focus on the model as it is explained in [15], since this framework describes the major regime-related prediction algorithms of the MinneTAC trading agent.

Also, in the latest configurations of the agent, there are no specific predictors assigned to prediction tasks for future days in a specific time span. Instead, ensemble prediction is used. Here, all implemented predictors predict for n days into the future (up to planning horizon h). The predictions are weighted and combined and the correctness of each prediction is evaluated daily. These daily evaluations result in weight changes. In general, the same optimal time spans for each technique are learned as are used in the framework, but ensemble prediction is a more flexible way of predicting and allows the usage of more predictors at once. The exact details of ensemble prediction are beyond the scope of this thesis and therefore we focus on the basic assignment of predictors when elaborating on the prediction framework.

Exponential Smoother Process

The exponential smoother regime prediction process is, according to Ketter et al., more reactive to the current market condition than any other method. This statement makes sense, since the exponential smoother process takes yesterday's information (normalized mean sales price) as input. This information is corrected (smoothed) with information on preceding days to reduce volatility.

The prediction process calculates a trend, $\widetilde{\mathrm{tr}}^{\min}_{d-1}$, in the minimum normalized mean sales price by using (2.3) and (2.4). The calculation is shown in (2.13), where β is set to 0.5:

$$\widetilde{\mathrm{tr}}^{\min}_{d-1} = \frac{\beta}{1-\beta} \cdot \left(\widetilde{\mathrm{np}}^{\min'}_{d-1} - \widetilde{\mathrm{np}}^{\min''}_{d-1} \right). \tag{2.13}$$

The exponentially smoothed maximum normalized trend, $\widetilde{\mathrm{tr}}^{\max}_{d-1}$, is calculated in a similar way. Using the minimum and maximum trends, the mid-range trend of the sales price ($\widetilde{\mathrm{tr}}^{\mathrm{np}}_{d-1}$) can be calculated as

$$\widetilde{\mathrm{tr}}^{\mathrm{np}}_{d-1} = \frac{\widetilde{\mathrm{tr}}^{\min}_{d-1} + \widetilde{\mathrm{tr}}^{\max}_{d-1}}{2}. \tag{2.14}$$

Using yesterday's value and the mid-range trend of sales prices, one can estimate the value of sales prices n days in the future in a way shown in (2.15), where h is the planning horizon:

$$\widetilde{\mathrm{np}}_{d+n} = \widetilde{\mathrm{np}}_{d-1} + (1+n) \cdot \widetilde{\mathrm{tr}}^{\mathrm{np}}_{d-1}, \quad \forall n = 0, 1, \ldots, h. \tag{2.15}$$

Recall that this horizon has a maximum of twenty days, and since the exponential smoothing process can be applied best for predicting today's regime, h is equal to zero for this process. Finally, the probability for each regime (given np) for n days in the future, $P\left(\widehat{R}_k | \widetilde{\mathrm{np}}_{d+n}\right)$, can be calculated similar to (2.11):

$$P\left(\widehat{R}_k | \widetilde{\mathrm{np}}_{d+n}\right) = \frac{p\left(\widetilde{\mathrm{np}}_{d+n} | \widehat{R}_k\right) P\left(R_k\right)}{\sum_{k=1}^{M} p\left(\widetilde{\mathrm{np}}_{d+n} | \widehat{R}_k\right) P\left(R_k\right)}, \quad \forall k = 1, 2, \ldots, M. \tag{2.16}$$

Here, the density of $\widetilde{\mathrm{np}}_{d+n}$ dependent on regime $\widehat{R}_k$ is calculated using (2.10) by marginalizing over the individual Gaussians and cluster centers, and thus we obtain

$$p\left(\widetilde{\mathrm{np}}_{d+n} | \widehat{R}_k\right) = \sum_{i=1}^{N} p\left(\widetilde{\mathrm{np}}_{d+n} | \zeta_i\right) P\left(\zeta_i | R_k\right). \tag{2.17}$$

Markov Process

Making short-term and long-term regime predictions can be done using Markov prediction and correction-prediction processes. In contrast to the exponential smoother process where future prices are predicted, resulting indirectly in predictions of future regimes, regimes are predicted directly. Also, Markov processes are less responsive to current market situations, because they also take into account a history of events.

For short-term regime predictions, a Markov prediction process is used. This process is based on the last price measurement and on a Markov transition matrix, referred to as $T\left(r_{d+n}|r_d\right)$. The latter matrix is created by means of a counting process on offline data and contains posterior probabilities of transitioning to regime r_{d+n} on day $d+n$ (i.e., n days in the future), given r_d, which is the current regime.

Note that for Markov processes, we introduce a new symbol for denoting regimes, r, to emphasize that we are not looking at the individual regimes in the way we were looking at them until now, because there is a focus shift. Now, the regimes represent rows and columns in a Markov transition matrix and we use probability vectors combined with transition matrices, instead of single regime probabilities.

Ketter et al. distinguish between two types of Markov predictions: n-day prediction and repeated one-day prediction. The first type is an interval prediction, where for each day n up to planning horizon h a Markov transition matrix is computed offline (per product, or at whatever level of detail the regime model is defined), whereas the second type only needs one Markov transition matrix (per product). This matrix is repeated n times up to planning horizon h.

The calculation of the n-day prediction for n days ahead is performed as

$$\vec{P}(\widehat{r}_{d+n}|\widetilde{\mathrm{np}}_{d-1}) = \sum_{r_{d+n}} \dots \sum_{r_{d-1}} \left\{ \vec{P}(\widehat{r}_{d-1}|\widetilde{\mathrm{np}}_{d-1}) \cdot T_n\left(r_{d+n}|r_{d-1}\right) \right\}, \quad \forall n = 0, 1, \dots, h, \quad (2.18)$$

where the previous (identified or predicted) posterior regime probabilities dependent on the normalized mean sales prices are multiplied with the applicable Markov transition matrix.

Hence, predictions for today (i.e., n is equal to zero) are based on the regime probabilities resulting from the regime identification of day $d-1$, while predictions for n days in the future are done recursively. The regime probabilities resulting from identification, i.e., $\vec{P}(\widehat{r}_{d-1}|\widetilde{\mathrm{np}}_{d-1})$, are obtained by rewriting (2.16), as is shown in (2.20) and (2.21). Note that we first need to rewrite the density of $\widetilde{\mathrm{np}}_{d+n}$ dependent on $\widehat{R}_k$, which is defined as shown in (2.17), to $\widetilde{\mathrm{np}}_{d-1}$ dependent on $\widehat{R}_k$. The latter density is used in (2.20)

in order to obtain the posterior regime probability. This density of $\widetilde{\text{np}}_{d+n}$ dependent on $\widehat{R}_k$ is calculated as

$$p\left(\widetilde{\text{np}}_{d-1}|\widehat{R}_k\right) = \sum_{i=1}^{N} p\left(\widetilde{\text{np}}_{d-1}|\zeta_i\right) P\left(\zeta_i|R_k\right), \tag{2.19}$$

$$P\left(\widehat{R}_k|\widetilde{\text{np}}_{d-1}\right) = \frac{p\left(\widetilde{\text{np}}_{d-1}|\widehat{R}_k\right) P\left(R_k\right)}{\sum_{k=1}^{M} p\left(\widetilde{\text{np}}_{d-1}|\widehat{R}_k\right) P\left(R_k\right)}, \quad \forall k = 1, 2, \ldots, M, \tag{2.20}$$

$$\vec{P}\left(\widehat{r}_{d-1}|\widetilde{\text{np}}_{d-1}\right) = \left\{P\left(\widehat{R}_1|\widetilde{\text{np}}_{d-1}\right), \ldots, P\left(\widehat{R}_M|\widetilde{\text{np}}_{d-1}\right)\right\}. \tag{2.21}$$

The same principles apply to the calculation of the repeated one-day prediction, as shown in (2.22). However, as we already explained, only one transition matrix is used, $T_0\left(r_d|r_{d-1}\right)$, which is repeated n times for predictions of n days in the future.

$$\vec{P}(\widehat{r}_{d+n}|\widetilde{\text{np}}_{d-1}) = \sum_{r_{d+n}} \cdots \sum_{r_{d-1}} \left\{\vec{P}(\widehat{r}_{d-1}|\widetilde{\text{np}}_{d-1}) \cdot \prod_{t=0}^{n} T_0\left(r_d|r_{d-1}\right)\right\}, \quad \forall n = 0, 1, \ldots, h. \tag{2.22}$$

There is an assumption in the repeated one-day prediction, i.e., in a time span of multiple days, the transition probabilities remain the same. A shortcoming of repeating matrices is the fact that the distribution of probabilities in the transition matrices will converge to fixed values after repeating the matrix a number of times. In other words, eventually a stationary distribution will emerge. For one-day predictions, this will occur sooner than for n-day predictions, simply because for every prediction for n days up to planning horizon h, the matrix should be repeated n times, whereas the n-day predictions have different matrices for each prediction and therefore no repeating occurs. Because of these observations, we prefer the n-day predictions over the repeated one-day predictions.

Long-term predictions are made using a Markov correction-prediction process. The latter process is almost similar to the Markov prediction process we already discussed. The difference is that the long-term prediction process is modeled based on the entire history of prices, instead of just the last price measurement.

Hence, we need to extend the probability of regime $\widehat{r}_{d-1}$ dependent on $\widetilde{\text{np}}_{d-1}$ to also incorporate all previous $\widetilde{\text{np}}$ values. This correction is done by applying a recursive Bayesian update to (2.19) and (2.20), as shown in (2.23) and (2.24).

$$P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right) =$$
$$\frac{p\left(\widetilde{\mathrm{np}}_{d-1}|\widehat{R}_k\right) P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-2}\right\}\right)}{\sum_{k=1}^{M} p\left(\widetilde{\mathrm{np}}_{d-1}|\widehat{R}_k\right) P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-2}\right\}\right)}, \quad \forall k = 1,2,\ldots,M, \tag{2.23}$$
$$\vec{P}\left(\widehat{r}_{d-1}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right) =$$
$$\left\{P\left(\widehat{R}_1|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right),\ldots,P\left(\widehat{R}_M|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right)\right\}. \tag{2.24}$$

Here, the regime probabilities are calculated while taking into account the predicted probabilities of yesterday $(d-1)$ at the day before yesterday $(d-2)$ and yesterday's probability density of $\widetilde{\mathrm{np}}_{d-1}$ dependent on $\widehat{R}_k$. This density is calculated by applying (2.19).

Then, predictions for today (i.e., n is equal to zero) are based on the corrected regime probabilities resulting from (2.24) on day $d-1$, while predictions for n days in the future are done recursively. We define the n-day variant of the Markov correction-prediction process as

$$\vec{P}\left(\widehat{r}_{d+n}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right) =$$
$$\sum_{r_{d+n}}\cdots\sum_{r_{d-1}}\left\{\vec{P}\left(\widehat{r}_{d-1}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right)\cdot T_n\left(r_{d+n}|r_{d-1}\right)\right\},$$
$$\forall n = 0,1,\ldots,h. \tag{2.25}$$

The same principles apply to the calculation of the repeated one-day correction-prediction, as shown in (2.26). Again, the difference is the usage of the Markov transition matrices:

$$\vec{P}\left(\widehat{r}_{d+n}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right) =$$
$$\sum_{r_{d+n}}\cdots\sum_{r_{d-1}}\left\{\vec{P}\left(\widehat{r}_{d-1}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\right)\cdot\prod_{t=0}^{n} T_0\left(r_d|r_{d-1}\right)\right\},$$
$$\forall n = 0,1,\ldots,h. \tag{2.26}$$

Note that the Markov transition matrices which are being used in (2.25) and (2.26) ($T_n\left(r_{d+n}|r_{d-1}\right)\forall n = 0,1,\ldots,h$ and $T_0\left(r_d|r_{d-1}\right)$, respectively) are the same matrices as used in (2.18) and (2.22). Furthermore, the prediction algorithms used are similar as well.

Final Remarks on Prediction

For each of the discussed processes, the future dominant regime can still be determined as done with regime identification in (2.12), i.e., by selecting the regime with the highest probability, given certain conditions regarding the estimated normalized mean (sales) price.

Furthermore, where needed, the (prior) regime probabilities for the first day of a game are set to 100% extreme scarcity, since this is the most

intuitive regime when the game starts, because no inventories of finished products exist yet.

Finally, Ketter states one can evaluate regime predictions by calculating a percentage on the number of times the correct dominant regime has been predicted, as well as the number of times a regime change has been predicted correctly within plus or minus two days [23]. These methods are based on discrete values, i.e., the dominant regimes. Ketter uses the Kullback-Leibler (KL) divergence to determine the closeness of all individual predicted regime probabilities (which are continuous) to the realized regime probabilities. Also, the prediction error is estimated using a Monte Carlo method.

Price Prediction

The predicted regimes up to h days in the future, which is done using equations introduced in the previous section, can be used to predict future price density distributions. Equation (2.27) shows a way to create a price prediction distribution for n days ahead, based on the predicted regimes, given a history of prices:

$$p\left(\widehat{\mathrm{np}}_{d+n} | \left\{\widetilde{\mathrm{np}}_1, \ldots, \widetilde{\mathrm{np}}_{d-1}\right\}\right) = \sum_{k=1}^{M} p\left(\mathrm{np} | R_k\right) P\left(\widehat{R}_{k,d+n} | \left\{\widetilde{\mathrm{np}}_1, \ldots, \widetilde{\mathrm{np}}_{d-1}\right\}\right), \quad \forall n = 0, 1, \ldots, h. \tag{2.27}$$

Here, $P\left(\widehat{R}_{k,d+n} | \left\{\widetilde{\mathrm{np}}_1, \ldots, \widetilde{\mathrm{np}}_{d-1}\right\}\right)$ is an element of a resulting probability vector of regime predictions in the previous section. Furthermore, $p\left(\mathrm{np} | R_k\right)$ represents the density of the normalized price np dependent on the regime R_k and is obtained using (2.10).

Online, the price prediction is estimated by sampling over np from 0.00 to 1.25 with an increment of 0.01, after which the distribution is discretized and normalized, so that its values sum up to one. Then, this distribution can be used to calculate for instance a mean distribution, or the 10%, 50%, and 90% percentiles used for predicting price trends. Also, the customer offer acceptance probability can be calculated using the cumulative density function of the price density distribution. The exact calculation is beyond the scope of this thesis.

2.3.3 Determining Regimes Differently in MinneTAC

Because MinneTAC's regime model is to be altered in this thesis, we take a look at fuzzy set theory, which has already been used in the past for modeling the identification of (changes of) economic regimes [20]. The existing regime model of the MinneTAC trading agent does not use fuzzy set theory. Hence, not only procurement information, but also fuzzy sets would be an appropriate extension to the regime model. While fuzzy sets are used

to express the uncertainty of a definition, normal sets are used to express the uncertainty of an occurrence. The regime determining problem clearly can be classified as a problem concerning uncertainty of a definition. Fuzzy set theory can be applied to two key components of regime identification and prediction: the determination of the clusters representing the different economic regimes and the regime determination based on these clusters.

While currently in the MinneTAC agent days are being classified to belong to a certain regime using probabilities and sets, game days can also be classified more precisely using fuzzy sets and membership functions, as introduced by Zadeh in 1965 [25]. In contrast to probabilities, these membership functions do not express the likelihood of regimes, but the degree of truth. With the current regime identification (as well as prediction), the regime of a certain day is assumed to be the regime with the highest likelihood, but when fuzzy set theory is applied, multiple regimes can partially apply to a single day, making the number of possible regimes infinite and the agent more accurate. This also adds more realism to the regime model, since in real life, economic regimes transition gradually, so that on an arbitrary day multiple regimes can hold.

When using fuzzy sets, the current dominant regime can be characterized as a fuzzy set R_r in the set containing all regimes R which has a membership function as displayed in (2.28):

$$\mu_{R_r} : R \to [0, 1] \,. \tag{2.28}$$

Membership functions can be of various types, for example triangular, trapezoidal or Gaussian. An example of a fuzzy set representation of regimes is shown in (2.29) (compare (2.12) and its resulting value). In this example, the identified or predicted dominant regime have a degree of membership of 50% in the oversupply regime and a 25% membership of the extreme oversupply regime:

$$R_r = \{0, 0, 0, 0.5, 0.25\} \,. \tag{2.29}$$

Implementing a fuzzy set representation of regimes would most likely require a rebuild of a significant part of the agent, since all decisions are made based on one identified or predicted regime, i.e., a crisp formulation of a regime. However, with Takagi-Sugeno fuzzy models, such a crisp output is also possible.

It is also possible to make regime definitions less crisp, using fuzzy `C-Means` clustering [26, 27]. Like the currently used `K-Means` algorithm, the proposed algorithm optimizes the cluster centers in a collection of data points (posterior probabilities for each Gaussian ζ_i in case of the MinneTAC agent) by calculating the distance of each data point to the cluster centers, using a distance measure like the Euclidian distance. The algorithm which is currently used does not allow data points to belong to two or more clusters with a certain membership, while the fuzzy `C-Means` algorithm does.

We believe that the agent could benefit from changing `K-Means` clustering into fuzzy `C-Means` clustering, since clusters (and their cluster centers) are defined more accurately, because important information, i.e., the extent to which data points match a certain cluster, is kept instead of simplified. Furthermore, the proposed clustering algorithm is more stable than `K-Means`, because outliers do not influence (offset) cluster centers much.

As shown in this section, fuzzy `C-Means` clustering and fuzzy sets can be used to obtain a new definition and classification of regimes. Because of the improved accuracy in identifying and determining regimes when applying the methods elaborated, the MinneTAC trading agent can benefit from implementing fuzzy sets and fuzzy `C-Means` clustering.

2.3.4 Regimes Summary

In this section, we have introduced (economic) regimes. After discussing some related work, we have discussed the regime model of MinneTAC, which identifies and predicts economic regimes in a TAC SCM game based on normalized mean sales prices, extensively. Finally, an adaptation of the regime model is suggested based on related work, i.e., introducing fuzzy set theory.

2.4 Competitors

MinneTAC is not the only agent competing in the TAC SCM game. This section elaborates on six agents which performed well at the TAC SCM Finals of 2007 and/or 2008 [28, 8], based on the results of a survey conducted by Collins et al. [2] and official publications about the agents. Each of the agents has different approaches to trading in the TAC SCM game.

The TacTex trading agent [3] of the University of Texas is discussed in section 2.4.1. Section 2.4.2 elaborates on University of Michigan's DeepMaize agent [4]. The CrocodileAgent [29], which is created by the University of Zagreb, is discussed in section 2.4.3. Subsequently, sections 2.4.4 and 2.4.5 discuss the PhantAgent of the University Politehnica of Bucharest [5] and the CMieux agent of the Carnegie Mellon University [7], respectively. Finally, section 2.4.6 describes the main characteristics of the Mertacor trading agent of the University of Thessaloniki [6].

2.4.1 TacTex

The TacTex agent uses machine learning algorithms to learn from historical (market) data. TacTex uses both offline bootstrapping and online learning, which means the agent can adapt to changing situations. TacTex has three predictive modules, two decision-making modules and two methods of adapting to opponent behavior based on past games [3].

The decision-making modules try to identify optimal behavior with respect to the predictions. These two models are called the *Supply Manager*, which handles all planning related to component inventories and purchases (i.e., it decides what and how to order), and the *Demand Manager*, which is used for planning related to computer sales and production.

In order to be able to perform their optimization tasks, the managers use three predictive models. The first model is the *Supplier Model*, which predicts the prices offered by suppliers in response to RFQs. The second predictive model which is used by the managers is the *Demand Model*. This model predicts the future demand using a Bayesian approach. Finally, the third model enabling the optimization tasks of the two decision-making modules, is the *Offer Acceptance Predictor*. This model enables the *Demand Manager* to predict the orders that will result from the offers it makes.

TacTex's predictive modules' predictions are based on observations on the current game. These predictions can be adapted by using historical data. The agent uses data of old games in which their current opponents were represented. The TacTex trading agent adapts predictions both in the initial component orders, as well as in the end-game sales, since making precise predictions is hard in begin-of-game and end-of-game scenarios, thus requiring some adaptation.

2.4.2 DeepMaize

The DeepMaize agent is designed in such a way, that it tries to optimize the estimation of the marginal values for each finished product and component input by using predictions about market conditions and constraints on production.

Optimization of constraints posed by the TAC SCM game on a realtime basis is done by the agent, by using third party optimization packages. As stated in [4], both centralized, high-level optimization and low-level optimization is used. The high-level optimization is used for a long-term projected production schedule and accounts for the major constraints, without worrying about the details. The low-level optimization decides about actions in the individual markets and keeps all the details in mind.

The DeepMaize agent mainly estimates the daily demand (defined as the number of customer requests for quotation) and tries to predict sales prices, based on historical data to which a `k-Nearest-Neighbor` algorithm has been applied. This data is split into two data sets; the first containing data on previous tournament games, and the second containing data on self-played games. Predictions are optimized by applying a logarithmic scoring rule.

The results of the survey presented in [2] indicate that the DeepMaize agent, in contrast to the machine learning approach of the MinneTAC and TacTex trading agents, mainly relies on empirical game-theoretic analyses

to be able to make predictions, which requires the conduction of a great many of experiments.

2.4.3 CrocodileAgent

The CrocodileAgent is based on the JADE [30] framework, which is a suitable Java-based framework for large-scale multi-agent simulations. Also, CrocodileAgent's decisions are taken with use of the IKB model [31], which is a strategy design framework for electronic trading markets, consisting of three layers, each of which the first letter of their name is included in the acronym IKB: the *Information* layer containing collected data from the environment, the *Knowledge* layer containing extracted knowledge from the previous layer, and finally the *Behavioral* layer, which is responsible for reasoning and decision making.

2.4.4 PhantAgent

The architecture of the PhantAgent [5], created by the University Politehnica of Bucharest, is divided into three inter-dependent modules: the *Computer Module*, *Component Module*, and *Factory Module*. PhantAgent does not use optimization algorithms for subproblems to be solved, but uses heuristics, since the different modules are inter-dependent, which may lead to sub-optimal overall solutions, even though the subproblems are optimized. Characteristic for the PhantAgent is the fact that it does not use complicated algorithms throughout the agent, but it uses simple heuristics and assumptions instead.

The first module, the *Computer Module*, is responsible for handling the computer sales in a TAC SCM game. Therefore, the module should decide which computers should be offered for selling at what price. Offer prices are calculated using the best highest price of the past three days, adjusted with a factor depending on current factory utilization. The PhantAgent tries to utilize the factory to its maximum if the profit is positive.

The *Component Module* handles component procurement as well as component cost estimation tasks. For instance, there should always be enough (but not too much) stock components available for the *Factory Module* to process. Also, procurement costs should be kept as low as possible.

Finally, allocating computer production is performed by the third module; the *Factory Module*. According to [5], the module handles the assembly factory and is responsible for fulfilling existent customer orders, producing computers in advance, deciding what customer RFQs could be delivered on time, and estimating how crowded the factory is.

2.4.5 CMieux

The results of the TAC SCM Finals of 2007 and 2008 show that the CMieux trading agent [7] performs well. In their paper, in which the CMieux agent is discussed on a high level, Benisch et al. state that the trading agent outperforms other TAC SCM contestants significantly in procurement and performs about as good as other agents in bidding. One thing that distinguishes the CMieux agent from other agents, is that this trading agent continuously re-evaluates its (low-level and high-level) strategies, whereas a lot of other participating trading agents do not perform certain actions continuously, but only once in a while.

The trading agent is supported by five modules. The first module is called the *Forecast Module*, which predicts prices of components and the future customer demand, as well as component arrivals. These predictions are made based on new, daily information gathered from the server (for instance customer requests for quotation or observed delays). Demand forecasting is done using a Poisson distribution, with which the mean and trend of the demand are predicted using linear least squares fit. Price prediction is calculated the same way for selling prices. Purchasing prices are predicted differently with a `Nearest-Neighbor` algorithm.

The forecasts are used by the *Strategy Module*, which takes high-level strategic decisions. The module provides functionality to decide which customer requests for quotation should be targeted on an arbitrary game day, as well as which finished products the agent should try to sell on the current day (also referred to as *target demand* and *desired to promise products*, respectively).

The third module is called the *Procurement Module*. This module determines which supplier offers to accept on an arbitrary day (i.e., selecting a good subset of promising offers) using a rule-based model. The expected offer costs of the suppliers are minimized by the module, by breaking the target demand into orders, which are, particularly in case of low prices on an arbitrary day, not necessarily related to game days in the near future, but could also be related to expected demand many days ahead. Also, an order can be divided over multiple suppliers in order to minimize the costs. The module is also responsible for sending requests for quotations to suppliers.

The main task of the *Bidding Module* is to respond to customer requests with price quotes, or in other words, to place bids. Bidding for customer orders is modeled with probability distributions, which are learned offline with a distribution tree. Each product type is handled separately and in-game, the bidding tasks are narrowed down to a continuous knapsack problem.

The last module of the CMieux agent is called the *Scheduling Module*. This module is responsible for generating a tentative tardiness minimizing assembly schedule for a couple of days ahead using a greedy algorithm, after sorting orders (using heuristics).

2.4.6 Mertacor

The Mertacor trading agent is a module-based agent, which relies on a combination of operations research techniques, heuristics, adaptive algorithms, and statistical modeling techniques. Its modules represent four main facets of the TAC SCM game and show some similarities between the modules of PhantAgent and CMieux.

The *Inventory Module* handles all inventory-related tasks. Mertacor uses an assemble-to-order inventory management technique, where components (and not assembled products) are kept in stock and a certain inventory level is maintained at all times for each component. Chatzidimitriou et al. state that this technique proves to be suitable in environments where assembly times are significantly smaller than replenishment times [6], which is the case in TAC SCM games. The *Inventory Module* is assigned to decide which component has to be replenished at a certain time.

The *Procuring Module* gets its orders from the *Inventory Module*. Although the latter module is responsible for maintaining healthy inventory levels, the task of the former module is to buy components for low prices. However, low prices do not have a high priority on the the first two days in a TAC SCM game. Here, the *Procuring Module* sends RFQs for components and procures expensive components, in order to fill inventories so that production can start immediately. Cheap components are procured on other days by means of RFQ probing.

The *Factory Module* is responsible for a number of tasks. First of all, the module generates production and delivery schedules on a daily basis. Furthermore, the module adjusts inventory levels and the available factory cycles. The *Factory Module* implements a simulator, which simulates the entire factory up to a few days into the future, so that decisions can be made which may effect the future, such as bidding by the *Bidding Module*.

The latter module, the *Bidding Module*, cooperates with the *Factory Module*. For instance, the *Bidding Module* is responsible for placing bids (offers) on customer RFQs. In order to maximize the factory performance, the bidding module places more offers on RFQs than the factory can handle, because not all offers are accepted. Bidding is done based on machine learning techniques, and in case a predictive model fails due to unexpected market changes, Mertacor switches back to simple mechanisms.

2.4.7 Summary Competitors

In this section, architectures and decision logic of several competitors have been discussed. Table 2.1 summarizes on these two aspects of the discussed agents.

All agents are more or less module-based. However, agents differ in the division of the tasks to be fulfilled daily into modules, depending on their

Agent	Architecture	Decision logic
TacTex	Predictive, adaptive, and decision-making modules which distinguish between demand and supply	Machine learning: offline bootstrapping and online learning
DeepMaize	High-level and low-level modules	Empirical game-theoretic analyses
CrocodileAgent	IKB model	Current game situations
PhantAgent	Product-oriented modules, i.e., computers, components, and factories	Simple heuristics and assumptions
CMieux	High-level strategy, low-level forecast, scheduling, procurement, and bidding modules	Heuristics, rules, statistical modeling techniques, continuous re-evaluation of strategies
Mertacor	Modules for the main facets of TAC SCM, i.e., inventories, procurement, factories, and bidding	Operations research techniques, heuristics, adaptive algorithms, statistical modeling techniques

Table 2.1: Summary of competitors.

focus. First of all, the DeepMaize agent mainly distinguishes between high-level and low-level optimizations, whereas TacTex distinguishes between demand-related and supply-related modules. The CrocodileAgent is built based on layers with different levels of abstraction using the IKB model. The PhantAgent is product-oriented, i.e., there are computer, component and factory modules. The CMieux agent distinguishes between a high-level strategy module and the low-level forecast and scheduling modules, apart from the procurement and bidding modules. Finally, the modules of the Mertacor agent are mainly based on main facets of the TAC SCM game. All architectures of the discussed agents have at least some resemblance to the MinneTAC architecture.

The agents not only have different modules, but they also differ in the way decisions are made. The TacTex agent is based on machine learning algorithms to learn from historical (market) data, whereas the DeepMaize agent mainly relies on empirical game-theoretic analyses to be able to make predictions. The PhantAgent is characterized by the fact that it does not use complicated algorithms throughout the agent, but that it uses simple heuristics and assumptions instead. One thing that distinguishes the CMieux agent from other agents, is that this trading agent continuously re-evaluates

its (low-level and high-level) strategies, whereas a lot of other participating trading agents do not perform certain actions continuously, but only once in a while. The Mertacor agent behaves similar to PhantAgent and CMieux.

2.5 Related Work Summary

In this chapter, we have discussed the main specifications of the TAC SCM game. In this game, six artificial trading agents compete on a simulated computer market for 220 days. Their tasks are to compose computers out of procured components and to sell these products to customers. The basic concepts of the game are customers, suppliers, and manufacturers (i.e., trading agents) together with their factories. The game supplies each of these concepts – except for the trading agents – and generates requests. Also, a market report comes out every twenty game days and other functionality such as banking is handled by the game as well.

Furthermore, we have elaborated on the MinneTAC trading agent in detail. Especially its regime model, which is to be altered in this research, has been discussed extensively. The regime model is currently based on sales information. Regimes are identified offline, after which identification and prediction can be done online. These predictions are used for price trend prediction.

Also, some general improvements to the regime model have been suggested. Because of the improved accuracy in identifying and determining regimes when applying the methods elaborated on, the MinneTAC trading agent can benefit from implementing fuzzy sets and fuzzy `C-Means` clustering. Multiple regimes can hold at once, which is more realistic, since regimes transition gradually.

Finally, the architecture and decision making processes of a few competitor agents have been discussed shortly, in order to illustrate the differences and similarities between the competitors and the MinneTAC agent. Although a wide variety of architectures and decision making methods are used, most of the agents show at least some resemblances to the MinneTAC agent.

Chapter 3

Identifying Promising Procurement Variables

In order to be able to research improvements on MinneTAC's current methods for regime identification and also regime prediction, we need – apart from the current required variables such as the normalized mean sales price – some procurement variables which are directly or indirectly dependent on the agent's behavior. These procurement variables are to be incorporated into the current methods. This chapter continues with identifying potentially good variables extracted from the procurement data available in the data set described in section 3.2, which are evaluated using `MATLAB` to apply feature selection.

3.1 Feature Selection

In our research, we need to select the most relevant procurement variables from our data set, or in other words, the variables which have the highest relevance to the target variable. This process is referred to as feature selection, variable selection, or feature extraction. About this topic, a lot of research has been published. This survey discusses several methods of feature selection briefly, based on an extensive book about the foundations and applications of feature selection, written by Guyon et al. [32], as well as an article on this topic by Guyon and Elisseeff [33].

According to Guyon et al., variable selection can be done in many different ways. One can divide the approaches into filter and wrapper algorithms. Wrappers analyze the variable importance by running the model of subject in all kinds of configurations and by evaluating it with some metric and therefore could be computationally intensive, while filters do not run the model in a large number of configurations. For filters, a distinction can be made between information theory-based, correlation-based, and decision tree-based approaches. One can divide wrappers into algorithms designed to

give optimal results, sequential selection algorithms, and stochastic search algorithms.

Many filters that use information theoretic criteria rely on empirical estimates of the mutual information between each variable and the target [33] by using their probability densities. It is harder to apply these approaches to continuous variables than to discrete variables due to the fact that with continuous variables, the densities cannot be estimated from frequency counts, and therefore, Guyon et al. advise to discretize continuous variables when applying information theory-based approaches like the Kullback-Leibler divergence (information gain). This should be taken into account if we decide to apply certain approaches.

For correlation-based variable selection algorithms, measures like (Pearson) correlation and variance are used, as well as for example the T-test. Correlation is used to measure dependencies, and one can estimate the predictive power of certain variables with the error rate.

One final filter approach is the usage of decision tree-based algorithms, such as random forests or even only a decision tree. These algorithms can be evaluated using measures like entropy, separability, and Gini-index. Sometimes, decision tree-based approaches are classified as a third main variable selection approach; the embedded methods [33]. This is due to the fact that these methods are embedded in the model to be evaluated.

As stated earlier, wrapper approaches could be computationally intensive and there are several different types of wrapper approaches. First of all, some algorithms are useful when one wants to obtain optimal results and the computation time is less important, such as the exhaustive search and the branch and bound algorithms. Furthermore, wrappers are most likely to be based on machine learning techniques, such as genetic algorithms and simulated annealing.

In their papers, Andrews et al. [34, 35] analyze the relevance of some sales and procurement variables in the TAC SCM game. Andrews et al. analyze how much certain variables can predict the winner of a TAC SCM game (i.e., the agent which has the largest bank account when the game ends). In their approach, they use the information gain metric to capture the amount of information gained about an agent's performance when knowing its value for some feature. The metric does not indicate which variables are responsible for a better performance, but do indicate which variables differentiate the winning agents. Correlation is not causation, however, and thus all that can be concluded is which variables are worth further analysis.

This is exactly what we need in our research, since we only need to know which procurement variables most likely could improve MinneTAC's regime identification (and prediction), just to narrow our scope. Wrapper (and also embedded) methods are not preferred, since implementing changes and running games is time consuming. Filter methods are less time consuming than wrapper methods and are relatively easy to perform. Most likely, there

are better methods to perform feature selection, which perhaps might not even need the discretization of variables, but, as stated by Andrews et al., the information gain is a general metric which gives us some valuable insight in the (variables of the) TAC SCM game, and therefore – also due to time constraints of this thesis – we use this metric as well. The details of the information gain metric are described in section 3.4.

3.2 Data Set

Before we can continue with identifying promising procurement variables, we need to introduce a data set, which contains historical data on games. This data set can not only be used for the identification of promising procurement variables, but can also be used for experiments later on.

The MinneTAC software for analyzing server logs (currently in development at revision 5403) is able to extract useful and valuable data from the server logs and converts it into an analyzable format, which can be used for research on improvements to the MinneTAC agent, for instance for measuring the performance of changes made in the agent. Our research does not only require some general game data, but also regime-related information about sales as well as procurement to be stored. In order to fulfill this need, we have updated the MinneTAC log software and we have created a couple of scripts to interact with the software, which filter the desired information from the game logs stored on the game servers and convert this into a readable `MATLAB` [36] format. With this data, it is possible to perform different kind of analyses, such as determining statistical values like minimum, maximum and mean, or to deduce new data. Some data might not be needed in our current research, but is still suitable for analyses in future research projects regarding the TAC SCM game and/or the MinneTAC agent, in particular procurement-related research.

This section continues with discussing the different types of data stored in the data set: general data, sales data, and procurement data are discussed in sections 3.2.1, 3.2.2, and 3.2.3. Since the trading agents in a TAC SCM game only have little information readily available, section 3.2.4 elaborates on online and offline data. Also, we have stated that the data set contains data on past games, and therefore we need to specify which games are stored in the data set. This is elaborated on in section 3.2.5. Finally, the contents of the data set are summarized in section 3.2.6.

3.2.1 General Data

Only little general data is stored on each game in the data set (see figure 3.1). This data includes player (i.e., trading agent) data, banking data and market report data. The trading agents competing in the game are identified through their unique names, which are stored in *PlayerList*. Banking data

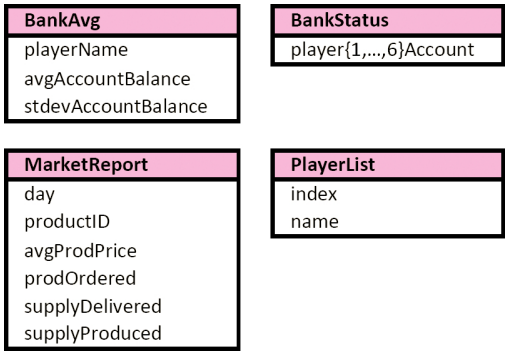


Figure 3.1: Contents of the general data.

on each trading agent, such as statistics on the account balances, is stored in *BankAvg* and *BankStatus*. The largest part of the general data is periodic market report data (*MarketReport*), which includes for example production prices, the number of products ordered, and suppliers' production capacity.

3.2.2 Sales Data

As stated in the introduction of this section, the data set contains regime-related information about sales. Figure 3.2 shows the sales information attributes in the data set.

All offers made to customers and all orders made by customers are stored in *AllOffers* and *AllOrders* respectively. Data is generated for the analysis of different market segments. Several offer and order attributes are stored, such as quantities, prices, reserve prices, lead times, etcetera. In the TAC SCM game, the reserve price is equal to the maximum price one is willing to pay. Because the negotiation between agents and customers is modeled using requests for quotation (RFQ), several RFQ-related attributes are stored as well in *RFQCount* and *RFQs*. Some related information about the total demand for a product and the total number of orders on an arbitrary day is included in *Demand* and *Orders*.

Not only demand, offer, and order information can be useful as sales information, but also data on deliveries, production cycles needed and finished inventories. This data is stored in *Deliveries*, *ProdCycle*, and *FinishedInv*.

3.2.3 Procurement Data

All procurement information stored in our data set contains identification numbers of suppliers (*suppID*), since all procurement-related actions and information flows involve suppliers. Most of the data also contains identification numbers of the manufacturers (trading agents) involved, which are labeled with *manID*. Some data is only applicable to suppliers and have no

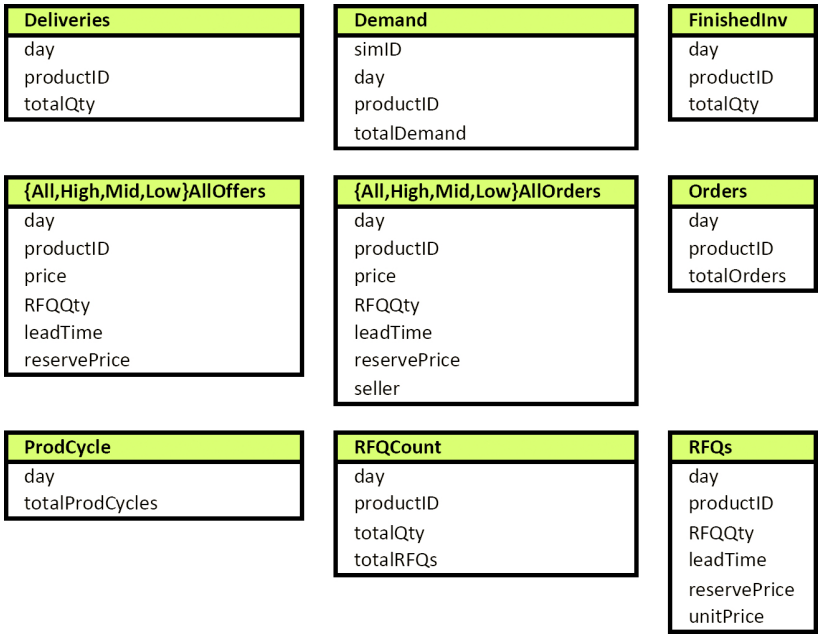


Figure 3.2: Contents of the sales data.

direct link to trading agents. Figure 3.3 visualizes the available procurement-related `MATLAB` files.

First, we discuss the data concerning both suppliers and manufacturers. The daily order ratios for components per manufacturer and supplier are stored in *OrderRatio*. These ratios, which are calculated daily, are defined as the total ordered quantity of a component on an arbitrary day divided by its total offered quantity. Also, all component orders and offers placed in a game are stored in the data set, giving the user insight in (among others) quantities, prices, and the type of orders and offers. This data is stored in *SuppOrders* and *SuppOffers*. Other demand-related data on components in the TAC SCM game can be found in *SuppDemand* and *SuppRFQs*, which contain information about the total daily demand of components per manufacturer, supplier and component, and about requests for quotations respectively. The data stored in *Reputation* represents the daily manufacturer's reputation with a supplier per manufacturer and supplier.

The data set also contains other information about suppliers; more specifically, about each supplier's two production lines. First of all, the capacity of each supplier is recorded in *SuppCap*. Furthermore, quantities are stored per supplier per production line (*SuppQty*), as well as the inventory levels (*SuppInv*). At the moment, the MinneTAC software for analyzing server logs (revision 5403), is not able to identify supplier prices due to persistent bugs and therefore, these prices are not included in the data set.

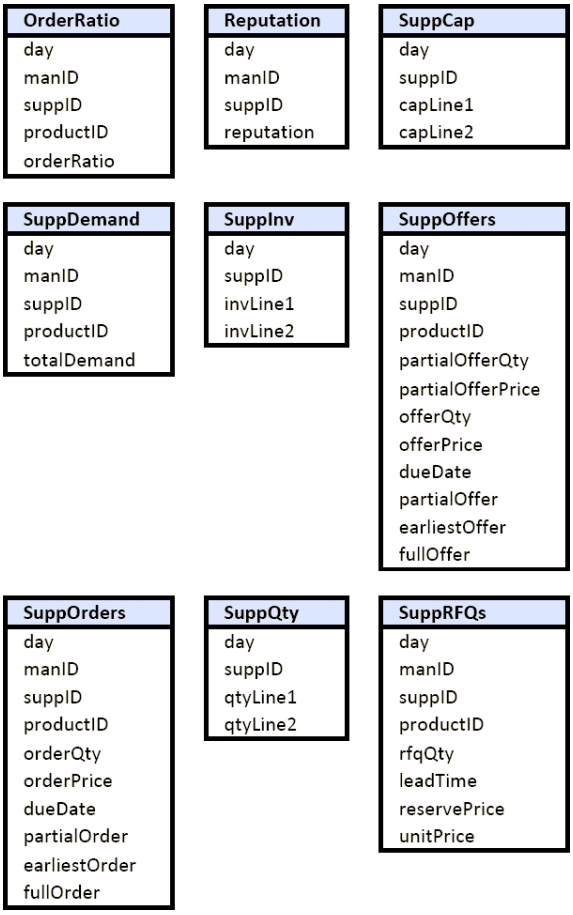


Figure 3.3: Contents of the procurement data.

Note that most of the data in the data set is bound to be a positive, real number. Occasionally, we use ones and zeros to denote properties (e.g., orders and offers, which can either be partial, earliest, or full), representing true and false, respectively.

3.2.4 Online and Offline Data

The data described in the previous sections is not always available to a trading agent during a TAC SCM game. In other words, one can distinguish between online and offline data in this data set. An environment similar to the TAC SCM game in which every player (link in the supply chain) would have perfect knowledge of the market conditions and all other players is very unrealistic and highly doubtful. We define online data as the available information to a trading agent which is visible during a game and thus the only information directly useable for identification and prediction of

regimes, the results of which can subsequently be incorporated into all kinds of decision making processes regarding, for instance, sales and procurement.

The fact that only a small part of the information is available online, does not mean this online information contains the best identifiers for (for instance) market conditions. It is possible that offline information is more valuable. However, if one wants to use offline information as input for predicting models, online approximators should be developed to estimate the desired data, since it is clearly not possible to use offline data online.

Certain (sales related) data is always available to the agent during a TAC SCM game [13]. First of all, the agent knows the entire history of its own received RFQs and orders. Furthermore, each day the agent receives for each PC type the minimum and maximum price of the previous day. Also, a market report is available to the agent, which is generated once every twenty days, containing aggregated information about the past twenty days. For each type of PC the request volume, order volume, and average price are stored.

3.2.5 Historical Game Data

As mentioned in the introduction of this section, as well as in section 1.3, the research which is to be done requires (testing with) a data set containing historical game data. The data set to be used contains data on a selection of the TAC SCM 2007 and 2008 Finals and Semi-Finals games run on the SICS tac3 [37] and tac5 [38] servers and the UMN tac01 [39] and tac02 [40] servers. This is the latest data available, which is likely to contain game situations similar to upcoming TAC SCM games. Older games have very different characteristics, because of the presence of other and more basic trading agents. By using data from 2007 and 2008, we minimize the risk of training and testing on non-representative data. Table 3.1 shows the identification numbers of the games stored in the data set, which is divided into a training set and a test set to be used for experiments later on. The games are grouped by year, game type, and server.

The games in the data set are ordered by game identification numbers. Therefore, the first six games of (for instance) the training set are TAC SCM 2008 Semi-Finals games run on the tac02 server at the UMN, whereas the last six games are the 2007 Semi-Finals.

Year	Type	Server	ID (train)	ID (test)
2007	Semi-Finals	SICS tac5	9323-9327	9321,9322,9328
2007	Finals	SICS tac3	7308-7312	7306,7307,7313
2008	Semi-Finals	UMN tac02	763-768	761,762,769
2008	Finals	UMN tac01	794-799	792,793,800

Table 3.1: Games stored in the data set.

3.2.6 Data Set Summary

The main contents of the data set to be used in this research – which contains historical game data from games introduced in section 3.2.5 – are summarized in table 3.2. This table displays all parts of the data set that are discussed in sections 3.2.1, 3.2.2, and 3.2.3. A short description of the elements and/or information inside is added for each part of the data set.

We have seen that the data set is divided into general data, sales data, and procurement data. General data only contains little information about players, banking and market reports. Sales-related information stored in the data set contains information mostly about offers, orders, and requests for quotation. Finally, more or less the same data is stored for the procurement side of the TAC SCM market.

3.3 Identifying Procurement Variables

Now that we have defined a data set, we can continue with identifying procurement variables. Since this research is one of the first steps in adding procurement information to MinneTAC's regime model, we strive to create variables with low complexity. This will make it easier to evaluate which procurement information might be of use in improving both the current regime identification and prediction. Also, the variables created have to be easily implementable in the current methods, which narrows down the number of possible procurement variables.

In our data set, each table containing procurement information related to manufacturers (see figure 3.3) can be summarized in one or two variables, which represent daily information, decomposed into products or product groups (market segments) where applicable. The daily information stored inside the variable is a result of a single value returning metric, such as the minimum, maximum, or average value of a range of data. Because the agent has a limited view on the market and its players, it is plausible to analyze variables which contain information the agent is able to store itself, and thus all variables should only contain information about the agent. Also, it is good to analyze variables which can be calculated on a manufacturer basis, keeping in mind the calculation of the information gain as explained in section 3.4. This means *SuppCap*, *SuppInv*, and *SuppQty* are not able to return information in a usable format, because they do not have a field containing manufacturer identification numbers.

In this thesis, the data set which is being used is not suitable to be used for variables directly, because – for instance – more than one value of a certain point of interest exists on an arbitrary day for a specific manufacturer and optionally a product or component. We can cope with this problem by means of averaging data, so that only one value remains. Also, it is possi-

Title	Description
BankAvg	Average bank account balances with standard deviation per player
BankStatus	Daily bank account balances per player
MarketReport	Market report for all products and components
PlayerList	List of players
Deliveries	Total delivered quantities per product or component per day
Demand	Total demand per product or component per day
FinishedInv	Total finished inventories per product per day
AllOffers	All offers per day
AllOrders	All orders per day
Orders	Total orders per product per day
ProdCycle	Production cycles needed per day
RFQCount	Number of RFQs and total requested quantity per item per day
RFQs	All RFQs for products per day
OrderRatio	Daily order ratio for components per manufacturer and supplier
Reputation	Daily manufacturer's reputation with a supplier per manufacturer and supplier
SuppCap	Daily supplier capacity per production line
SuppDemand	Daily total demands for components per manufacturer and supplier
SuppInv	Daily supplier inventory per production line
SuppOffers	Daily offers for components per manufacturer and supplier
SuppOrders	Daily orders for components per manufacturer and supplier
SuppQty	Daily supplier quantity per production line
SuppRFQs	Daily RFQs for components per manufacturer and supplier

Table 3.2: Contents of the data set.

ble to calculate a maximum and minimum value, but this makes calculations later on more complex. Although averaging data causes a loss of information, we choose to average the data to make it usable for creating variables for the regime model, because this thesis is a first step into adding procurement information and we want to demonstrate only the effect of the addition of procurement information to the regime model.

Not all tables in the data set can be processed the same way. Three different types of variables can be distinguished. The first type of variable, to which we refer to as $x_{d-1,m,g}$, is constructed in a way that it returns a single value for every day's (d) preceding day ($d-1$), manufacturer m, and product or product group g, and is applicable to all data set tables which contain columns for manufacturers, suppliers, and components. An example of a variable of this type is the procurement offer price. To extract a variable from a table containing values for x, for every manufacturer the value of x is averaged over all suppliers and components. In mathematical terms, the calculation of $x_{d-1,m,g}$ is:

$$x_{d-1,m,g} = \frac{\sum_{s=1}^{\text{numS}} \sum_{c=1}^{\text{numC}_g} x_{d-1,m,g,s,c}}{\text{numX}_{d-1,m,g}}. \tag{3.1}$$

The average is calculated by means of a counting process using numS and numC_g as the number of suppliers and components (per product or market segment), respectively. The number of entries of x for product (group) g on day $d-1$ for manufacturer m is defined as $\text{numX}_{d-1,m,g}$ and $x_{d-1,m,g,s,c}$ represents the value of x on day $d-1$ for component c of product g with manufacturer m and supplier s. Using (3.1), daily average values for the order ratio, quantities and prices of supplier offers and orders, and requests for quotation – again both their quantities and (reserve) prices, but also lead times – can be calculated on a product basis by substituting x and numX in the equation.

Because we would like the variable to include some information about other preceding days as well, so that it represents a trend instead of an event, we apply an exponential smoother to the variable. The smoothed variable $\widetilde{x}_{d-1,m,g}$ is calculated as shown in (3.2), where we set γ to a value of 0.5:

$$\widetilde{x}_{d-1,m,g} = \gamma \cdot x_{d-1,m,g} + (1-\gamma) \cdot \widetilde{x}_{d-2,m,g}. \tag{3.2}$$

Smoothing is done by taking a certain percentage (fifty percent in our case) of yesterday's value of x. Then, the remaining percentage is taken of the previous value of variable x, i.e., the day before yesterday's value, after which both values are added up. This is a less complex way of smoothing than we apply for the normalized mean sales price, but it still smoothes out the possible volatility of the variable.

The second variable type is applicable to the same tables as the first type, but differs from the first type because it does not use products or market

segments, but individual components. Thus, offer prices are a good example of this variable type. The values for the variable are typically calculated by applying

$$y_{d-1,m,c} = \frac{\sum_{s=1}^{\text{numS}} y_{d-1,m,c,s}}{\text{numY}_{d-1,m,c}}. \tag{3.3}$$

In this equation, $y_{d-1,m,c}$ represents the value of y on day $d-1$ with manufacturer m and component c. All values for y with all suppliers (denoted by s) are summed and divided by the number of entries found in the log on day $d-1$ with manufacturer m and component c, denoted by $\text{numY}_{d-1,m,c}$. Again, smoothing is applied to the variable using an γ of 0.5 as shown in (3.4), resulting in $\widetilde{y}_{d-1,m,c}$:

$$\widetilde{y}_{d-1,m,c} = \gamma \cdot y_{d-1,m,c} + (1-\gamma) \cdot \widetilde{y}_{d-2,m,c}. \tag{3.4}$$

Tables in the data set which lack a column with product identification numbers, but have columns for manufacturers and suppliers need to be averaged in a different way compared to (3.2) and (3.4). The third and final variable type can be denoted as

$$z_{d-1,m} = \frac{\sum_{s=1}^{\text{numS}} z_{d-1,m,s}}{\text{numS}}, \tag{3.5}$$

$$\widetilde{z}_{d-1,m} = \gamma \cdot z_{d-1,m} + (1-\gamma) \cdot \widetilde{z}_{d-2,m}, \tag{3.6}$$

in which $z_{d-1,m}$ is the average value of z on day $d-1$ with manufacturer m. Smoothing is done the same way as for (3.1) and (3.3)

This variable type does not involve products or product groups or components, but only averages the sum of all values of z on day d with manufacturer m for all suppliers s by dividing the latter sum by the number of suppliers, numS. It is possible to apply (3.5) and (3.6) to the manufacturers' reputations with the suppliers, by substituting z.

Type 1 ($\widetilde{x}_{d-1,m,g}$)	Type 2 ($\widetilde{y}_{d-1,m,c}$)	Type 3 ($\widetilde{z}_{d-1,m}$)
$\text{orderRatio}_{d-1,m,g}$	$\text{orderRatio}_{d-1,m,c}$	$\text{reputation}_{d-1,m}$
$\text{demand}_{d-1,m,g}$	$\text{demand}_{d-1,m,c}$	
$\text{offerQty}_{d-1,m,g}$	$\text{offerQty}_{d-1,m,c}$	
$\text{offerPrice}_{d-1,m,g}$	$\text{offerPrice}_{d-1,m,c}$	
$\text{orderQty}_{d-1,m,g}$	$\text{orderQty}_{d-1,m,c}$	
$\text{orderPrice}_{d-1,m,g}$	$\text{orderPrice}_{d-1,m,c}$	
$\text{RFQQty}_{d-1,m,g}$	$\text{RFQQty}_{d-1,m,c}$	
$\text{RFQResPrice}_{d-1,m,g}$	$\text{RFQResPrice}_{d-1,m,c}$	
$\text{RFQLeadTime}_{d-1,m,g}$	$\text{RFQLeadTime}_{d-1,m,c}$	

Table 3.3: Variables extracted from the data set.

Table 3.3 gives an overview of all variables that can be extracted from the data set, using the three equations discussed in this section. As stated earlier, each manufacturer-based table with procurement information in our data set offers data to convert into one or two variables. For an overview of all available procurement tables in the data set, see section 3.2.3. The tables used for the variables of the first and second type are *OrderRatio*, *SuppDemand*, *SuppOffers*, *SuppOrders*, and *SuppRFQs*. The *Reputation* table is used for generating a variable of the third type.

3.4 Usage of the Information Gain for Evaluating Procurement Variables

As is the case in [34] and [35], we need to evaluate which feature (variable) helps to identify or predict a certain target. However, Andrews et al. try to identify which features differentiate a winning agent from the other agents, whereas we try to identify which features add information to the current process of regime identification and prediction. Therefore, our approach will be somewhat different from the approach proposed by Andrews et al.

Information about these valuable features could possibly mean a difference, but also changing complete strategies of the trading agent (based on these findings) could have a positive effect on the agent's performance. Due to the fact that the decisions of the MinneTAC trading agent are based on the identification and prediction of economic regimes, it is plausible to add typical (differentiating) procurement information to the regime-related algorithms, in the hope to improve the results of the agent by taking into account particular procurement-related aspects. We restrict the usage of our findings to only incorporating new information into the regime model, since this is the scope of our research and thus no implementations related to strategy changes are discussed.

In order to evaluate the variables (or features), the information gain metric is applied to game data. For now, we experiment on one variable at a time, but experiments on combinations of variables can perhaps be done in future. In our analyses, we try to determine whether the first variable type is better than the second type or the other way around. Furthermore, the best performing variables (i.e., the variables with the highest information gain) are selected.

The information gain is an entropy-based metric which indicates how much better we can predict a certain target by knowing certain features. In our case, the target is the dominant regime and the features are all procurement variables introduced in section 3.3.

We define the target values as the currently identified dominant regimes, because this is the direct result we would like to improve. In [34] and [35], winning a game is selected as target value, but to us, winning a game is

only an indirect consequence of improved regime identification. Because we would like to keep using the same regime labels as are used currently, the currently identified dominant regimes are the closest target value we can get at the moment.

Generally, the metric is used for calculating the information gained on some discrete outcome (in our case the dominant regime) from a discrete attribute (variable). Since the procurement variables introduced in section 3.3 are continuous and the information gain metric needs a discrete attribute, we need to discretize the variables. This is done by applying a partition level of 25, making the partitions small enough to avoid loosing too much information, but large enough to avoid overfitting. Also, the target has to be discrete, and thus we will not use the regime probabilities, but the dominant regime only.

As stated earlier, the information gain is an entropy-based metric. According to Mitchell [41], the entropy is a commonly used measure in information theory, which characterizes the purity of an arbitrary collection of examples. Let W be a collection of game results, numW the number of possible values of W (in our research this value will be five), and $P(w)$ the probability that W takes on value w. Assuming a uniform probability distribution, the latter probability is equal to the proportion of W belonging to class w. The entropy of a collection of game results, entropy (W), is defined as

$$\text{entropy}(W) = \sum_{w=1}^{\text{numW}} -P(w) \log_2 P(w). \tag{3.7}$$

Now let V be an attribute (procurement variable), numV the number of possible values of V (which is, as stated earlier in this section, 25), $P(v)$ the probability that V takes on value v, and $P(w|v)$ the probability that W takes on value w, given v. The entropy of a collection of game results W given an attribute V, entropy $(W|V)$, is defined as

$$\text{entropy}(W|V) = \sum_{v=1}^{\text{numV}} P(v) \left(\sum_{w=1}^{\text{numW}} -P(w|v) \log_2 P(w|v) \right). \tag{3.8}$$

Using (3.7) and (3.8), the information gained on outcome W from attribute V, gain (W, V), can be calculated using

$$\text{gain}(W, V) = \text{entropy}(W) - \text{entropy}(W|V). \tag{3.9}$$

Here, the entropy of a collection of game results W given an attribute V is subtracted from the entropy of W.

The information gain metric as explained above will be applied to variables gathered from game logs from several games from our data set. We define a training set which contains 22 games, representing both the TAC SCM 2007 and 2008 Finals and Semi-Finals games. An overview of the

identification numbers of the games stored in our training set is given in table 3.1 of this chapter.

Section 3.5 discusses the results of applying the information gain metric to each defined procurement variable on the aggregation of data on all games stored in the data set. In our analyses, we use identified daily (dominant) regimes on a per product basis as target values, being based on the regime probabilities returned by models trained on TAC SCM 2005 Semi-Finals and Finals [42, 9] with five regime clusters (`K-Means`), twenty-five or sixteen Gaussians (for individual products and product groups, respectively), fixed means, and fixed variances. For each variable q, information gains are calculated per game o per manufacturer m, which are averaged afterwards:

$$\text{gain}\,(W)_q = \frac{\sum_{o=1}^{\text{numO}} \sum_{m=1}^{\text{numM}} \text{gain}\,(W)_{q,o,m}}{\text{numO} + \text{numM}}. \tag{3.10}$$

For the first and second variable types, individual information gains are calculated for each product (group) or component, after which an average information gain is calculated. Since daily (dominant) regimes are defined for products and market segments, and thus are based on normalized mean prices for these products and segments, an extra step has to be added to the calculation methods of information gains for both component-based variables and variables neither based on components nor on products.

For the second variable type, we need to find a way to associate the dominant regimes with components in order to be able to calculate information gains for component-based variables. Therefore, for each component, the information gains of an arbitrary component-based variable are calculated for each product or market segment which uses the component. Then, an average information gain can be calculated over the gains of the associated products and market segments, which is considered as the information gain for the component. As for the third variable type, information gains for each product and market segments are averaged.

As stated earlier in this chapter, applying the information gain metric to the proposed procurement variables helps us to identify potentially good variables to identify and predict economic regimes in the TAC SCM game, which suggests further research on implementing these variables in MinneTAC's regime model is feasible. We continue our research with the procurement variables with the highest information gains. These variables are most likely to improve the overall performance of the MinneTAC agent, if they are implemented correctly in its regime model.

3.5 Information Gain Results

Applying the information gain metric to each variable (type) as explained in section 3.4 yields the information gains as shown in table 3.4. Next to the

calculated information gains (gain, (3.9)), the highest achievable information gains are shown (maxGain, (3.11)), accompanied with a percentage which expresses how much of this maximum is achieved (relGain, (3.13)).

Variable	gain	maxGain	relGain
$\text{offerPrice}_{d-1,m,g}$	0.7393	1.7054	43.3518%
$\text{orderPrice}_{d-1,m,c}$	0.7104	1.7046	41.6737%
$\text{offerPrice}_{d-1,m,c}$	0.6148	1.7046	36.0647%
$\text{orderPrice}_{d-1,m,g}$	0.5400	1.7054	31.6633%
$\text{RFQLeadTime}_{d-1,m,g}$	0.5106	1.7054	29.9422%
$\text{orderQty}_{d-1,m,c}$	0.5049	1.7046	29.6188%
$\text{RFQResPrice}_{d-1,m,g}$	0.4909	1.7054	28.7820%
$\text{RFQLeadTime}_{d-1,m,c}$	0.4622	1.7046	27.1128%
$\text{orderRatio}_{d-1,m,g}$	0.4555	1.7054	26.7100%
$\text{RFQResPrice}_{d-1,m,c}$	0.4521	1.7046	26.5238%
$\text{orderQty}_{d-1,m,g}$	0.4310	1.7054	25.2743%
$\text{orderRatio}_{d-1,m,c}$	0.4161	1.7045	24.4091%
$\text{RFQQty}_{d-1,m,g}$	0.3901	1.7054	22.8747%
$\text{demand}_{d-1,m,g}$	0.3833	1.7054	22.4772%
$\text{RFQQty}_{d-1,m,c}$	0.3671	1.7046	21.5333%
$\text{demand}_{d-1,m,c}$	0.3598	1.7046	21.1088%
$\text{offerQty}_{d-1,m,g}$	0.3174	1.7054	18.6090%
$\text{offerQty}_{d-1,m,c}$	0.2932	1.7046	17.1993%
$\text{reputation}_{d-1,m}$	0.0571	1.7063	3.3452%

Table 3.4: Information gains for each variable.

In order to be able to value the information gains calculated for each variable properly, it should be noted that the highest achievable information gain for each variable in our training set W, $\text{maxGain}(W)$, is equal to its entropy, as shown in (3.11):

$$\text{maxGain}(W) = \text{entropy}(W). \tag{3.11}$$

The games stored in W sometimes contain failing agents, and the games also vary in the events happened, resulting in different data for each game (and manufacturer). Generally, the maximum information gain is about 1.7050 on average.

Equation (3.12) shows how the maximum information gain is calculated for W, which represents the $\text{orderRatio}_{d-1,m,g}$ of product 12 for the manufacturer with identification number 10 in the first game of our data set. The entropy (maximum information gain) of this particular subset yields approximately 2.1618. The subset contains 219 values of the variable, of which 42 values are associated with the first regime, 18 with the second regime, etcetera. The maximum information gain is higher than the one shown

in table 3.4, which can be explained by the fact that the table only contains average maximum information gains, whereas our example is the maximum information gain for one specific combination of game, manufacturer, and product.

$$\begin{aligned}\text{maxGain}\,(W) = &-\left(\frac{42}{219}\log_2\frac{42}{219}\right)-\left(\frac{18}{219}\log_2\frac{18}{219}\right)\\ &-\left(\frac{26}{219}\log_2\frac{26}{219}\right)-\left(\frac{64}{219}\log_2\frac{64}{219}\right)\\ &-\left(\frac{69}{219}\log_2\frac{69}{219}\right)\\ \approx\ &2.1618. \end{aligned} \tag{3.12}$$

Variables with high information gains (approaching the maximum information gains) are more likely to perform well when implemented in the regime model. Variables are rated based on their relative information gain, i.e., how much of their maximum information gain is achieved. Equation (3.13) shows the calculation of the relative information gain for subset W, $\text{relGain}\,(W)$:

$$\text{relGain}\,(W) = \frac{\text{gain}\,(W)}{\text{maxGain}\,(W)} \times 100\%. \tag{3.13}$$

By looking at table 3.4, but especially at figure 3.4 (which shows the relative information gains of each variable), a few observations can be made. First of all, exponentially smoothed product-based mean procurement-side offer prices of the preceding day yield the highest (relative) information gain. Furthermore, some of the variables of the first and second type seem to return good (relative) information gains, whereas the third variable type falls behind.

Two component-based variables (i.e., mean procurement offer prices and mean order prices) and one product-based variable (i.e., mean offer prices) have information gains with a value clearly higher than the other variables, and distinguish themselves from the other procurement variables. To illustrate the magnitude of (out) performance, the graph shows that the information gains of the top three procurement variables are higher than one standard deviation above the mean of the calculated information gains.

Roughly, variables of the first type outperform their equivalents of the second variable type a little in most cases. However, order quantities and prices on a component basis perform notably better than their product-based equivalents. We cannot conclude that product-based variables outperform component-based variables, since product-based variables outperform their equivalents mostly (seven times against two times), but the average amount by which product-based variables outperform the procurement

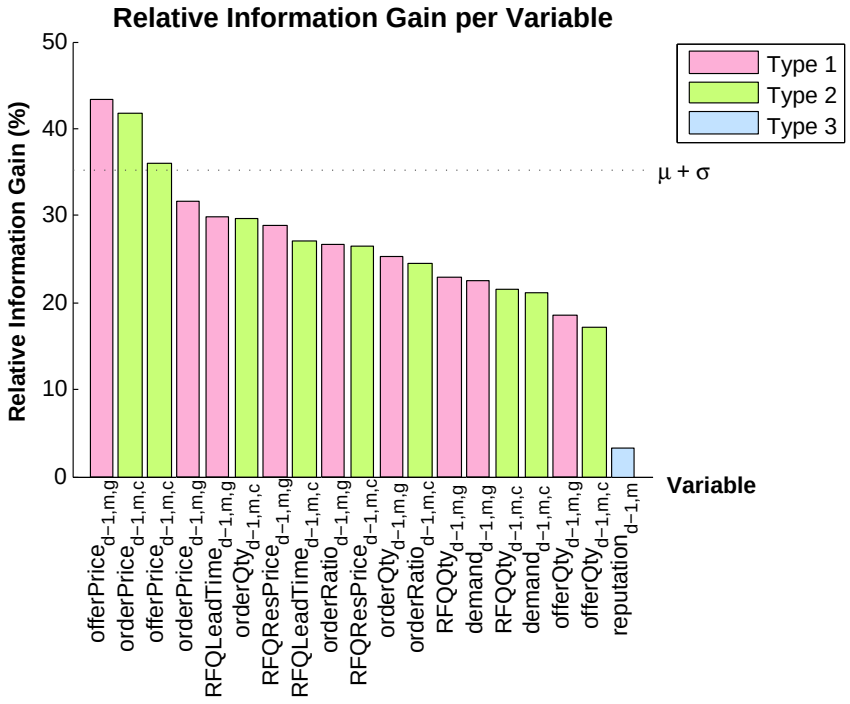


Figure 3.4: Graphical representation of the relative information gains for each procurement variable.

variables based on components is smaller than the average amount by which the component-based variables outperform their equivalents. These findings are supported by table 3.5. The total percentages represent the sum of the percentages by which a variable type outperforms the other variable type. The average percentages are calculated by dividing the total percentages by the number of times a variable type outperforms the other variable type.

As demonstrated, especially the exponentially smoothed product-based mean procurement offer prices variable (offerPrice$_{d-1,m,g}$) returns good results in the information gain analyses, followed by component-based order prices and offer prices (orderPrice$_{d-1,m,c}$ and offerPrice$_{d-1,m,c}$, respectively).

	Type 1	Type 2
Outperforming	7 times	2 times
Total percentage	69.4898%	48.8045%
Average percentage	9.9271%	24.4022%

Table 3.5: Comparison of the information gains of product-based and component-based variables (types 1 and 2, respectively).

Therefore, one can conclude that, because the information gains of these three variables are notably higher than the gains that have been calculated for the other variables, knowing the offer prices (both component-based and product-based) and order prices (component-based) is most likely to improve the chances of predicting the winning agent for each game.

3.6 Conclusions on the Identification of Promising Procurement Variables

In this chapter, some feature selection techniques have been suggested. Regimes can be determined differently when applying fuzzy set techniques. The feature selection technique which appears to be most suitable, i.e., the information gain, is selected to be used when continuing our research. With the information gain metric, we have determined which procurement variable is most likely to affect the performance of the MinneTAC trading agent.

The results of the information gain analyses on the training set defined in table 3.1 show that mean order prices on a per component basis and product-based and component-based mean procurement-side offer prices are most likely to improve the overall trading agent's performance. Therefore, they should be implemented in the regime model, since the latter model is an important part of the MinneTAC agent as a whole, being the core of daily decision making. The three identified procurement variables only are applicable to the MinneTAC agent, because only limited information is available on the other agents in a TAC SCM game by means of the market report which becomes available every twenty game days, containing little information on competitors and market conditions, which is not of great value anymore after only a few game days. We believe information collected by the MinneTAC agent on order and procurement offer prices is good enough to indicate the market conditions, since the agent's behavior influences the market and is influenced by the market as well.

In order to keep changes to the regime model manageable, only one variable is selected to extend the regime model. Even so, it is not feasible to select more than one variable based on these information gain analyses, because no combinations of variables have been analyzed. Product-based procurement variables mostly outperform their component-based equivalents, so chances are that if a product-based variable is selected, it will generate better improvements to the performance of the agent than component-based variables, even though the latter variables outperform product-based variables to a greater extent than the other way around in case they do outperform their equivalents.

Taking into account the former observations, as well as the observations in section 3.5, we select the overall best performing procurement variable: exponentially smoothed procurement-side offer prices based on products.

The calculation of the variable, to which we will refer to as $\widetilde{\text{op}}_{d-1,m,g}$, is shown in (3.14) and (3.15):

$$\text{op}_{d-1,m,g} = \frac{\sum_{s=1}^{\text{numS}} \sum_{c=1}^{\text{numC}_g} \text{op}_{d-1,m,g,s,c}}{\text{numOp}_{d-1,m,g}}, \tag{3.14}$$

$$\widetilde{\text{op}}_{d-1,m,g} = \gamma \cdot \text{op}_{d-1,m,g} + (1-\gamma) \cdot \widetilde{\text{op}}_{d-2,m,g}. \tag{3.15}$$

In these equations, m only refers to the MinneTAC agent, since the only information readily available is information about the agent itself. As already explained, in the TAC SCM game environment, it is a non-trivial task to collect – or even estimate – data on procurement-related activities of the competitors due to game constraints. The equation is derived from equations (3.1) and (3.2).

We continue our research using yesterday's exponentially smoothed product-based mean procurement offer prices $\widetilde{\text{op}}_{d-1}$. The procurement variable is added to the current regime model, after which the performance is evaluated.

Chapter 4

Altering MinneTAC's Regime Identification and Prediction

In this chapter, we try to find a way to alter MinneTAC's offline regime identification and prediction, while leaving the online regime output intact. In other words, the new regime model with added procurement information should still return the probabilities for each regime. Online identification only embodies matching an arbitrary game situation (per day and product group) to offline identified clusters, referred to as regimes. Therefore, it does not need to be changed a lot, and thus this chapter focuses on altering MinneTAC's offline regime identification. However, regime prediction is mainly done during a game, and thus this chapter also focuses on altering MinneTAC's online regime prediction.

This chapter continues in section 4.1 with elaborating on several aspects which come to our attention when trying to alter MinneTAC's regime model. Section 4.2 discusses a new (or extended) framework for identifying and predicting regimes to be used by the trading agent, and section 4.3 concludes the chapter.

4.1 Considerations

As already explained in section 2.3.2, regimes are currently identified offline by means of clustering posterior probabilities of individual Gaussians (given normalized mean sales-side prices). These regimes are assigned labels by means of correlation studies.

If we want to add procurement information to the regime identification and prediction processes, it is likely that the core of the identification process being the basis of the prediction processes, the Gaussian Mixture Model, has to be altered, resulting in new posterior probabilities and thus in new

clusters. By adding a dimension to the Gaussian Mixture Model, i.e., mean procurement offer prices, new clusters which represent the regimes will have to be identified, making the correctness of the definitions of each current regime doubtable.

After adding a dimension to the Gaussian Mixture Model, not only the normalized mean prices influence the properties of the Gaussians and clusters (i.e., shape and location), but also the mean procurement-side offer prices. Therefore, regimes cannot be defined and labeled purely based on normalized mean prices anymore. New clusters are likely to emerge, since using the procurement dimension possibly gives another separation of data than using the normalized mean price dimension only. When these clusters are projected onto the normalized mean prices, it is possible that they are located elsewhere with respect to the current clusters. The newly identified clusters can then even overlap each other (completely) if a new meaningful dimension is added to the clustering process.

Because the data being the basis of clustering is enriched with procurement information, defining and labeling regimes becomes harder. It is not trivial which (current) regime label should be assigned to which cluster (assuming there are three or five clusters), and more importantly, which behavior should be associated with each cluster. It even is questionable if the current regimes are applicable at all. By assigning the correct labels to the new regimes, the MinneTAC agent is able to make correct decisions, whereas assigning the wrong labels might have disastrous consequences.

To cope with this problem, it is possible to perform correlation studies on the newly identified regime clusters, in order to be able to label them correctly using the current regime labels. These correlation studies, if done properly, enable the MinneTAC agent to use the altered regime identification process without having the need to change any other (related) parts.

Now that we have identified the exact part of the regime model where adding procurement information could effect the regime identification the most, we can create a new framework for regime identification and prediction. This framework is discussed in the next section.

4.2 A New Framework

We demonstrate the updated regime model by extending the equations elaborated on in section 2.3.2 in such a way that they also incorporate mean procurement-side offer prices. Visualizations are based on data from the training set defined in chapter 3 (which is demonstrated in table 3.1 of the latter chapter). We first discuss the extended regime identification process in section 4.2.1, followed by the extensions to the current regime prediction process (section 4.2.2) and regime model applications, which is discussed in section 4.2.3.

4.2.1 Extensions to Regime Identification

The current regime model is based on a Gaussian Mixture Model (GMM) with N Gaussian components which have fixed means and variances. This might lead to good results when fitting a model on one dimension, but after adding a dimension to the model, fixed means and variances might prevent the GMM to reach a good fit. Therefore, we do not constrain the means and variances for now.

As is the case with the current model, we apply the Expectation-Maximization algorithm to determine the Gaussian components of the GMM and their prior probabilities, $P(\zeta_i)$. The Gaussian components are, unlike the components of the current model, based on both np and op. For now, the number of Gaussian components, N, is equal to 3, because this helps visualizing and explaining the main concepts of the model.

Because of the addition of the procurement offer prices (op), we extend the density of the normalized mean sales price, $p(\text{np})$, to the density of the normalized mean price and mean offer price, $p(\text{np} \cap \text{op})$, such that

$$\begin{aligned} p(\text{np} \cap \text{op}) &= \sum_{i=1}^{N} P(\zeta_i)\, p(\text{np}|\zeta_i)\, p(\text{op}|\zeta_i \cap \text{np}) \\ &= \sum_{i=1}^{N} P(\zeta_i)\, p(\text{np} \cap \text{op}|\zeta_i), \quad \forall i = 1, 2, \ldots, N. \end{aligned} \tag{4.1}$$

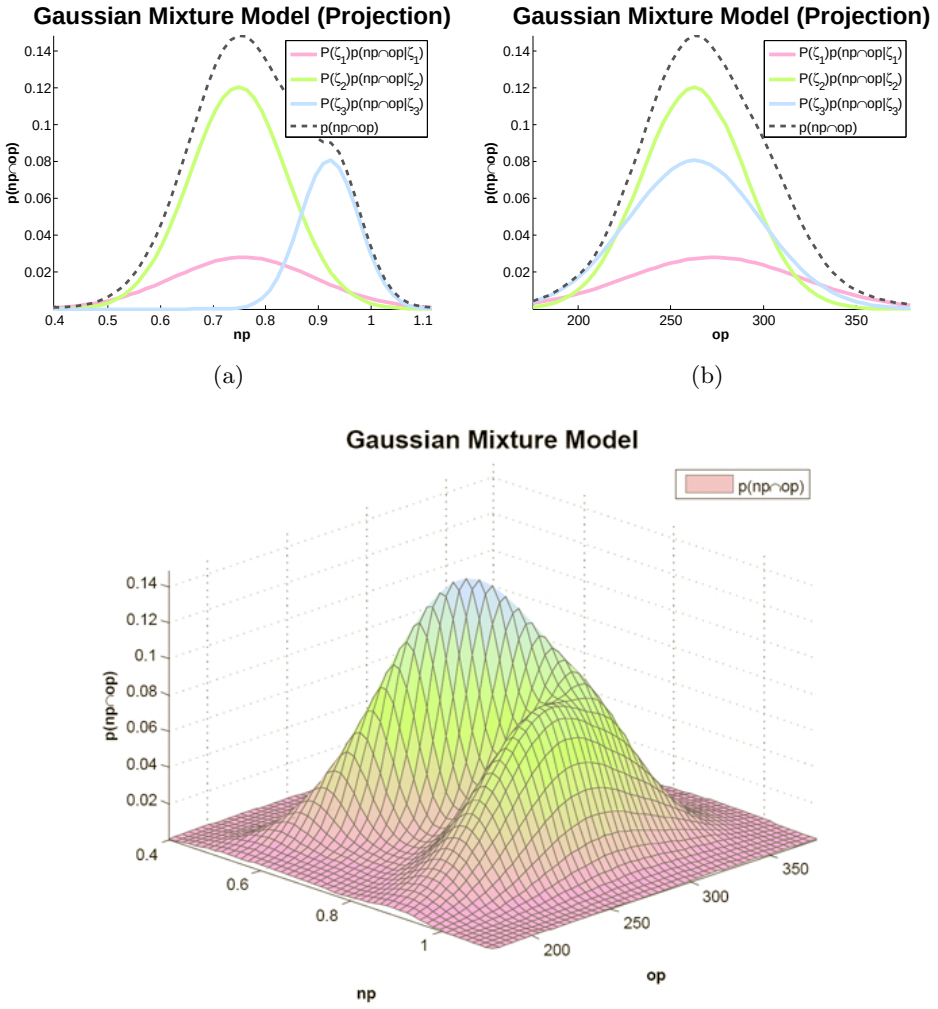
The density is equal to the sum of all Gaussian components $p(\text{np} \cap \text{op}|\zeta_i)$ multiplied by their prior probabilities $P(\zeta_i)$. We define a typical two-dimensional Gaussian component as

$$\begin{aligned} p(\text{np} \cap \text{op}|\zeta_i) &= p\left(\text{np} \cap \text{op}|\mu_{\text{np}_i} \cap \mu_{\text{op}_i} \cap \sigma_{\text{np}_i} \cap \sigma_{\text{op}_i}\right) \\ &= A e^{-\left(\frac{(\text{np}-\mu_{\text{np}_i})}{2\sigma^2_{\text{np}_i}} + \frac{(\text{op}-\mu_{\text{op}_i})}{2\sigma^2_{\text{op}_i}}\right)}, \end{aligned} \tag{4.2}$$

where A is the amplitude of the Gaussian, μ_{np_i} and μ_{op_i} are the means of the i-th Gaussian on the normalized mean price and mean offer price axes, and σ_{np_i} and σ_{op_i} are their respective standard deviations.

Figure 4.1 shows plots of a two-dimensional GMM created using the equations discussed above. The model contains three Gaussian components, which do not have fixed means and variances, and is trained with a maximum of fifteen hundred iterations on data on the low product segment. Figures 4.1(a) and 4.1(b) show projections of the individual Gaussians and the density of the normalized mean sales price and mean offer price onto the axes of both variables. To give a proper understanding of the characteristics of the density, this density is shown as a surface in figure 4.1(c).

The posterior probability for each Gaussian component, $P(\zeta_i|\text{np} \cap \text{op})$, follows from (4.1) after applying Bayes' rule. The resulting posterior probability can be denoted as shown in (4.3).



(a) (b)

(c)

Figure 4.1: A two-dimensional Gaussian Mixture Model on normalized mean (sales) price and mean procurement offer price, using three Gaussian components, which do not have fixed means and variances. This model is trained with a maximum of fifteen hundred iterations on data on the low product segment, using data from the training set defined in table 3.1.

$$P\left(\zeta_i|\text{np} \cap \text{op}\right) = \frac{P\left(\zeta_i\right) p\left(\text{np} \cap \text{op}|\zeta_i\right)}{\sum_{j=1}^{N} P\left(\zeta_j\right) p\left(\text{np} \cap \text{op}|\zeta_j\right)}, \quad \forall j = 1, 2, \ldots, N. \tag{4.3}$$

Equation (4.3) applies for each Gaussian, and thus the vector of posterior probabilities for the two-dimensional Gaussian Mixture Model is equal to the vector described in (4.4):

$$\eta\left(\text{np} \cap \text{op}\right) = \left[P\left(\zeta_1|\text{np} \cap \text{op}\right), P\left(\zeta_2|\text{np} \cap \text{op}\right), \ldots, P\left(\zeta_N|\text{np} \cap \text{op}\right)\right]. \tag{4.4}$$

For each combination of normalized mean prices and procurement offer prices, we can calculate $\eta\left(\text{np} \cap \text{op}\right)$ using the Gaussian Mixture Model we have fit. Clustering the posterior probabilities in M clusters is done using the same algorithm as in the current regime model: `K-Means`. Although fuzzy `C-Means` clustering is more intuitive to use and is more stable in usage, we use the `K-Means` clustering algorithm, because we do not want to change the way regimes are to be interpreted by the agent. Clustering is done in fifteen replicates, using a maximum of one hundred iterations. The squared Euclidean distance measure is used to measure distances to the cluster centers for each data point. Figure 4.2 shows results of applying the `K-Means` clustering algorithm to the GMM we have fit to our data on low-range products with three clusters. A clear separation of clusters is visible.

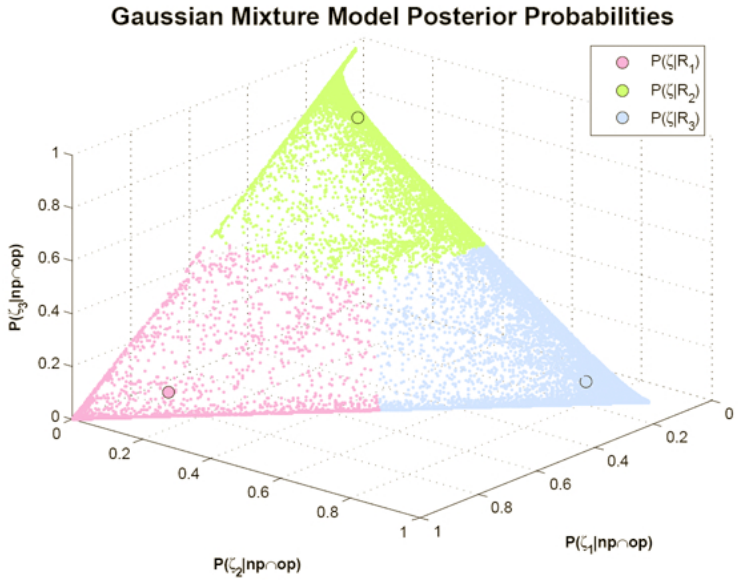


Figure 4.2: Three identified regime clusters within the low product segment after applying fifteen replicates of the `K-Means` algorithm (with a maximum of one hundred iterations) to posterior probabilities of a two-dimensional Gaussian Mixture Model, based on product-related data from the training set defined in table 3.1.

The cluster centers $P(\zeta|R_k)$ correspond to regimes (in our case for products belonging to the low segment), but these clusters do not tell us which cluster represents which regime. Let us not worry about that for now, since this is not important for understanding the framework. Let us assume we know how to assign the proper regime label to each cluster. Then we can rewrite $p(\text{np} \cap \text{op}|\zeta_i)$ in a form that shows the dependence of the normalized sales price and mean procurement offer price on the regime R_k:

$$p(\text{np} \cap \text{op}|R_k) = \sum_{i=1}^{N} P(\zeta_i|R_k)\, p(\text{np} \cap \text{op}|\zeta i), \quad \forall i = 1, 2, \ldots, N. \tag{4.5}$$

In (4.5), $P(\zeta_i|R_k)$ refers to the N by M matrix resulting from the `K-Means` algorithm, and $p(\text{np} \cap \text{op}|\zeta i)$ refers to the individual Gaussians. When applying Bayes' rule, we obtain the probability of regime R_k dependent on the sales and offer prices, as defined in (4.6):

$$P(R_k|\text{np} \cap \text{op}) = \frac{P(R_k)\, p(\text{np} \cap \text{op}|R_k)}{\sum_{j=1}^{M} P(R_j)\, p(\text{np} \cap \text{op}|R_j)}, \quad \forall j = 1, 2, \ldots, M. \tag{4.6}$$

Figure 4.3 shows a plot of the regime probabilities (given normalized sales price and procurement offer price) for products of the low segment, resulting from a Gaussian Mixture Model and clustering its posterior probabilities in three clusters. We observe that each identified regime is dominant for certain combinations of both variables the model is based on, albeit not with a high confidence, since the probabilities of the clusters often are close to each other.

Regime probabilities can be calculated for different combinations of normalized mean prices and procurement-side offer prices. We choose fifty normalized mean prices and fifty mean offer prices and calculate the regime probabilities per cluster for each combination of both variables. Values of a variable are equally distant from each other and range from the minimum value of the variable in the data set to its maximum value. This results in M fifty by fifty matrices containing regime probabilities.

Now that we have defined a new regime model for identifying regimes offline – by adding a procurement variable, i.e., offer prices – we can update the online regime identification. There is no direct need to change the algorithm which is currently used. However, we do need to add the procurement variable identified in chapter 3, so that the online identified regime $\widehat{R}_r$ on an arbitrary game day d is dependent on yesterday's (i.e., $d-1$) exponentially smoothed normalized mean price $\widetilde{\text{np}}_{d-1}$ and yesterday's exponentially smoothed procurement offer price $\widetilde{\text{op}}_{d-1}$. The result is shown in (4.7).

$$\widehat{R}_r \; s.t. \; r = \underset{1 \leq k \leq M}{\text{argmax}} \; \vec{P}(\widehat{R}_k|\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}). \tag{4.7}$$

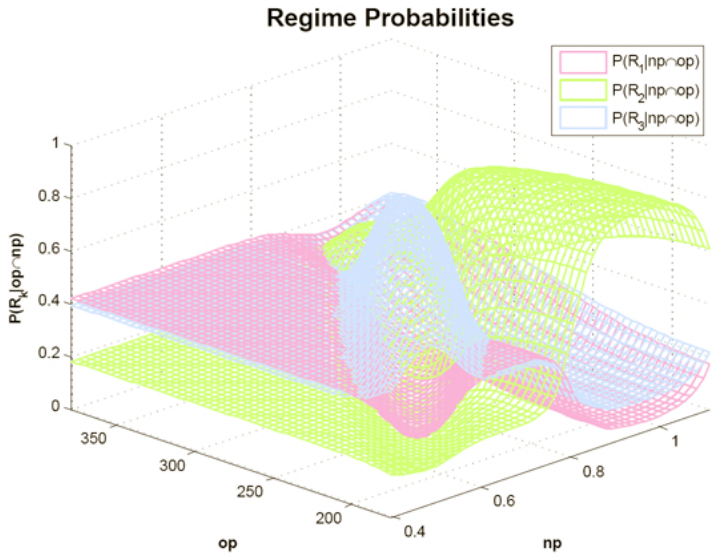


Figure 4.3: Regime probabilities (given normalized price and mean procurement offer price) for products of the low segment, resulting from a Gaussian Mixture Model with two dimensions, of which its posterior probabilities are clustered in three clusters. The probabilities are based on data from the training set defined in table 3.1.

The regime probabilities can be estimated online using each regime's fifty by fifty probability matrix. Instead of the one-dimensional linear interpolation which is currently used, two-dimensional linear interpolation should be used in the new regime model. Determining the dominant regime remains unchanged, and thus the dominant regime is still equal to the regimes with the highest probability.

One can conclude that in general, the regime identification still works similar to the current regime (identification) model. However, a dimension has been added to the Gaussian Mixture Model, causing differently structured probability densities as well as regime clusters. This requires reformulating the entire regime identification model.

4.2.2 Extensions to Regime Prediction

In section 2.3.2, several regime prediction techniques are introduced. This section continues with elaborating on extensions to these techniques. First, regime prediction using an exponential smoother process is discussed. Then, Markov prediction processes are discussed, followed by Markov correction-prediction processes. Recall that each process is most suitable for a specific time span. Exponential smoothing can best be applied to predict today's

regime probabilities, whereas Markov processes are more suitable for short-term and long-term predictions. Again, ensemble prediction is not considered.

Exponential Smoother Process

As stated in section 2.3.2, today's regime probabilities are estimated using an exponential smoother process. This process calculates a trend of a certain variable. In the initial model, this variable represents the exponentially smoothed normalized mean sales price. Then, future values of the variable can be calculated, which can then be used to estimate future regime probabilities.

In order to extend this approach to include procurement information as well, we leave the prediction of the normalized prices n days into the future ($\widetilde{\text{np}}_{d+n}$) as it is. Because of the fact that the procurement variable (i.e., procurement offer prices) represents a mean value and we do not have minimum and maximum, the trend of this variable cannot be calculated similar to the trend of the sales price. Also, different smoothing is applied to procurement offer prices than to sales prices, which means the two Brown linear exponential smoothing components used for calculating the sales price trend are not available for our procurement variable.

We calculate the trend of $\widetilde{\text{op}}_{d-1}$, defined as $\widetilde{\text{tr}}^{\text{op}}_{d-1}$, as shown in (4.8). Here, the trend is equal to the difference between yesterday's (exponentially smoothed) procurement offer price and the price of the day before yesterday:

$$\widetilde{\text{tr}}^{\text{op}}_{d-1} = \widetilde{\text{op}}_{d-1} - \widetilde{\text{op}}_{d-2}. \tag{4.8}$$

Then, future values for n days into the future up to planning horizon h are calculated similar to future values of $\widetilde{\text{np}}$, as shown in (4.9). Here, the calculated trend is added $1+n$ times to the last known value of the procurement offer prices. We express the future values of $\widetilde{\text{op}}$ mathematically as

$$\widetilde{\text{op}}_{d+n} = \widetilde{\text{op}}_{d-1} + (1+n) \cdot \widetilde{\text{tr}}^{\text{op}}_{d-1}, \quad \forall n = 0, 1, \ldots, h. \tag{4.9}$$

Now that we have future values for day $d+n$ of $\widetilde{\text{np}}$ and $\widetilde{\text{op}}$, we can extend the calculation of the regime probabilities for n days into the future. This calculation is an extension to (2.16) and is done as follows from (4.10):

$$P\left(\widehat{R}_k | \widetilde{\text{np}}_{d+n} \cap \widetilde{\text{op}}_{d+n}\right) = \frac{p\left(\widetilde{\text{np}}_{d+n} \cap \widetilde{\text{op}}_{d+n} | \widehat{R}_k\right) P\left(R_k\right)}{\sum_{k=1}^{M} p\left(\widetilde{\text{np}}_{d+n} \cap \widetilde{\text{op}}_{d+n} | \widehat{R}_k\right) P\left(R_k\right)}, \quad \forall k = 1, 2, \ldots, M. \tag{4.10}$$

Here, the density of $\widetilde{\text{np}}_{d+n}$ and $\widetilde{\text{op}}_{d+n}$ dependent on regime R_k is calculated using (4.5) by marginalizing over the individual Gaussians and cluster centers, which is shown in (4.11).

$$p\left(\widetilde{\text{np}}_{d+n} \cap \widetilde{\text{np}}_{d+n} | \widehat{R}_k\right) = \sum_{i=1}^{N} p\left(\widetilde{\text{np}}_{d+n} \cap \widetilde{\text{np}}_{d+n} | \zeta_i\right) P\left(\zeta_i | R_k\right). \tag{4.11}$$

We conclude that probabilities and densities used for the exponential smoother prediction process are now also dependent on procurement information. Also, trends are calculated differently for this newly added procurement information. However, the basic operation of the regime prediction model (regarding the exponential smoother process) is still quite similar to the current model.

Markov Process

Now that we have extended the prediction of regime probabilities for today, we continue with extending the short-term and long-term prediction techniques, which are Markov prediction and Markov correction-prediction, respectively. Short-term predictions are defined as predictions for one to ten days into the future, whereas long-term predictions are defined as predictions for eleven to twenty days into the future.

As already explained in section 2.3.2, both Markov techniques are quite similar. For both techniques, predictions are based on identified regimes, but for Markov correction-prediction, the identified regimes are corrected first, which makes them also dependent on the history of the variables instead of just one value per variable.

The Markov transition matrices used in the predictions, $T\left(r_{d+n} | r_d\right)$, do not need to be redefined, since they are only dependent implicitly on the variables. Also, we focus on the n-day predictions only, and thus the repeated one-day predictions are omitted, because of observations made in section 2.3.2.

One can extend the Markov prediction process to include procurement information by using identified regime probabilities on day $d-1$ based on $\widetilde{\text{np}}_{d-1}$ and $\widetilde{\text{op}}_{d-1}$, instead of just $\widetilde{\text{np}}_{d-1}$. Equations (4.12) through (4.14) show how a vector with these regime probabilities is constructed:

$$p\left(\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1} | \widehat{R}_k\right) = \sum_{i=1}^{N} p\left(\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1} | \zeta_i\right) P\left(\zeta_i | R_k\right), \tag{4.12}$$

$$P\left(\widehat{R}_k | \widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right) = \frac{p\left(\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1} | \widehat{R}_k\right) P\left(R_k\right)}{\sum_{k=1}^{M} p\left(\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1} | \widehat{R}_k\right) P\left(R_k\right)}, \quad \forall k = 1, 2, \ldots, M, \tag{4.13}$$

$$\vec{P}\left(\widehat{r}_{d-1} | \widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right) = \left\{P\left(\widehat{R}_1 | \widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right), \ldots, P\left(\widehat{R}_M | \widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right)\right\}. \tag{4.14}$$

Equations (4.12) and (4.13) are derived from (4.5) and (4.6), respectively. Similarly, it is possible to extend the identification of regime probabilities on day $d-1$, while taking into account the entire history of values of both variables. The calculation of a vector containing these corrected probabilities (being the basis of the Markov correction-prediction process) is performed by evaluating two expressions, which are

$$P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right)=$$
$$\frac{p\left(\widetilde{\mathrm{np}}_{d-1}\cap\widetilde{\mathrm{op}}_{d-1}|\widehat{R}_k\right)P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-2}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-2}\right\}\right)}{\sum_{k=1}^{M}p\left(\widetilde{\mathrm{np}}_{d-1}\cap\widetilde{\mathrm{op}}_{d-1}|\widehat{R}_k\right)P\left(\widehat{R}_k|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-2}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-2}\right\}\right)},$$
$$\forall k=1,2,\ldots,M, \tag{4.15}$$
$$\vec{P}\left(\widehat{r}_{d-1}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right)=$$
$$\left\{P\left(\widehat{R}_1|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right),\ldots,\right.$$
$$\left.P\left(\widehat{R}_M|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right)\right\}. \tag{4.16}$$

Here, a Bayesian correction is applied by identifying the regime probabilities while taking into account the predicted probabilities of yesterday $(d-1)$ on the day before yesterday $(d-2)$ and yesterday's probability density of $\widetilde{\mathrm{np}}_{d-1}$ and $\widetilde{\mathrm{op}}_{d-1}$ dependent on $\widehat{R}_k$. This density is calculated by applying (4.12).

Now the vectors shown in (4.14) and (4.16) can be plugged into the n-day Markov prediction and correction-prediction processes, as shown in (4.17) and (4.18), respectively:

$$\vec{P}(\widehat{r}_{d+n}|\widetilde{\mathrm{np}}_{d-1}\cap\widetilde{\mathrm{op}}_{d-1})=$$
$$\sum_{r_{d+n}}\cdots\sum_{r_{d-1}}\left\{\vec{P}(\widehat{r}_{d-1}|\widetilde{\mathrm{np}}_{d-1}\cap\widetilde{\mathrm{op}}_{d-1})\cdot T_n\left(r_{d+n}|r_{d-1}\right)\right\},$$
$$\forall n=0,1,\ldots,h, \tag{4.17}$$
$$\vec{P}\left(\widehat{r}_{d+n}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right)=$$
$$\sum_{r_{d+n}}\cdots\sum_{r_{d-1}}\left\{\vec{P}\left(\widehat{r}_{d-1}|\left\{\widetilde{\mathrm{np}}_1,\ldots,\widetilde{\mathrm{np}}_{d-1}\right\}\cap\left\{\widetilde{\mathrm{op}}_1,\ldots,\widetilde{\mathrm{op}}_{d-1}\right\}\right)\cdot\right.$$
$$\left.T_n\left(r_{d+n}|r_{d-1}\right)\right\},\quad\forall n=0,1,\ldots,h. \tag{4.18}$$

Here, the previous (identified for $n=0$, or otherwise predicted) posterior regime probabilities dependent on the (history of) normalized mean sales prices and mean procurement offer prices are multiplied with the applicable Markov transition matrix.

In general, the basic operation of the regime prediction model (regarding Markov processes) is still equal to the current model. However, probabilities

and probability densities are now also dependent on procurement information, forcing us to redefine the entire framework.

4.2.3 Extensions to Price Prediction

Extending the price density distribution of n days into the future (up to planning horizon h) shown in (2.27) leads to a two-dimensional density function, $p\left(\widehat{\text{np}}_{d+n} \cap \widehat{\text{op}}_{d+n} | \left\{\widetilde{\text{np}}_1, \ldots, \widetilde{\text{np}}_{d-1}\right\} \cap \left\{\widetilde{\text{op}}_1, \ldots, \widetilde{\text{op}}_{d-1}\right\}\right)$, which is defined as

$$
\begin{aligned}
&p\left(\widehat{\text{np}}_{d+n} \cap \widehat{\text{op}}_{d+n} | \left\{\widetilde{\text{np}}_1, \ldots, \widetilde{\text{np}}_{d-1}\right\} \cap \left\{\widetilde{\text{op}}_1, \ldots, \widetilde{\text{op}}_{d-1}\right\}\right) = \\
&\sum_{k=1}^{M} p\left(\text{np} \cap \text{op} | R_k\right) P\left(\widehat{R}_{k,d+n} | \left\{\widetilde{\text{np}}_1, \ldots, \widetilde{\text{np}}_{d-1}\right\} \cap \left\{\widetilde{\text{op}}_1, \ldots, \widetilde{\text{op}}_{d-1}\right\}\right), \\
&\forall n = 0, 1, \ldots, h. \qquad (4.19)
\end{aligned}
$$

Again, we marginalize over k densities based on regimes and over predicted regime probabilities for n days into the future. The density shown here is again created by sampling over the variables. However, because a dimension has been added, an increment of 0.01 for np would result in time consuming calculations, which we try to avoid.

Therefore, sampling over np is done by using 50 equal increments, for values ranging from 0.00 to 1.25. This leads to a more course-grained approximation of the density, but it is assumed that it is still fine-grained enough. Also, op is sampled over by using 50 equal increments, for values ranging from the minimum op the regime model is trained on and the maximum op. This leads to 2,500 samples with which a density grid is created, in contrast to 126 samples used for the approximation of the one-dimensional density function.

The agent does not need a two-dimensional density function, since only the sales price trends are needed for further decision making. Therefore, the grid is projected onto the np axis by selecting the maximum density on the op axis for each value of np. Then, the cumulative density function (CDF) of the price density distribution can be calculated again, after the density function has been discretized and normalized. Finally, the CDF is used for estimating medians for each day in the future, which are used for calculating daily price trends.

4.3 Concluding Remarks

In this chapter, we have seen how two-dimensional Gaussian Mixture Models are defined and how to compute their posterior probabilities. Clustering the latter probabilities is done using the `K-Means` algorithm, in order to make sure the way the regimes are interpreted by the agent remains the same. One should keep in mind that, although new regimes can be defined easily

with the framework, defining new regime clusters also implies relabeling the clusters, which is a problem to be solved. Furthermore, we presented a way to relate the identified regime clusters to the normalized mean sales price and mean procurement offer price.

Also, we have extended the prediction processes and the resulting sales price (trend) predictions, so that they are based on both sales and procurement information. The basic operations are still similar to the current model, but other probabilities and probability densities are used, since they are now based on procurement information (i.e., offer prices) as well.

In our next chapter, we need to use the framework elaborated on in this chapter for creating new regime models. For this, mostly the same configurations as presented in this chapter can be used.

Chapter 5

Experimenting with a New Regime Model

This chapter discusses offline and online experiments related to regime identification and prediction. We try to determine an optimal model configuration offline, which is to be used in online experiments. Section 5.1 elaborates on experiments done offline, whereas section 5.2 discusses online experiments. Section 5.3 summarizes this chapter into a few conclusions regarding the experimental results.

5.1 Offline Regime Experiments

This section elaborates on regime-related experiments we run offline using `MATLAB` to simulate specific game situations, build regime models, and evaluate experimental results. The experiments include training regime models using the framework discussed in chapter 4, and evaluating these regime models.

5.1.1 Experimental Setup

Experiments are done based on the data set introduced in chapter 3. Models are trained using the training set shown in table 3.1, and are evaluated mostly using the test set defined in the latter table. Before any experiments are done, outliers in the data are removed. In our experiments, we evaluate models with different numbers of regimes and other settings, but eventually, one model is chosen to be implemented for online testing. Furthermore, a separate model is trained for each product or market segment.

On a side note, as stated earlier, current implementations of the regime prediction framework differ slightly from the framework as it is explained in this thesis. However, we implement the newly defined framework exactly as proposed. Also, no ensemble prediction is implemented; the predictors are

assigned to the appropriate time spans discussed in chapters 2 and 4. This simplifies calculations, but should still give feasible results.

Training

Training regime models is done based on the framework discussed in chapter 4. This implies that for regime identification purposes, we create Gaussian Mixture Models with no fixed means and variances and with a maximum number of iterations of fifteen hundred. We are able to vary the number of Gaussians and of course the data we train on (i.e., we distinguish our models per product segment). Furthermore, with respect to clustering the posterior probabilities, the `K-Means` clustering algorithm is to be applied with the same configurations (i.e., squared Euclidean distance measure, fifteen replicates, a maximum number of one hundred iterations, and a fifty by fifty probability matrix per cluster), but we are able to change the number of clusters desired. The settings for the experiments on training identification models are shown in table 5.1, together with prediction settings, which are discussed later on in this section.

The number of Gaussians N in the Gaussian Mixture Model (GMM) is set to either three, five, ten, fifteen or twenty-five in our experiments. Increasing the number of Gaussians used to fit a GMM on data leads to more complex models and increases computation times. To shorten the total computation time of the experiments, we only go up to twenty-five Gaussians, which should give desired results given the current regime model, which can do with sixteen Gaussians. To shorten computational time even further, only a selection of numbers of Gaussians is evaluated.

For the number of regime clusters M we experiment with three and five and of both numbers we try to find an optimal configuration, since the current regime model works with three or five regimes (i.e., labels have been defined for these regimes and games have been successfully played using them). Other numbers of regimes are applicable as well, but are not used very often. Furthermore, with three and five regimes we will keep things understandable, because identified regime clusters can be related to real-world regimes from economic theory (with or without extreme regimes).

Besides varying the number of Gaussians and clusters, we also vary the product segments the models are trained on. In our experiments, models are trained for low-, mid-, and high-range products, so that we get a view on the way models fit on different types of products, without going into too much detail on products individually.

As shown in table 5.1, for regime predictions, we use several techniques, which are all discussed in chapter 4. Recall that no ensemble prediction is used, and therefore each prediction algorithm is assigned to the most appropriate time span. For the prediction of today's regime probabilities, exponential smoothing is applied to the last known values of the selected

Technique	Property	Value
GMM	Data	Training set (all)
	Fixed means	No
	Fixed variances	No
	Maximum iterations	1500
	Number of Gaussians (N)	$\{3, 5, 10, 15, 25\}$
	Product segments	$\{\text{low}, \text{mid}, \text{high}\}$
Clustering	Data	Training set (all)
	Algorithm	`K-Means`
	Distance measure	Squared Euclidean
	Replicates	15
	Maximum iterations	100
	Number of clusters (M)	$\{3, 5\}$
	Product segments	$\{\text{low}, \text{mid}, \text{high}\}$
	Probability matrices	$M\times$ 50 by 50
Pred. today	Data	None
	Technique	Exp. smoother pred.
	Days	$d + 0$
Pred. short-term	Data	Training set (all)
	Technique	Markov pred.
	Days	$d + 1, \ldots, d + 10$
Pred. long-term	Data	Training set (all)
	Technique	Markov corr.-pred.
	Days	$d + 11, \ldots, d + 20$

Table 5.1: Overview of settings for offline training experiments.

variables, in order to be able to calculate a trend. This trend can be used to predict future values and thus future regime probabilities can be determined. In contrast to the other prediction methods which are discussed next, for this type of prediction no specific training is required.

For the prediction of the regime probabilities up to ten days in the future, training is required. Here, an n-day Markov prediction is applied, which uses offline trained transition matrices (one for each day up to the planning horizon). These matrices are trained using data from the training set, as well as the probabilities of the identified regimes in our new models. Note that n-day prediction is used instead of repeated one-day prediction, because of observations made in section 2.3.2.

As an extension to the Markov prediction process, we apply a Markov correction-prediction process to compute regime probabilities up to twenty days in the future. The same Markov transition matrices are used as for Markov predictions, but identified regimes are corrected before any other calculations take place.

Evaluation

Evaluating the trained models is done in several ways, demonstrated in table 5.2, which should – once combined – give a sufficient view on the models on which decisions about the performance of models can be made. Apart from that, evaluation is done based on both training and test sets.

The training set is used for evaluating the regime clusters resulting from the clustering of the posterior probabilities of a Gaussian Mixture Model. This evaluation is done by means of a correlation analysis, the results of which can be used to assign proper labels to the identified regime clusters. Seventy-five hundred data points drawn from the training set are used to calculate the Pearson correlation between the identified dominant regime and economic regime identifiers, such as factory utilization and finished goods. This number of samples is large enough to ensure p-values below 0.01.

The test set is used for evaluating the course of regime probabilities over game time. For evaluation purposes, the results of a typical agent in a typical game are extracted from the test set to generate graphs from. We choose the second manufacturer of the fourth game in the test set (with identification number 792tac01).

Technique	Property	Value
Correlation	Data	Training set (all)
	Type	Pearson
	Number of samples	7500
	Product segments	{low, mid, high}
Probabilities	Data	Test set (792tac01)
	Agent ID	11
	Product segments	{low, mid, high}
Entropy	Data	Test set (792tac01)
	Agent ID	11
	Product segments	{low, mid, high}
Relative Entropy	Data	Test set (all)
	Product segments	{low, mid, high}
Prediction	Data	Test set (all)
	Product segments	{low, mid, high}

Table 5.2: Overview of settings for offline evaluation experiments.

To see whether models give feasible regimes over time (e.g., there should not be one dominant regime throughout the game and there should not be a completely different regime every single day), we take a look at graphs on the course of regime probabilities. If applicable, the regime probabilities generated by the current regime model are taken into account as well. Also, the confidence of the models is measured using the entropy, of which graphs are evaluated for our selected typical agent and game. As stated

in section 2.3.2, low entropies indicate a high confidence, whereas high entropies indicate low confidence. Furthermore, an average relative entropy is calculated for each model, so that models can be compared to each other.

The entropy of a set of online estimated regime probabilities $\widetilde{R}$ on an arbitrary game day d for a specific game f and product group g and a typical agent m is calculated as shown in (5.1). This equation is used for generating graphs of the course of entropies estimated online during a specific game for a typical agent and product segment:

$$\begin{aligned}\text{entropy}(\widetilde{R})_{g,f,m,d} = \sum_{k=1}^{\text{M}} &- P\left(R_k|\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right)_{g,f,m,d} \\ &\log_2 P\left(R_k|\widetilde{\text{np}}_{d-1} \cap \widetilde{\text{op}}_{d-1}\right)_{g,f,m,d}. \end{aligned} \tag{5.1}$$

The relative entropy is calculated as shown in (5.2). Here, the relative entropy is calculated by expressing the entropy as a percentage of the maximum entropy, which is equal to $\log_2 M$:

$$\text{relEntropy}(\widetilde{R})_{g,f,m,d} = \frac{\text{entropy}(\widetilde{R})_{g,f,m,d}}{\log_2 M} \times 100\%. \tag{5.2}$$

The average relative entropy of each individual model is obtained as shown in (5.3) through (5.6). Here, we calculate the entropies for each product (1, ..., numP), game (1, ..., numG), manufacturer (1, ..., numM), and day (1, ..., numD). Per product, the daily entropies are averaged per manufacturer to obtain average game entropies, which are averaged again so that one single average entropy remains per product. To obtain the average relative entropy per model, the average entropies of all product segments are averaged. We define average relative entropies as

$$\text{avgRelEntropy}(\widetilde{R})_{g,f,m} = \frac{\sum_{d=1}^{\text{numD}} \text{relEntropy}(\widetilde{R})_{g,f,m,d}}{\text{numD}}, \tag{5.3}$$

$$\text{avgRelEntropy}(\widetilde{R})_{g,f} = \frac{\sum_{m=1}^{\text{numM}} \text{avgRelEntropy}(\widetilde{R})_{g,f,m}}{\text{numM}}, \tag{5.4}$$

$$\text{avgRelEntropy}(\widetilde{R})_{g} = \frac{\sum_{f=1}^{\text{numG}} \text{avgRelEntropy}(\widetilde{R})_{g,f}}{\text{numG}}, \tag{5.5}$$

$$\text{avgRelEntropy}(\widetilde{R}) = \frac{\sum_{g=1}^{\text{numP}} \text{avgRelEntropy}(\widetilde{R})_{g}}{\text{numP}}. \tag{5.6}$$

Regime predictions are evaluated by means of counting processes. We express the number of times a correct regime is predicted as a percentage of the total number of predictions. Also, the number of times a regime change is correctly predicted (using a safety margin of plus or minus two days) is expressed as a percentage. The prediction performance of the new

regime model is compared to the prediction performance of the current regime model (described in section 2.3.2). These measures have already been introduced in section 2.3.2. Here, we also introduced other measures, but these measures are computationally more intensive and our currently selected measures provide a quick look at the performance of the prediction.

Now that we have defined our training and evaluation experiments, we can go on to presenting the experimental results in section 5.1.2.

5.1.2 Experimental Results

After running experiments using the setup as described in section 5.1.1, we find that the higher the number of Gaussians, the lower the average relative entropy. This is the case for models based on three regimes, as well as models which are based on five regimes. The three-regime models generally have somewhat lower entropies than their five-regime equivalents. Table 5.3 supports the latter observations.

Usually, increasing the number of Gaussians leads to a decrease in average relative entropies. This means models become more confident when Gaussians are added. This phenomenon can be explained by the fact that one can define a more detailed probability model, resulting in more accurate predictions and thus a higher confidence.

Because of our setup, we do not look any further than 25 Gaussians and as the results show, it is likely we also do not need to look at more models based on more Gaussians. Although the entropy keeps declining when more Gaussians are added to the model, the amount with which the entropy declines keeps getting smaller.

A confident model (i.e., a regime model which returns low entropy values) only implies a good model to a certain extent. Of course a model needs a certain amount confidence, because otherwise it cannot predict anything well, but too much confidence can easily lead to overfitting: the model returns wrong values with absolute certainty. Thus, we evaluate the course of the estimated regime probabilities over game time for game 792tac01.

	3 clusters	5 clusters
3 Gaussians	75.0898%	79.2392%
5 Gaussians	65.7491%	59.3567%
10 Gaussians	50.5709%	57.6967%
15 Gaussians	50.8337%	54.4893%
25 Gaussians	46.3429%	47.1262%

Table 5.3: Overview of calculated average relative entropies for models with different combinations of number of Gaussians and number of clusters. Settings are to be found in section 5.1.1. For calculation of the average relative entropies, (5.6) is used.

In what follows, the estimates – and, where applicable, other evaluations – are based on data related to the agent with identification number 11 in the latter game and trained models are analyzed for low-, mid-, and high-range products.

Three-Regime Models

This section discusses experiments on regime models based on three regimes. First, the optimal number of Gaussians is determined, after which further analysis is done on the best performing models.

Determining the Optimal Number of Gaussians

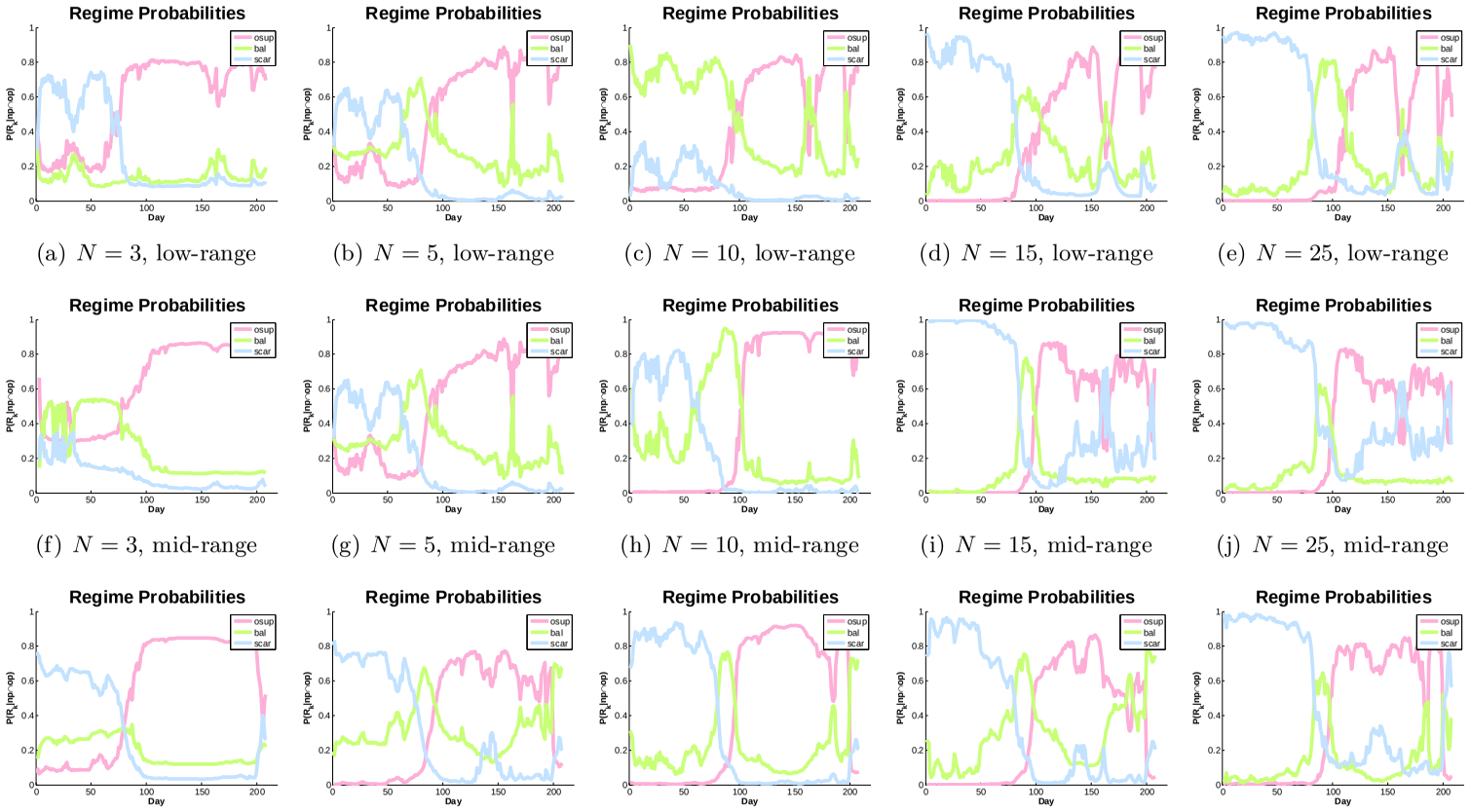
Figure 5.1 shows the course of regime probabilities during the game for models trained on data on low-, mid-, and high-range products. For each model, three clusters have been identified. It is clear that different numbers of Gaussians lead to different courses of regime probabilities. Regime clusters are labeled by means of a correlation analysis, which is discussed in more detail later on.

Models trained with three Gaussians do not perform very well. Although low-range products seem to have a plausible model, the regime probabilities and/or the dominant regimes of mid- and high-range products are infeasible. For mid-range products, the game starts with an oversupply regime, which is highly unlikely, since no finished products are available at the start of an arbitrary game. Each game should start with a scarcity regime. Furthermore, the model for high-range products seems to experience some overfitting, which is indicated by a high probability for the same regime for a long time.

Five-Gaussian and ten-Gaussian models return feasible regime probability estimations. Probabilities vary over time, but there are not too many regime changes. Also, in contrast to the three-Gaussian models, the game starts with a scarcity regime for all evaluated product segments. However, there is one exception to make; a regime model with ten Gaussians which is trained on low-range product data returns a high estimated probability for a balanced regime at the start of a game, which does not fit well to the statement made earlier that games should start with a dominant scarcity regime.

Besides the regime models built on five and ten Gaussians, models created with fifteen or twenty-five Gaussians give good results as well. However, for the mid-range products, we see some regime swapping between scarcity and oversupply around day 150, which might not be necessary.

For other agents, the same observations hold, and sometimes these agents yield more extreme results. For instance, for the first agent of the first game in our test set, as well as the fifth agent of the fifth game in our test set, only one dominant regime is identified for approximately the entire game when more than ten Gaussians are used for fitting a Gaussian Mixture Model.



(a) $N = 3$, low-range (b) $N = 5$, low-range (c) $N = 10$, low-range (d) $N = 15$, low-range (e) $N = 25$, low-range

(f) $N = 3$, mid-range (g) $N = 5$, mid-range (h) $N = 10$, mid-range (i) $N = 15$, mid-range (j) $N = 25$, mid-range

(k) $N = 3$, high-range (l) $N = 5$, high-range (m) $N = 10$, high-range (n) $N = 15$, high-range (o) $N = 25$, high-range

Figure 5.1: Course of estimated regime probabilities during an arbitrary game (ID 792tac01) for a typical agent (ID 11) for low-, mid-, and high-range products. The number of clusters (M) used in the two-dimensional Gaussian Mixture Model (based on normalized mean sales-side prices and mean procurement-side offer prices) is three, whereas the number of Gaussians (N) is equal to either three, five, ten, fifteen, or twenty-five.

It seems that only with five Gaussians, always a feasible regime model is created.

As we already observed when calculating the average relative entropies, the more Gaussians we use to fit a Gaussian Mixture Model, the higher the confidence of the regime model. The question is: how confident must our optimal regime model be? We do not want to risk overfitting the model (i.e., the model returns more or less the same estimates for regime probabilities for many days in a row in a game), so we do not need to use a lot of Gaussians. Also, using less Gaussians saves computation time.

As we have seen in the graphs, using three Gaussians did not give feasible results, whereas fifteen and twenty-five Gaussians seem to risk overfitting. Therefore, five or ten Gaussians are likely to be our best pick, even though using ten Gaussians for fitting a Gaussian Mixture Model does not always work out well. Supported by the latter observations and by the fact that adding more Gaussians to the model leads to more or less the same results (apart from some regime swapping), albeit somewhat more extreme, a total number of five Gaussians is selected to be the best setting for a three-regime model.

Further Analysis of the Selected Models

Now that we have determined the number of Gaussians which possibly yields good regime models when three regimes are to be identified, we can take a look at the characteristics of these models, after which we can continue evaluating the performance. Figure 5.2 shows the Gaussian Mixture Model fit on low-range product data from the training set using five Gaussians. The Expectation-Maximization algorithm as mentioned in chapters 2 and 4 is used for determining the optimal values of the model's parameters. Fitting for mid- and high-range products produces models which look more or less the same as the model which is shown in the figure.

As we take a look at the posterior probabilities for each cluster (defined as $P\left(R_k|\text{np} \cap \text{op}\right), \forall k = 1, 2, 3$ after applying the `K-Means` algorithm to the posterior probabilities of the Gaussian Mixture Model) in figure 5.3, we notice that there are clearly dominant regimes for most of the combinations of normalized mean sales-side prices and mean procurement-side offer prices. Low sales prices and high (procurement) offer prices do not yield convincing dominant regimes, since the difference between the regime probabilities is rather small. Products belonging to the mid- and high-range result in similar posterior probability plots. Note that at this point, no labels have been assigned to identified clusters yet. Therefore, the colors of the different clusters in this figure do not necessarily match the colors of the regimes in figure 5.1.

In addition to the graphs showing the course of the estimated regime probabilities for the low-, mid-, and high-range products (based on data from

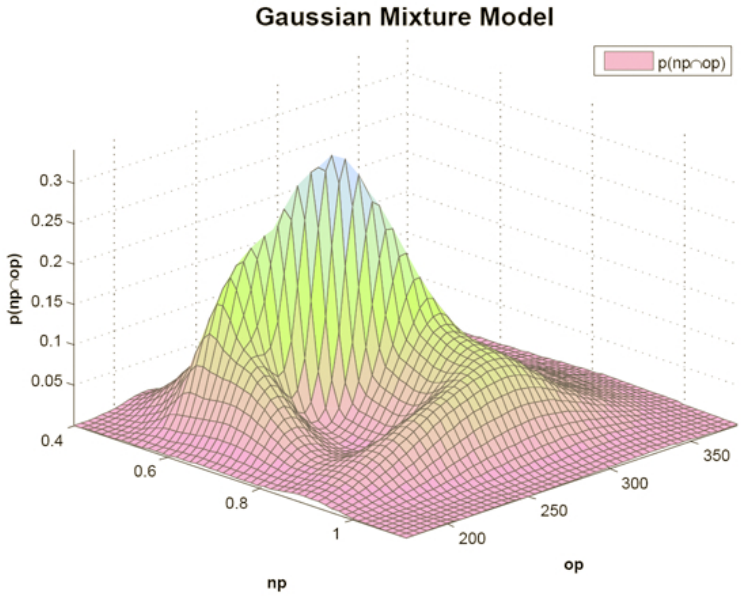


Figure 5.2: A two-dimensional Gaussian Mixture Model on mean sales price and mean procurement offer price, using five Gaussian components, with no fixed means and variances. This model is trained with a maximum of fifteen hundred iterations on training data (table 3.1) on the low product segment.

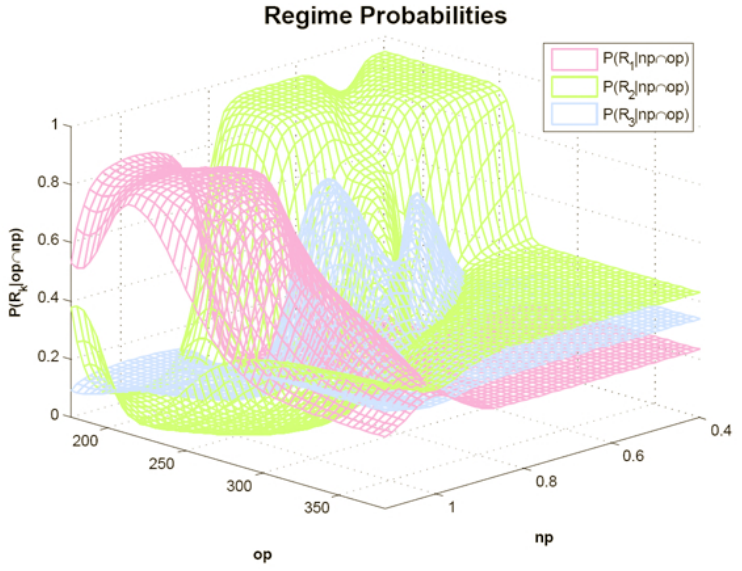
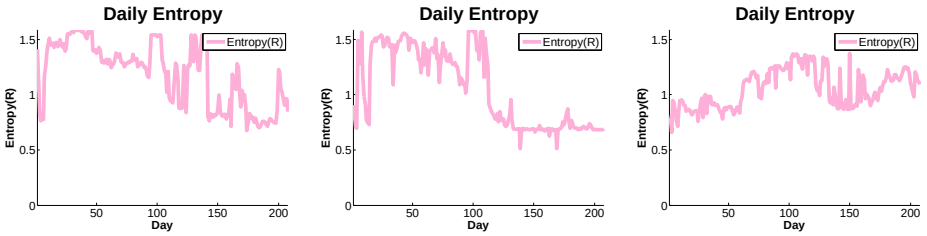


Figure 5.3: Regime probabilities (given mean sales price and mean procurement offer price) for low-range products, resulting from a two-dimensional Gaussian Mixture Model, of which its posterior probabilities are clustered in three clusters. The probabilities are based on training data (table 3.1).

the training set), we can also evaluate the daily entropies of the estimated regime probabilities for the second agent of the fourth game in our test set. These entropies are referred to as entropy$(\widetilde{R})_{g,f,m,d}$, where g is either low, mid, or high, f is equal to four, m equals two and d is for all game days. The daily entropies of the observed game and agent from the test set are shown in figure 5.4.



(a) low-range (b) mid-range (c) high-range

Figure 5.4: Daily entropies of the estimated regime probabilities for a specific game (ID 792tac01) and a typical agent (ID 11) using test data (table 3.1). Regimes are estimated based on clustered regime probabilities resulting from a two-dimensional Gaussian Mixture Model with five individual Gaussians. Three regimes are identified.

The latter figure follows directly from figure 5.1, because regime probabilities close to each other generate high entropies and a clearly dominant regime results in a low entropy. Therefore, on days in the game where the estimated regime probabilities are for instance close to each other, the entropy has a high value. Figure 5.4 enables the viewer to see quickly how close regime probabilities are on an arbitrary game, which is why we also take a look at these plots, instead of just the course of the regime probabilities.

As we could already predict using the average relative entropies (table 5.3), daily entropies in the analyzed game are indeed rather high, i.e., close to the maximum entropy value for this number of regimes. We recall from section 5.1.1 that this maximum value is equal to $\log_2 M$, where M is the number of regimes. Therefore, the maximum entropy is approximately 1.5850, and thus daily entropies approaching this value are considered as high entropies.

The entropy often approaches its maximum value throughout the game, but low entropy values are obtained occasionally as well. The models for low-, mid-, and high-range products predict regimes in the beginning of the game with high confidence, which should be correct, since a game always starts with a scarcity regime. However, the models' confidence declines rapidly after the first few days, and thus the daily entropy values increase. Often, end-game situations are hard to identify regimes for. However, the model for mid-range products seems to be very confident of its estimations.

We do not know whether the estimations are correct, but we do know there is a possibility that some overfitting has occurred.

In order to be able to assign regime labels to the regimes (i.e., scarcity, balanced, and oversupply), we need to perform some correlation studies. As stated earlier, these correlation studies are applied to each model shown in figure 5.1, so that a proper label can be assigned to each regime cluster. We now continue with elaborating on the correlation studies performed on three-regime models based on a Gaussian Mixture Model with five individual Gaussians.

Figure 5.5 shows the correlations of the identified regime clusters of a model trained on low-range product data with several regime indicators. The p-values for the correlation analysis are all approximately zero, i.e., less than 0.01. The correlations are similar to those of the models trained for mid-range and high-range products. A few indicators (or factors) are derived from [43] and [17], where regime models are introduced with three and five (labeled) regimes. The indicators used for our correlation studies are both market parameters as well as the variables used for training the regime models. Market parameters are the game day, the amount of finished goods, factory utilization and sales-side offer ratio (offers divided by demand).

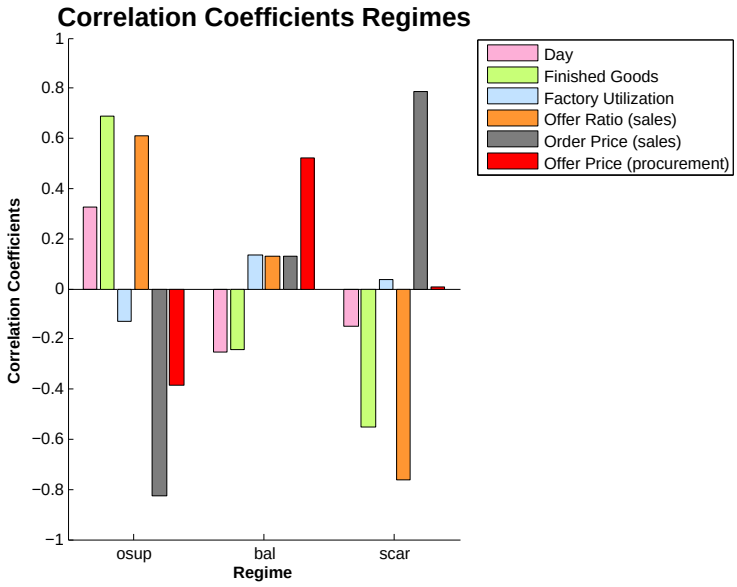


Figure 5.5: Correlation coefficients of identified regime clusters, resulting from a two-dimensional Gaussian Mixture Model with five individual Gaussians created based on low-range product data. Three regimes are identified.

Looking at the graph, we see results similar to those presented in [43] and [17]. Market parameters correlate the same way with the new regimes as they do with the current regimes. Also, the normalized mean (sales) prices

seem to correlate similarly. However, we notice that the procurement offer prices do not always correlate in a logical way with the regimes.

An oversupply situation is strongly and positively correlated with time, the amount of finished goods available, and the sales-side offer ratio. This means that an oversupply regime mostly occurs towards the end of a game. Also, a lot of finished goods are available and the offer ratio is high in an oversupply situation, which is according to the associated economic theory, because there are too many goods available and the demand falls behind. There is a negative correlation with factory utilization, which means that factories are used less in an oversupply situation, which is feasible, because there is no need to produce at full power, but agents will keep producing because of a possible predicted regime change in the (near) future. Finally, there is a (strong) negative correlation with normalized mean sales-side prices (also referred to as order prices) and mean procurement-side offer prices, which indicates low prices.

Scarcity correlates with the market parameters differently. We notice that scarcity is more likely to occur in the beginning of a game. Finished goods and (sales-side) offer ratio show a strong negative correlation, which means that there is a lack of finished goods and the offer ratio is low, which points out a possibility that the customer demand is higher than the offered quantity of finished goods. Sales prices show a strong positive correlation, indicating high prices, which is sound to economic theory. Offer prices show almost no correlation at all.

A balanced situation shows low positive and negative correlations with all indicators, indicating it is a true balanced situation. However, procurement offer prices show a strong (and positive) correlation with the balanced regime.

As we have seen, correlation studies show that, despite some side notes, regime labels can be assigned to the identified regime clusters, because the clusters actually represent economical regimes defined in economic theory. Furthermore, the entire analysis of the selected model discussed above shows that a regime model based on normalized mean (sales) prices and procurement-side offer prices is feasible when five Gaussians are used in the Gaussian Mixture Model and three regime clusters are identified. To examine the applicability of the newly defined regime model even more, we take a look at the accuracy of the model's predictions, compared to the current model.

Looking at the prediction performance of the selected regime model (compared to the performance of the current model), one can observe improvements, as well as deteriorations. This observation is supported by table 5.4. Here, the accuracy measured in a percentage of correctly predicted regimes and regime switches. The table shows the performance of the new model compared to the current model for three market segments. The score of the best performing model is printed bold. Note that the current model is

represented by a one-dimensional Gaussian Mixture Model based on sixteen Gaussians (using fixed means and variances) and mean sales prices.

Correct	Market	New model	Existing model
Regime	Low-range	**56.21%**	51.86%
	Mid-range	**68.15%**	52.93%
	High-range	**50.91%**	41.91%
Time	Low-range	38.36%	**52.78%**
	Mid-range	**53.60%**	43.44%
	High-range	33.81%	**46.30%**

Table 5.4: Prediction performances of a two-dimensional Gaussian Mixture Model with three clusters and five individual Gaussians (new model) compared to the performance of a one-dimensional five-cluster Gaussian Mixture Model with sixteen individual Gaussians (existing model). Results are shown for three market segments and are based on test data (table 3.1).

One can conclude that the newly defined two-dimensional Gaussian Mixture Model performs better in predicting the correct regimes up to twenty days in the future than the current model. This makes sense, since we have defined less clusters, and thus the probability of predicting the correct dominant regime automatically increases. In one market segment, the new regime model predicts moments of regime change more accurately than the current regime model, whereas the other market segments fall behind.

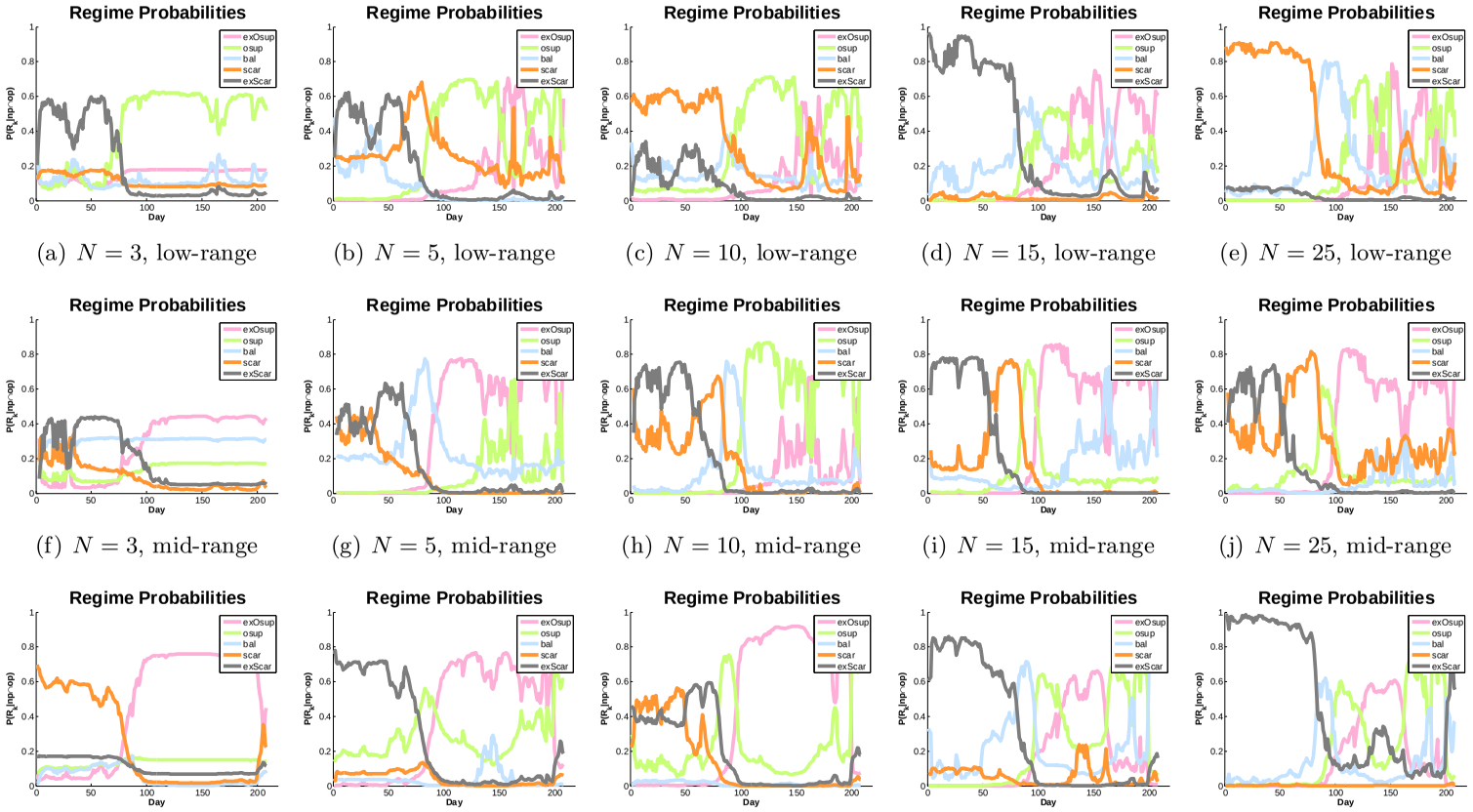
Taking into account all previous observations in the evaluation of the selected configuration of the new regime model, one can conclude that, although regime labels can be assigned to the clusters and the model is feasible, the prediction performance of the model falls somewhat behind.

Five-Regime Models

This section discusses experiments on regime models based on five regimes. First, the optimal number of Gaussians is determined, after which further analysis is done on the best performing five-regime models.

Determining the Optimal Number of Gaussians

Figure 5.6 shows the course of regime probabilities during the game for models trained on data on low-, mid-, and high-range products. For each model, five regime clusters have been identified. It is clear that different numbers of Gaussians lead to different courses of regime probabilities. Also, regimes tend to switch more often. Again, regime clusters are labeled by means of a correlation analysis quite similar to the analysis discussed for three-regime models. Details of the correlation studies for five-regime models are given later on.



(a) $N = 3$, low-range (b) $N = 5$, low-range (c) $N = 10$, low-range (d) $N = 15$, low-range (e) $N = 25$, low-range

(f) $N = 3$, mid-range (g) $N = 5$, mid-range (h) $N = 10$, mid-range (i) $N = 15$, mid-range (j) $N = 25$, mid-range

(k) $N = 3$, high-range (l) $N = 5$, high-range (m) $N = 10$, high-range (n) $N = 15$, high-range (o) $N = 25$, high-range

Figure 5.6: Course of estimated regime probabilities during an arbitrary game (ID 792tac01) for a typical agent (ID 11) for low-, mid-, and high-range products. The number of clusters (M) used in the two-dimensional Gaussian Mixture Model (based on normalized mean sales-side prices and mean procurement-side offer prices) is five, whereas the number of Gaussians (N) is equal to either three, five, ten, fifteen, or twenty-five.

Models trained with three Gaussians do not perform well at all. For mid-range products, the model tends to remain indecisive, since the estimated probabilities for each regime cluster stay close to each other throughout the game. For low-range and high-range products, the probabilities of three regimes are close to each other, whereas the other two regimes always have high probabilities. For each model holds that the game only knows approximately two dominant regimes. These regimes are extremes to each other, e.g., scarcity versus extreme oversupply for products of the high-segment. Three Gaussians seems to result in a very bad fit on our data.

Five-Gaussian models seem to perform better, but still not convincible. For each product segment, a clear dominant regime can be estimated daily. However, for high-range products, two dominant regimes are identified (apart from an almost negligible third), which are the two extreme variants of scarcity and oversupply. This does not seem to be a feasible model. As for the other two evaluated product segments, the course of regime probabilities over game time looks feasible. However, for a few other agents and games, probabilities similar to the high-range model are returned.

Gaussian Mixture Models based on ten individual Gaussians perform well. Although not all regime clusters are used in each model, there is no hopping between extreme regimes as we encountered with other models. Note that for most other games and agents all regime clusters are used; these games give nice patterns of the distribution of regime probabilities over game time. The graph of estimated regime probabilities for the low-range product segment reveals that the identified dominant regime at the beginning of the game is not extreme scarcity, but scarcity. This might be caused by the fact that the differences between these two regimes are relatively small for this product segment. However, this is just a minor issue, since both regimes represent a certain extent of scarcity.

According to the plots in figure 5.6, models with fifteen and twenty-five Gaussians perform well at first sight. However, (extreme) scarcity regimes are not always used and their probabilities remain approximately zero throughout the game. As is the case with the three-regime models, overfitting occurs with other agents and games. For instance, the third agent in the third game of our test set (769tac02), as well as the first agent in the first game (761tac02) return the same dominant regime for almost every single day.

This being said, models built on fifteen or twenty-five Gaussians do not suit our needs well. Furthermore, three-Gaussian models have low performance, possibly caused by the fact that there are more clusters than Gaussians. If we fit Gaussian Mixture Models using five Gaussians, we see too few different dominant regimes in a game on a general basis. This leaves our ten-Gaussian models, which perform reasonably, although not perfectly. On a general note, a lot of regime skipping ("hopping") occurs in many regime models, but apparently we cannot avoid that. Supported by the latter

observations, we choose to select a total number of ten Gaussians to be the best setting for a five-regime model.

Further Analysis of the Selected Models

Now that we have determined the number of Gaussians which possibly yields good regime models when five regimes are to be identified, we can take a look at the characteristics of these models, after which we can continue evaluating the performance. Figure 5.7 shows the Gaussian Mixture Model fit on low-range product data from the training set using ten Gaussians. Again, the Expectation-Maximization algorithm as mentioned in chapters 2 and 4 is used for determining the optimal values of the model's parameters. Fitting for mid-range products produces a model which looks similar to this model, i.e., low probability densities and one clear peak. A Gaussian Mixture Model fit on high-range product data results in a model with higher probability densities and lacks a clear peak.

As we take a look at the posterior probabilities for each low-range product regime cluster in figure 5.8, we notice – in contrast to our observations for the three-regime model – that there are clearly dominant regimes for nearly any combination of sales and offer prices. Note that some clusters are dominant on a larger area than other clusters. The models for mid-range and high-range products result in less complex posterior probability plots.

The daily entropies of the observed game (792tac01) and agent (identification number 11) from the test set are shown in figure 5.9. The relative difference between the daily entropy values and the maximum entropy (which is equal to $\log_2 5$, i.e., approximately 2.3219) is on average lower for five-regime models than for three-regime models. This can also be concluded using the average relative entropies in table 5.3, which indicates that identification with three-regime models fit using five Gaussians results in a higher average relative entropy than with five-regime models based on ten Gaussians.

Except for the high-range products model, the moving average of the entropies remains the same throughout the game, so on a time frame of about ten days, approximately the same average entropy holds. The regime model fit on low-range product data has a higher volatility than the mid-range products model, illustrated by some large spikes about halfway through the game (and on).

In contrast to the three-regime models, all evaluated five-regime models show no clear difference between mid-game confidence and the confidence of the estimations at the beginning of a game. The latter confidence was high for all three-regime models, but is low for all five-regime models. This could be caused by the fact that extreme scarcity and scarcity regimes are close to each other, and therefore their probabilities are close to each other, leading to a higher entropy and thus a lower confidence.

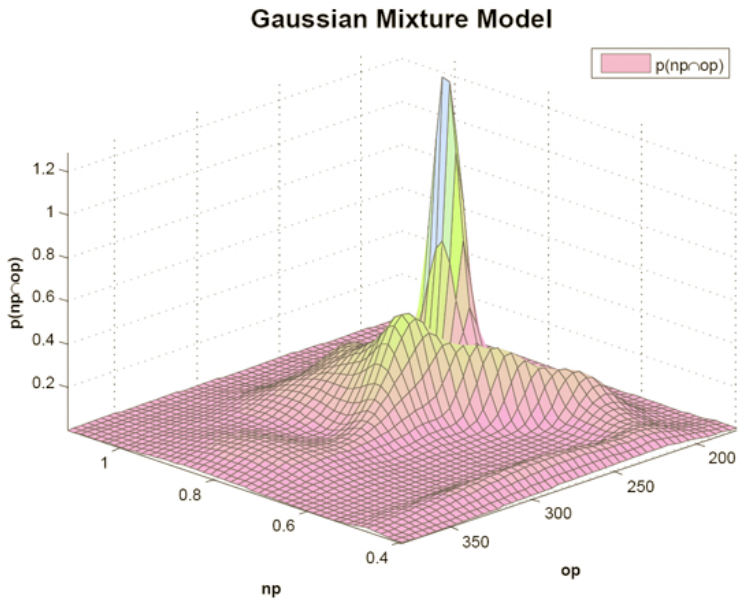


Figure 5.7: A two-dimensional Gaussian Mixture Model on mean sales price and mean procurement offer price, using ten Gaussian components, with no fixed means and variances. This model is trained with a maximum of fifteen hundred iterations on training data (table 3.1) on the low product segment.

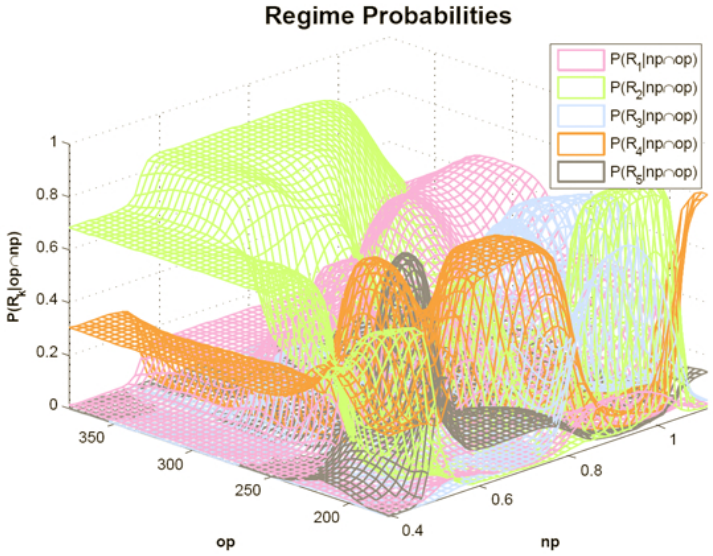
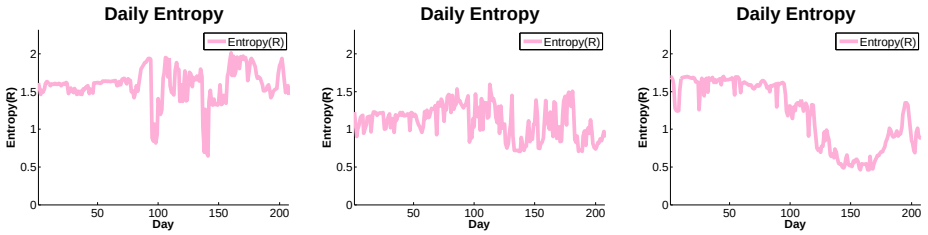


Figure 5.8: Regime probabilities (given mean sales price and mean procurement offer price) for low-range products, resulting from a two-dimensional Gaussian Mixture Model, of which its posterior probabilities are clustered in five clusters. The probabilities are based on training data (table 3.1).



(a) low-range (b) mid-range (c) high-range

Figure 5.9: Daily entropies of the estimated regime probabilities for a specific game (ID 792tac01) and a typical agent (ID 11) using test data (table 3.1). Regimes are estimated based on clustered regime probabilities resulting from a two-dimensional Gaussian Mixture Model with ten individual Gaussians. Five regimes are identified.

Furthermore, the model fit on high-range product data shows an increase in confidence (i.e., we observe an entropy drop) halfway through the game, which declines again near the end of the game. We did not observe this for three-regime models in figure 5.4. The sudden change in confidence can also be derived from figure 5.6, where the high-range ten-Gaussian model identifies an extreme oversupply regime with a high probability for a long time.

Labeling the regimes is done the same way as for three-regime models by using correlation analyses. As stated earlier, these correlation studies are applied to each model shown in figure 5.6, so that a proper label can be assigned to each regime cluster. We now continue with elaborating on the correlation studies performed on five-regime models based on a Gaussian Mixture Model with ten individual Gaussians.

Figure 5.10 shows the Pearson's correlations of the identified regime clusters of a model trained on low-range product data with several regime indicators. Again, the p-values for the correlation analysis are all less than 0.01. The correlations are similar to those of the model trained for mid-range products. The high-range product model results in slightly different correlations for the oversupply regime. The same market parameters and variables are used as indicators as is the case with three-regime models.

Looking at the graph, we see very similar results to those presented earlier for three-regime models (figure 5.5). The main difference is that for scarcity and oversupply extreme variants have been added. They share characteristics with their non-extreme variants, but show slightly different correlations (positive or negative).

Again, as is the case with three-regime models, the balanced regime shows low positive and negative correlations with all indicators. Also, procurement-side offer prices show different correlations with the each regime

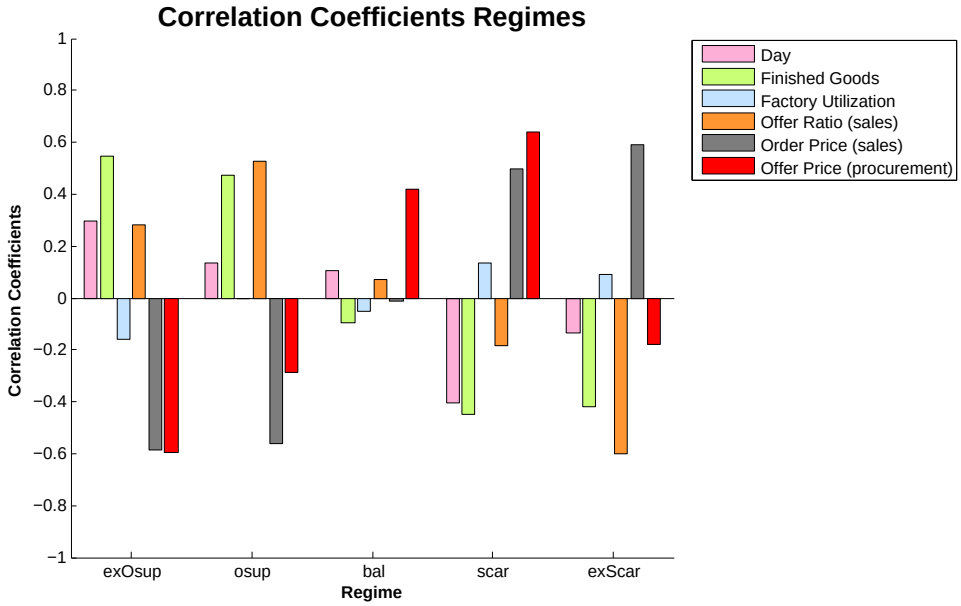


Figure 5.10: Correlation coefficients of identified regime clusters, resulting from a two-dimensional Gaussian Mixture Model with ten individual Gaussians created based on low-range product data. Five regimes are identified.

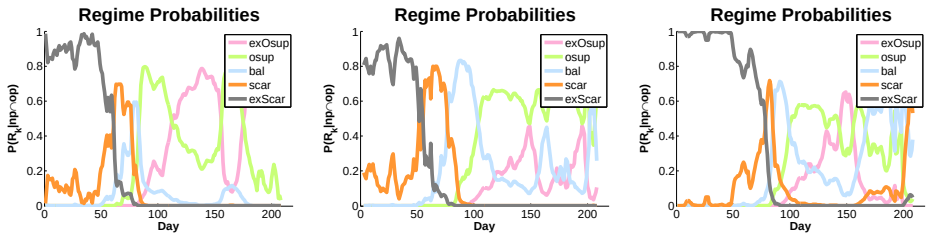
and do not follow the correlations of the normalized mean sales-side prices. we notice that the mean offer prices do not always correlate in a logical way with the regimes.

Both oversupply regimes are strongly and positively correlated with time, the amount of finished goods available, and the offer ratio. The extreme oversupply regime is more correlated with time than the oversupply regime, indicating that extreme oversupply mostly occurs towards the end of a game and that in end-game situations the probability of having an extreme oversupply regime is higher than the probability of having an oversupply regime. Also, a lot of finished goods are available and the offer ratio is high in both oversupply situations, which is according to the associated economic theory. There is a negative correlation with factory utilization in extreme oversupply situations, which means that factories are used less in these situations. Oversupply regimes show a less strong (negative) correlation with factory utilization. Finally, there is a (strong) negative correlation with normalized mean prices and mean procurement offer prices, which indicates low prices. The characteristics of both oversupply regimes match those elaborated on in [17].

Scarcity regimes also show resemblances with the regime characteristics shown in [17]. In contrast to (extreme) oversupply, scarcity is more likely to occur in the beginning of a game. Extreme scarcity occurs less when a game

starts. Finished goods and offer ratio show a strong negative correlation, which means that there is a lack of finished goods and the offer ratio is low. For extreme scarcity situations, the correlation with finished goods is about the same as for scarcity situations, but offer ratios show stronger correlations. Sales prices show a strong positive correlation, indicating high prices. Again, extreme scarcity shows stronger correlations than scarcity.

Plots of the course of estimated regime probabilities using the current regime model for the same agents and the same game as in figure 5.6 give us more insight in the feasibility of the new regime model. Figure 5.11 shows estimated regime probabilities for the three evaluated product segments. A one-dimensional Gaussian Mixture Model based on sixteen Gaussians (using fixed means and variances) is used to estimate the regime probabilities of five regimes using normalized mean sales prices.



(a) low-range (b) mid-range (c) high-range

Figure 5.11: Course of estimated regime probabilities during an arbitrary game (ID 792tac01) for a typical agent (ID 11) for low-, mid-, and high-range products. The number of clusters (M) used in the one-dimensional Gaussian Mixture Model (based on normalized mean sales-side prices) is five, whereas the number of Gaussians (N) is equal to sixteen. Fixed means and fixed variances are used.

If we compare figures 5.6 and 5.11, we observe that the estimated regimes differ. Furthermore, the volatility of the current and new regime models is similar, as well as some regime switching moments. Mostly, the models differ in the identified (dominant) regime. The identified regime in the current model is often close to the identified regime in the new model, e.g., a scarcity situation in the current model versus an extreme scarcity situation or a balanced situation in the new regime model. This means regimes are identified differently, which can possibly mean improvements in the agent's overall performance.

It should be noted that comparing identified regimes of the new regime model with those of the current regime model does not tell much about the quality of the new regime model. For instance, if the identified regimes are exactly the same, one could wonder whether adding procurement information influences the regime model. However, identical regime probabilities

could also just be a coincidence, emerging when procurement information takes on the right values. Therefore, one can only use these comparisons to observe characteristics such as volatility.

Looking at the prediction performance of the selected regime model (compared to the performance of the current model), one can observe small improvements, as well as small deteriorations. This observation is supported by table 5.5. Here, the accuracy measured in a percentage of correctly predicted regimes and regime switches. The table shows the performance of the new model compared to the current model for three market segments (i.e., low-range, mid-range, and high-range products). The score of the best performing model is printed bold.

Correct	Market	New model	Existing model
Regime	Low-range	46.43%	**51.86%**
	Mid-range	40.63%	**52.93%**
	High-range	40.48%	**41.91%**
Time	Low-range	48.68%	**52.78%**
	Mid-range	**53.00%**	43.44%
	High-range	**50.93%**	46.30%

Table 5.5: Prediction performances of a two-dimensional Gaussian Mixture Model with five clusters and ten individual Gaussians (new model) compared to the performance of a one-dimensional five-cluster Gaussian Mixture Model with sixteen individual Gaussians (existing model). Results are shown for three market segments and are based on test data (table 3.1).

In contrast to the prediction performance of the two-dimensional three-cluster Gaussian Mixture Model discussed earlier, this model predicts regime switches slightly more accurately than the current model (most of the time). Also, regimes are predicted with a lower accuracy than the current model. The differences between the performances are smaller than we have seen with the three-regime model compared to the current regime model.

As is the case with three-regime models, correlation studies on five-regime models show that regime labels can be assigned to the identified regime clusters. Furthermore, the entire analysis of the selected model discussed above shows that a regime model based on normalized mean (sales) prices and procurement-side offer prices is feasible when Gaussian Mixture Models are fit on data using ten individual Gaussians and when five regime clusters are identified. Also, predicting future regimes and regime switches is about as accurate as is the case in the current regime model.

We observe that the addition of procurement information does not cause instable regime models. Because the models are trained differently than the existing regime model, they will give other outputs during a game. Thus, it is likely our new regime model will influence the game results, but we

cannot predict at this point whether the amount of cash at the end of a game will be higher or lower, and thus whether there is any advantage to using procurement information. However, we can state that we do not loose much stability when using a new regime model instead of MinneTAC's existing regime model.

5.1.3 Offline Regime Experiments Conclusions

After analyzing several different regime models, we find that none of the analyzed models perform perfectly, so in order to be able to select feasible regime model configurations, it appears we have to find a balance between low entropies (i.e., high confidence), plausible courses of estimated regime probabilities and computation time. Therefore, we continue our research with our second configuration.

We have presented two possible configurations of regime models to be used in an online game environment. Our first configuration is based on three regimes and five individual Gaussians, whereas our second configuration is based on five regimes and ten individual Gaussians. All other GMM- and clustering-specific parameters can be found in table 5.1 (section 5.1.1). The confidence of our second configuration is higher than the confidence of our first configuration. Now we continue our research with our second configuration.

5.2 Online Regime Experiments

Now that an optimal regime model configuration is determined, we can implement and test this configuration in the MinneTAC trading agent. This section elaborates on the experimental setup of our online experiments and on the experimental results.

5.2.1 Experimental Setup

In contrast to the experiments in the previous section, no history games are used, but newly run games. Online experiments are done by implementing the new regime model in the agent and by running test games with the MinneTAC agent using the current and the new regime model.

Agents and Games

Besides the fact that there is no need to train models based on game data which are available from the game servers and that new games need to be run instead, online experiments differ from the offline experiments in several ways. First of all, offline experiments are conducted per market segment, but online experiments are done at the individual product level.

This allows an agent to be more reactive in the supplier market, because the products belonging to a certain market segment can have different regimes, for instance caused by a shortage of a certain component which is not used by all products in a market segment. Furthermore, the offline experiments do not use ensemble prediction, but online this type of prediction is easy to implement and gives more accurate results. Even so, the most recent agent configurations, one of which is used as a benchmark, all implement ensemble prediction.

For benchmarking purposes, we run a number of games with the configuration of the 2008 MinneTAC agent, where regime prediction is done using the three prediction algorithms which are introduced in chapter 2, i.e., exponential smoother prediction, Markov prediction, and Markov correction-prediction. For the sales-related tasks the regimes are used for, the agent is constructed in a way such that means and trends follow from each of the predictors. These predicted means and trends are used for (for example) product pricing. The predictors each identify yesterday's regime and predict future regimes based on the prediction techniques explained in chapter 2.

Our own MinneTAC variant is almost equal to the benchmark agent. The two agents only differ in the used predictors, since our MinneTAC variant implements the new regime model and thus requires other predictors. As is the case with the predictors used in the offline experiments, the implemented predictors of the MinneTAC agent are exactly implemented as defined in the framework proposed in chapter 4, in contrast to the predictors used in the current version of MinneTAC which is used for benchmarking purposes. However, as stated earlier, we use ensemble prediction in online predictions.

By changing the predictors only, the effect of the new regime model can be measured most accurately, since no other changes are made. In case experiments are conducted and analyzed properly, changes in for instance game results can be attributed to the changes made in the regime model of the MinneTAC agent.

As stated earlier, we need to run new games in order to be able to analyze the performance of the new regime model online. Figure 5.12 breaks down the online experiments into competitor sets, experiment sets (and their associated seed sets), and MinneTAC variants.

A number of games are played against different sets of competitors (ω). We choose two sets of competitors, i.e., a set of easy competitors and a set of tougher agents, so that the effects of the new regime model can be measured in different environments.

Because of randomness in each game, we need to select a number of different sets of game seeds randomly (ρ). These sets are used for each variant of the agent (θ) and are the same in the different sets of competitors. Recall that we have two agent variants, i.e., the benchmark agent and an agent configuration implementing the new regime model. The usage of seed sets enables us to run TAC SCM games where exactly the same characteristics

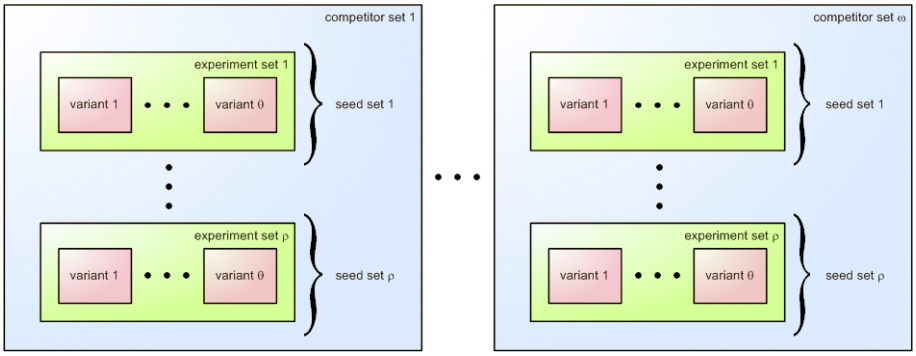


Figure 5.12: Experimental setup of online experiments, where ω refers to the number of competitor sets, ρ refers to the number of seed sets (which are linked to experiment sets), and θ refers to the number of MinneTAC variants.

– such as bank interests and storage costs – hold. Thus, it is possible to replay a certain game with the same competitor set and game characteristics, but with a different MinneTAC configuration. In other words, each game scenario (a certain seed set) is run for both agent variants, so that the effect of the new regime model can be evaluated.

In order to reduce the effect of randomness in the agents' behavior, we need to run a significant number of experiment sets. Because of the time constraints for this thesis, we run forty experiment sets per competitor set. The number of games to be run boils down to eighty individual games per set of competitors, and thus a total number of one hundred and sixty games are to be run.

Table 5.6 shows the main settings of the online experiments. The games run in the experiments follow the TAC SCM 2006 game specifications [13], which are also discussed in chapter 2. In our experiments, we refer to the benchmark agent as BENCH, whereas the new MinneTAC agent using the new regime model is referred to as RIPPI, which is an acronym for Regime Identification and Prediction using Procurement Information, to distinguish between the current regime model – which only uses sales information – and the adapted regime model.

As stated earlier, both the benchmark agent and the new agent configuration are tested against two sets of competitors. The competitors in the first set are dummy agents, which are merely very basic agents supplied by the TAC SCM game. The agents only send one RFQ for components to a randomly selected supplier and accept each offer. Furthermore, dummy agents apply a random factor when determining customer RFQ offer prices and handle customer orders in due date order [44].

The competitors in the second set are competitors which are discussed in chapter 2. Since there is no binary version of the CMieux agent publicly

Property	Value
Game specifications	TAC SCM 2006
MinneTAC variants (θ)	2, which are: {BENCH, RIPPI}
Competitor sets (ω)	2, which are: {Dummy-1, Dummy-2, Dummy-3, ... Dummy-4, Dummy-5}, and {TacTex07, Crocodile05, DeepMaize07, ... PhantAgent07, Mertacor05}
Experiment sets (ρ)	40, using randomly selected game seeds

Table 5.6: Overview of settings for online game experiments.

available, but the other agents discussed in this thesis have released binary versions [45], TacTex, CrocodileAgent, DeepMaize, PhantAgent, and Mertacor are included in the competitor set. The latest available binaries at the time of writing are used, so for CrocodileAgent and for Mertacor this means the 2005 binaries are used, whereas for the other agents holds that the 2007 binary versions are used.

Evaluation

For evaluation purposes, the agent's amount of money in cash at the end of each game is measured, together with the associated total orders placed by customers. For each participant in games played with a certain competitor set, the mean and standard deviations for each variable are calculated, to give an indication of the game characteristics.

For the MinneTAC agent, the results of the benchmark variant (BENCH) and the new variant (RIPPI) are compared. The average alteration of cash and orders (both absolute and relative) are considered. We define the mean absolute change in cash as

$$\text{absChCash} = \frac{\sum_{q=1}^{\text{numQ}} \text{cashRIPPI}_q}{\text{numQ}} - \frac{\sum_{q=1}^{\text{numQ}} \text{cashBENCH}_q}{\text{numQ}}, \tag{5.7}$$

where q refers to a game, numQ represents the number of games, and cashRIPPI_q and cashBENCH_q contain MinneTAC's amount of cash at the end of game q for the RIPPI variant and the BENCH variant, respectively. The mean absolute change in orders is defined similarly:

$$\text{absChOrders} = \frac{\sum_{q=1}^{\text{numQ}} \text{ordersRIPPI}_q}{\text{numQ}} - \frac{\sum_{q=1}^{\text{numQ}} \text{ordersBENCH}_q}{\text{numQ}}. \tag{5.8}$$

In the latter equation, q and numQ refer to a game and the number of games, respectively. Furthermore, ordersRIPPI_q and ordersBENCH_q repre-

sent MinneTAC's total number of orders at the end of game q for the RIPPI variant and the BENCH variant, respectively.

Relative changes are calculated using the absolute mean changes in (5.7) and (5.8). Here, we divide the absolute mean changes in cash and orders by the absolute (non-negative) mean of the BENCH results, which is written out in (5.9) and (5.10):

$$\text{relChCash} = \frac{\frac{\sum_{q=1}^{\text{numQ}} \text{cashRIPPI}_q}{\text{numQ}} - \frac{\sum_{q=1}^{\text{numQ}} \text{cashBENCH}_q}{\text{numQ}}}{\left|\frac{\sum_{q=1}^{\text{numQ}} \text{cashBENCH}_q}{\text{numQ}}\right|}, \tag{5.9}$$

$$\text{relChOrders} = \frac{\frac{\sum_{q=1}^{\text{numQ}} \text{ordersRIPPI}_q}{\text{numQ}} - \frac{\sum_{q=1}^{\text{numQ}} \text{ordersBENCH}_q}{\text{numQ}}}{\left|\frac{\sum_{q=1}^{\text{numQ}} \text{ordersBENCH}_q}{\text{numQ}}\right|}. \tag{5.10}$$

If the average amount of money in cash (i.e., the agent's bank account) at the end of a game or the average total number of orders placed by customers is larger for RIPPI than for BENCH, this is considered as an improvement. Otherwise, the change is a deterioration of the game results.

However, a difference alone in cash or orders does not say much about the effects of the new regime model, because it can be a coincidence that the agent performs better or worse on average, for instance because of an outlying result. Therefore, we need to apply a statistic to evaluate the significance of the difference between both variants of the MinneTAC agent. Differences are considered to be significant when it is unlikely they have occurred by chance. In statistics, the paired T-test is often used for this purpose. However, this is a parameterized test, which requires or assumes the samples to be from the same normal distribution.

In [17], the Wilcoxon paired two-sided signed rank test is used to assess significance between experiments. This statistic is unparameterized and, in contrast to the paired T-test, does not require its samples to be from a normal distribution. This is most suitable to our experiments. The test hypothesizes that the difference between the two samples (i.e., mean cash or orders of BENCH and RIPPI) comes from a continuous, symmetric distribution with a median of zero. Rejecting this hypothesis (for a p-value of less than 0.05) means that the two samples significantly differ from each other and expresses a certain significance.

Therefore, improvements (or deteriorations) of game results measured using the absolute or relative difference between the benchmark agent and the new MinneTAC agent, are considered to be proven when they are significant. Otherwise, the executed changes to the regime model (i.e., adding procurement information) are not likely to have a large impact on the game results of the agent.

Measure	Agent
Mean	All participants
Standard deviation	All participants
Mean absolute change	MinneTAC
Mean relative change	MinneTAC
Signed rank test	MinneTAC

Table 5.7: Overview of evaluation methods for online game experiments, which are employed for each competitor set, for both available amount of money in cash (bank account) at the end of a game and the total number of customer orders to the agent.

The performance measures used per competitor set for both cash and customer orders are summarized in table 5.7. Now that we have defined the settings of our online games and our evaluation criteria, we can go on to presenting the experimental results in section 5.2.2.

5.2.2 Experimental Results

After running forty games against TAC SCM dummy agents for both MinneTAC configurations (i.e., the benchmark configuration which implements the current regime model and the new configuration which implements the new regime model), we see that, not surprisingly, the MinneTAC agent outperforms the dummy agents. This is supported by table 5.8, where the agents are ranked on performance. The scores of the dummy agents show strong resemblances among themselves, which makes sense, since they all have the same characteristics.

	Cash (×1000)		Orders	
	BENCH	RIPPI	BENCH	RIPPI
MinneTAC	20,307 (13,247)	16,475 (5,744)	2,995 (1,245)	3,824 (628)
Dummy-3	12,477 (3,904)	13,780 (2,902)	3,242 (338)	3,165 (281)
Dummy-5	12,558 (3,791)	13,699 (3,102)	3,237 (367)	3,158 (309)
Dummy-2	12,407 (3,782)	13,839 (3,113)	3,227 (334)	3,171 (307)
Dummy-4	11,925 (4,921)	14,029 (2,981)	3,207 (339)	3,185 (307)
Dummy-1	12,139 (4,234)	13,777 (2,694)	3,248 (317)	3,152 (294)

Table 5.8: Overview of the game results of each participant. Forty games are run for both configurations of MinneTAC (i.e., BENCH and RIPPI) against the same set of (dummy) competitors. Results are shown for the amount of money in cash at the end of a game and the associated number of customer orders (rounded mean and standard deviation).

Measure	Cash	Orders
Overall change (absolute)	-3,832,396	829
Overall change (relative)	-18.8721%	27.6926%
Signed rank (p-value)	0.0878	0.0003

Table 5.9: Overview of the change in performance of the MinneTAC agent. Forty games are run for both configurations of MinneTAC (i.e., BENCH and RIPPI) against the same set of (dummy) competitors. Results are shown for the amount of money in cash at the end of a game and the associated number of customer orders.

However, for MinneTAC, we observe a decrease in the mean available amount of money at the end of a game when using the new regime model for sales-related decision making instead of the current regime model. We observe a larger increase in the average number of customer orders placed with the MinneTAC. The average absolute and relative changes are displayed in table 5.9.

In order to be able to classify the increase in the number of customer orders and the small increase in the amount of money as significant changes, we apply the Wilcoxon paired two-sided signed rank test. Here, the null hypothesis we need to reject is that the difference between the game results of both tested MinneTAC variants comes from a continuous, symmetric distribution with a median of zero. Not being able to reject the null hypothesis means that the difference of the results is too small to be classified as significant.

We set the significance level to 5%, which means that p-values returned by the signed rank test below 0.05 reject the null hypothesis. Any other values fail to reject the hypothesis. Applying the Wilcoxon signed rank test to the agent's bank account balance at the end a game results in a low p-value: 0.0878. However, the value is not below 0.05 and thus this means the decrease in money is not significant enough, and could have occurred by chance. Still, the odds are that the new regime model leads to a decrease in cash, because of its low p-value. The number of customer orders however, has increased significantly. The p-value for orders is about 0.0003, which is a strong evidence that the improvement is not a coincidence.

According to table 5.10, games run against competitors which are more intelligent and harder to compete with than the TAC SCM dummy agents, such as PhantAgent and TacTex, turn out to be very tough competitions. The average amount of money for each agent is lower than we have seen in the dummy games. The average number of customer orders per agent also indicates fierce competition, especially in combination with the low amount of money. It appears that products are sold against lower prices and thus optimizing sales-related (supportive) decision processes, such as the regime model, becomes more important.

	Cash (×1000)		Orders	
	BENCH	RIPPI	BENCH	RIPPI
PhantAgent07	8,510 (6,033)	8,770 (6,813)	6,411 (468)	6,392 (417)
TacTex07	7,708 (6,432)	7,979 (7,695)	6,756 (373)	6,500 (512)
DeepMaize07	3,248 (3,656)	3,534 (3,842)	4,524 (820)	4,271 (1,000)
Mertacor05	1,515 (4,754)	1,398 (5,263)	5,857 (620)	5,645 (712)
MinneTAC	-746 (5,310)	-1,273 (3,343)	3,735 (863)	4,461 (357)
Crocodile05	-2,251 (6,340)	-2,221 (7,824)	6,361 (548)	6,245 (572)

Table 5.10: Overview of the game results of each participant. Forty games are run for both configurations of MinneTAC (i.e., BENCH and RIPPI) against the same set of (tough) competitors. Results are shown for both the amount of money in cash at the end of a game and the associated number of customer orders (rounded mean and standard deviation).

In contrast to the dummy games, MinneTAC is not able to beat the other agents in a set of forty games. However, as is the case in the games played against TAC SCM dummy agents, the new MinneTAC variant (RIPPI) generates lower profits (measured in terms of available money in cash) and an increase in the number of customer orders when compared to the benchmark variant.

Table 5.11 shows that the average decrease in money is about seventy-one percent, whereas the total amount of associated customer orders (market share) increases on average with about nineteen percent. Again, we test the significance of the observed changes using the Wilcoxon paired two-sided signed rank test. It appears that the decrease in money is very likely to have occurred by chance and thus the expected generated amount of money in cash at the end of a game is more or less equal for both variants of MinneTAC. On the other hand, the significance of the increase of orders has been proved, which supports the findings for the dummy games that the number of customer orders are likely to increase when using the new regime model.

Measure	Cash	Orders
Overall change (absolute)	-527,209	725
Overall change (relative)	-70.6552%	19.4132%
Signed rank (p-value)	0.1254	0.0000

Table 5.11: Overview of the change in performance of the MinneTAC agent. Forty games are run for both configurations of MinneTAC (i.e., BENCH and RIPPI) against the same set of (tough) competitors. Results are shown for both the amount of money in cash at the end of a game and the associated number of customer orders.

5.2.3 Online Regime Experiments Conclusions

In our online experiments, two game environments have been considered. The first environment is an easy environment, in which the MinneTAC variants (a benchmark configuration containing the old regime model and a new configuration containing the new regime model) have been tested against relatively easy competitors, i.e., TAC SCM dummy agents. The second environment is a highly competitive one, in which MinneTAC competes with tough competitors, which belong to the current top performing agents. Although the environments are different, experimental results are more or less comparable.

We conclude that the number of customer orders to the MinneTAC agent increases significantly with roughly twenty-five percent. This improvement is less noticeable in highly competitive games. Furthermore, there is a slightly insignificant decrease in the amount of money for nineteen percent in games played against dummy agents. In contrast, we observe an insignificant but large decrease in profits of around seventy-one percent when competing with well performing artificial trading agents.

The p-values resulting from the Wilcoxon paired two-sided signed rank tests (used for measuring significance of changes) are more or less equal in both environments for orders, i.e., changes are very significant, as well as for MinneTAC's bank account balances at the end of a game. The significance of the latter changes is not high enough, but will increase when increasing the number of experiments.

5.3 Regime Experiments Conclusions

In our experiments presented in this chapter, we have evaluated several configurations of regime models offline. Some of them are based on three regimes, and others are based on five regimes. The regime models perform about the same as the current regime models, but now they are also based on procurement information. Because of their performances, regime models based on both sales and procurement information are likely to perform well online.

Therefore, the best performing regime model, i.e., a two-dimensional five-regime Gaussian Mixture Model based on ten Gaussian components, has been selected to be implemented in the MinneTAC agent. Using the new regime model in the MinneTAC agent, we have run several games against different competitors while using different seed sets to simulate varying game characteristics. It seems that when using the new regime model MinneTAC's market share increases significantly due to an increase in the number of customer orders linked to MinneTAC, but the amount of money at the end of a game is somewhat likely to, but does not necessarily have to be affected.

Chapter 6

Discussion

This chapter discusses the results of both offline and online regime identification and prediction experiments. Section 6.1 discusses our offline experiments, whereas section 6.2 elaborates on our online experiments. Section 6.3 summarizes the discussions.

6.1 Offline Regime Experiments

This section discusses the results of offline regime identification and prediction experiments and elaborates on various subjects, such as entropies (section 6.1.1) and regime hopping (section 6.1.2). Also, correlations and regime labeling are discussed in section 6.1.3, as well as identifying the correct number of regimes (section 6.1.4). Finally, the results of regime prediction are discussed in section 6.1.5.

6.1.1 Entropy

Although the entropy has been introduced as a measure for the quality of a regime model, it only measures the quality partially. As stated in chapter 5, the entropy measures the "confidence" of a model, or in other words, how certain the model is of its own predictions. In case of low entropy values, the model's confidence is high, and in case of high entropy values, the confidence is low. The entropy returns low values if one regime has a high probability (given normalized mean sales price and mean offer price) and the other regimes have low probabilities. However, if a model has great confidence (low entropy), it does not say anything about the quality of the model in terms of the correctness of the predicted regime probabilities or dominant regimes.

Therefore, we cannot use the entropy values as a measure by itself, but we have to take into account the feasibility of the course of estimated regime probabilities as well, in order to be able to decide which models (i.e., which

settings) perform well. Although the entropy might not be the best measure, we can use it after we pre-selected promising models by looking at the course of regime probabilities. In case of doubt, the model with the lowest entropy can be selected.

Models with low entropy values (and many Gaussians) have shown to be very confident about their regime identifications. However, plots on their estimated regime probabilities over time tell us something different and something valuable, which supports our statement that having low entropies does not imply being a good model, since incorrect regimes can be identified. Although the regimes are identified with high confidence, it is clear that only a few regimes are identified, often one regime for a great many of days in a row (with almost full certainty) in an arbitrary game, which is rather unrealistic.

Another point of discussion is the fact that five-regime models are not very confident about their identified regimes in the beginning of the game and mostly, their confidence does not change very much over game time. This could be caused by the fact that the five regimes look a lot like each other (e.g., extreme scarcity and scarcity), and therefore "related" regimes will generate high probabilities, resulting in somewhat higher entropies as is the case with three-regime models.

6.1.2 Regime Hopping

We notice a lot of regime hopping occurs in five-regime models. Regime hopping is defined as skipping one or more intermediary regimes during a regime switch. We believe that regime hopping could be caused by the fact that a variable (or dimension) has been added to the model. If we look closely to the way the current model works, we see that no regime is skipped at any time. By adding a variable, we can all of a sudden hop from one regime to the other regime. Perhaps this is caused by the fact that on an arbitrary game day, the procurement information tells us a big regime change is coming, even though the sales prices don't show us that (yet). This would mean that the added information actually adds value to the regime model.

To a certain extent, regime hopping might not be so strange as it appears. In the existing model, a single continuous variable is used which has a complete ordering, and thus hopping is not possible. Now we have two continuous variables. These variables have partial ordering, which is why hopping is possible.

In our new three-dimensional space, where regimes could be located (partially or fully) behind one another, and therefore a small change in a value of one variable (dimension) could mean a large regime switch. In that case, one variable plays a large role in identifying economic regimes. However, constantly hopping from one extreme regime to the other extreme regime, as well as identifying the same regime over and over again does not

make sense, because one of the properties of economic regimes is that they change gradually over time.

6.1.3 Correlations and Labeling

In our correlation studies, we have seen that each regime's correlation with market properties gives us valuable information, since it is testable with economic theories and heuristics. Our new regime clusters have the same characteristics as the existing clusters, i.e., low sales prices, many finished goods and a high (sales-side) offer ratio in oversupply situations, and high prices, a small amount of finished goods and a low offer ratio in case of scarcity situations. The balanced regime is always somewhere in between.

One could wonder whether the regimes are still the same as the current regimes. In an economic sense, we still have the same regimes, because their correlation coefficients match those elaborated on in [43] and [17]. The only thing that has been changed is the fact that regimes are not only identified based on the normalized mean (sales) prices, but also based on procurement information, i.e., the mean procurement-side offer prices. Now, other data points belong to the regime clusters than in the current model, which results in different identified regimes.

A better question would be whether the regimes are still valid, because one strange thing we noticed during our correlation experiments, is the fact that the regimes do not correlate in a logical way with procurement offer prices if we label regimes based on the correlations with the other indicators. Other correlations with days, finished goods, factory utilization, offer ratio, and sales order price make sense, but there appears to be some randomness in the correlation with mean offer prices. In our understanding, these prices should be correlated more or less the same way as normalized prices.

The randomness in the correlation can be caused by the fact that (some of) the observed agents active in the games stored in our data set have non-rational techniques for procurement-related decision making, causing the offer prices (as well as order prices) to behave in a non-logical way. It is hard to define regimes when taking the procurement offer prices into account, and based on the other market parameters and the normalized mean sales price, we believe that the regimes make sense.

6.1.4 Number of Regimes

In section 5.1.3 we decided to use five regimes in our regime model. Just by looking at the average relative entropies, this decision would be questionable, since three-regime models generally have lower entropies than their five-regime equivalents. However, the relative entropy of the best performing five-regime model is lower than the relative entropy of the best performing three-regime model. This is caused by the fact that different numbers of

Gaussians are used. But apart from that, as stated earlier in this discussion, the entropy is not a measure on its own, but should only be used as a guidance.

The MinneTAC agent has been using regime models which are based on five regimes for quite some time now and these models have proven to be successful. Furthermore, as stated by Ketter et al. [43], we can capture exceptional situations with the extreme regimes.

Our adapted regime models with five regimes experience a lot of regime hopping, whereas the three-regime models do not experience regime hopping at all. As been discussed in this chapter, to a certain extent, regime hopping does not make regime models infeasible.

Considering the latter observations, the decision of using a five-regime model to do our online testing is correct. All we need to do in order to decide which regime model performs best is to combine statistics (entropy) with heuristics and common sense (graphs). In other words, configuring the new regime model is all about finding a balance between low entropies, plausible courses of estimated regime probabilities and computation time.

6.1.5 Regime Predictions

Although Ketter et al. [43] indicate that about eighty to ninety percent of the predicted dominant regimes and regime switches is correct, repeating the same experiments as done with the new regime models results in much lower scores. This difference can be caused by the fact that we experiment on 2007 and 2008 data, which contains different games than the ones used in [43]. This may result in market conditions which are harder to predict, because agents are getting more advanced every year, which causes other decision making and thus different game characteristics.

Our experiments show that the the newly defined five-regime model predicts regime changes more accurately than the three-regime model analyzed earlier. Nevertheless, with the data set used in this thesis, the two-dimensional five-cluster Gaussian Mixture Model has a performance similar to the currently used regime model. These observations are supported by tables 5.4 and 5.5. We can conclude that the addition of procurement information generally has no affect on the performance.

It should be noted that in some cases, the five-regime model predicts future regimes worse than the current (sales-based) model. This may look strange at first, but the observed performance hit can be explained by the fact that the new model is characterized by two variables instead of one. This means that the future course of two variables should be predicted, which makes it easier to make mistakes. If the performance of the new model is more or less the same as the current model, it could mean that the model is feasible enough, since the model does not perform worse.

However, the performance of the models with respect to prediction does not tell us much about the effect the new models will have on the agent. To a certain extent, if a model predicts its own future values correctly, the model has a high predictive quality, which is desired. Nevertheless, we did not change the applied techniques in the regime model, but we only changed the regime definitions and extended some of the applied techniques. This means we should expect more or less the same results when it comes down to – for example – prediction performance. The agent's performance will not be influenced by the correctness of the predictions as much as by the change in regime definitions, and therefore, having approximately the same prediction accuracy as the current model does not mean the new model cannot perform better than the current model.

6.2 Online Regime Experiments

In online experiments with an implementation of the new regime model in the MinneTAC agent, we observed that the agent is able to survive in markets with easy competitors, but fails to beat strong competitors such as DeepMaize and PhantAgent. Generally, the amount of money in cash at the end of the game is less than the amount the current agent is expected to generate, whereas the number of customer orders is increased. This holds for both evaluated game environments.

The fact that MinneTAC's number of customer orders in a game changes (increases) significantly when using the new regime model, but profits tend to decrease – although not significantly – indicates there is a structural error in pricing. Procurement information structurally distorts the price trend estimation and thus the price to be set for customer orders. The number of orders increases, but prices decline, which does not result in higher profits. These phenomena are interrelated, since lower prices yield more orders and more orders yield lower prices. On a side note, the significance of performance changes can be improved by running more experiments. Tough competitors seem to require more experiments than the TAC SCM dummy agents.

Experiments show that the addition of procurement information to the regime model of the MinneTAC trading agent changes the characteristics of the agent, but does not result in increased profits. It is even possible profits decrease when applying the new regime model. Since the performance of agents in TAC SCM games is measured using their bank account balances at the end of each game, one could conclude that MinneTAC's performance is not improved when using the new regime model.

However, this is not entirely true. The performance in terms of MinneTAC's number of customer orders improves significantly. This is an opportunity the agent should make use of. If the order pricing mechanisms are

improved, not only the number of customer orders is likely to increase, but the bank account balances are likely to increase as well.

Besides a possible suboptimal pricing mechanism, a possible explanation for the observed lower profits is the fact that the price density function which is used for predicting price trends is merely a projection of a two-dimensional density function. This causes loss of data and therefore, the extended regime model might actually perform less accurately, especially because too low price trends are likely to be estimated: MinneTAC sets unnecessary low order prices, which leads to more customer orders, but also to lower profits. Projection as it is done currently (by using maximum values) only works well for non-rotated Gaussians, because in that case Gaussians have the same shape in each projection. However, our regime models have rotated Gaussians and thus applying a different type of projection is very likely to improve performances.

Furthermore, there could be problems regarding the added procurement information itself. In contrast to our assumption, this information – offer prices of the MinneTAC agent – might not be representative enough for the entire market, for instance due to non-standard behavior of MinneTAC. The two-dimensional regime model is trained on market procurement information and therefore, training the regime model on MinneTAC's regime information might help. This is, however, not guaranteed, because the agent still has a limited view on what is actually going on on the market. Otherwise, it is worth trying to train the regime model on a larger data set.

Another cause of the performance drop related to the procurement information is the fact that only an average value of the offer prices on a specific day is smoothed. Perhaps data should be smoothed as is currently done with the normalized mean prices, i.e., by applying Brown linear exponential smoothing to minimum and maximum values. This type of smoothing should give more accurate results.

Also, the calculated trend of the procurement information, which is used in the exponential smoother prediction process, is in reality only a rough indication of the trend and tends to be a very nervous estimation. This can lead to biased procurement and sales decisions.

Subsequently, no normalization is applied to procurement information currently. This non-normalized data could result in unwanted model behavior, since the model is only trained for a certain offer price range. It might be the case that during test games, prices exceed the minimum or maximum values the regime model is trained for. Normalizing the data so that it is always within a certain range could help improve the performance.

Finally, as the online experiments show that MinneTAC's internal regime model does not fit the reality in a game, we believe that introducing more adaptivity to the regime model is likely to improve MinneTAC's flexibility. This way, feedback is also directly given to the regime model, which enables MinneTAC to correct errors in estimations, which occur when market or

game conditions cause deviations from learned patterns. Structural errors as we have seen in our experiments are less likely to occur when good adaptivity is introduced.

6.3 Discussion Summary

We have discussed several aspects of the offline and online experimental results, such as the meaning and relevance of entropies and the occurrence of regime hopping (i.e., skipping one or more intermediary regimes during a regime switch). We conclude that regime hopping is feasible, because a dimension has been added to the regime model, resulting in two continuous variables which have a partial ordering.

Furthermore, the correctness of the assigned regime labels and the related number of regimes have been discussed. It is concluded that, although there seems to be randomness in the correlation with procurement offer prices, the regime labels are correct, as well as the related number of regimes.

Finally, offline regime prediction performance and the results of online experiments have been put into perspective. We conclude that in our offline prediction experiments, the new regime model shows similar performance to the existing regime model. However, in online experiments the new regime model seems to perform worse than the existing model. In this chapter, we have discussed several possible causes of the performance hit in online experiments, such as an insufficient projection of a two-dimensional model onto a one-dimensional price density.

Chapter 7

Conclusions

In this thesis, we have investigated the effects of adding procurement information to MinneTAC's sales-based regime model. Using the information gain metric, offer prices have been found to be the most valuable procurement information with which MinneTAC's regime model can be extended. Extending the regime model has been done by adding a dimension to the Gaussian Mixture Model which is the foundation of the regime model.

Offline experiments show that the extended regime model based on five regimes and ten Gaussians should perform about the same as the current regime model. It is concluded that it is not a trivial task to find an optimal configuration of a regime model, but that a balance has to be found between low entropy values (i.e., high confidence), plausible courses of estimated regime probabilities and computation time.

Despite the promising results of offline experiments, online performance of a MinneTAC agent implementing the sales- and procurement-based regime model is not as good as we hoped for. Overall, it is likely (but not significantly proven) that at the end of a TAC SCM game, the bank account balance of an agent implementing the extended regime model is lower compared to the amount of money of an agent using the current regime model. However, the number of customer orders gathered during a game increases significantly, so there is some improvement in performance. Unfortunately, this development is not likely to lead to higher profits.

We suggest to evaluate the different causes of the decrease in performance with respect to game profits in future research. For instance, because of a structural error in the extended regime model, resulting in low prices to be set, profits fall behind. We believe that the projection of a two-dimensional model onto a one-dimensional price density is done insufficiently and requires further research.

Also, introducing adaptivity or other pricing mechanisms, combined with the extended regime model could eventually lead to increased profits. Finally, it is possible to train the regime model on more or different pro-

curement information. Not only other data (e.g., offer quantities), but also other smoothing techniques applied to offer prices fall within the scope of the meaning of different procurement information.

Applying fuzzy `C-Means` clustering for determining regimes and adding a fuzzy interpretation of regimes to MinneTAC's regime model is something we will leave for further research as well. We believe that this will have a great impact on the way regimes are to be interpreted by the agent, because then, multiple regimes can hold partially at the same time. Finally, it is worth researching the use of procurement information in sales decision processes without a regime model, or even the use of this information in completely different decision processes of artificial trading agents.

Bibliography

[1] G. Ghiani, G. Laporte, and R. Musmanno, *Introduction to Logistics Systems Planning and Control*. John Wiley & Sons, Ltd, 2004.

[2] J. Collins, W. Ketter, and M. Gini, "Flexible Decision Control in an Autonomous Trading Agent," *Electronic Commerce Research and Applications (ECRA)*, Forthcoming 2009.

[3] D. Pardoe and P. Stone, "TacTex-05: A Champion Supply Chain Management Agent," in *Proceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI'06)*, pp. 1489–94, July 2006.

[4] C. Kiekintveld, J. Miller, P. R. Jordan, and M. P. Wellman, "Controlling a Supply Chain Agent Using Value-Based Decomposition," in *Proceedings of the 7th ACM conference on Electronic commerce (EC'06)*, (New York, NY, USA), pp. 208–217, ACM, 2006.

[5] M. Stan, B. Stan, and A. M. Florea, "A Dynamic Strategy Agent for Supply Chain Management," in *Proceedings of the Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC '06)*, (Washington, DC, USA), pp. 227–232, IEEE Computer Society, 2006.

[6] K. C. Chatzidimitriou, A. L. Symeonidis, I. Kontogounis, and P. A. Mitkas, "Agent Mertacor: A robust design for dealing with uncertainty and variation in SCM environments," *Expert Systems with Applications*, vol. 35, no. 3, pp. 591–603, 2008.

[7] M. Benisch, A. Sardinha, J. Andrews, and N. Sadeh, "CMieux: Adaptive Strategies for Competitive Supply Chain Trading," in *Proceedings of the 8th International Conference on Electronic Commerce (ICEC'06)*, 2006. Extended version.

[8] University of Minnesota, "TAC SCM 2008 Finals Results." Website, 2008. Available online, `http://tac01.cs.umn.edu:8080/history/competition/25/ttest/index.html`; last visited July 21, 2008.

[9] SICS AB, "TAC SCM 2005 Finals Results." Website, 2005. Available online, http://www.sics.se/tac/page.php?id=50; last visited July 21, 2008.

[10] SICS AB, "TAC SCM 2006 Finals Results." Website, 2006. Available online, http://www.sics.se/tac/page.php?id=61; last visited July 21, 2008.

[11] SICS AB, "TAC SCM 2007 Quarter-Finals Results (Group B)." Website, 2007. Available online, http://www.sics.se/tac/page.php?id=69; last visited July 21, 2008.

[12] University of Minnesota, "TAC SCM 2008 Semi-Finals Results (Group A)." Website, 2008. Available online, http://tac01.cs.umn.edu:8080/history/competition/24/ttest/index.html; last visited July 21, 2008.

[13] J. Collins, R. Arunachalam, N. Sadeh, J. Eriksson, N. Finne, and S. Janson, "The Supply Chain Management Game for the 2006 Trading Agent Competition," tech. rep., Carnegie Mellon University, November 2005.

[14] Apache Excalibur project, "Apache Excalibur and Fortress IOC Container." Website, 1997–2008. Available online, http://excalibur.apache.org/; last visited March 18, 2008.

[15] W. Ketter, J. Collins, M. Gini, A. Gupta, and P. Schrater, "Tactical and Strategic Sales Management for Intelligent Agents Guided By Economic Regimes." Submitted to Management Science, 2008.

[16] D. R. Osborn and M. Sensier, "The Prediction of Business Cycle Phases: Financial Variables and International Linkages," *National Institute Economic Review*, vol. 182, no. 1, pp. 96–105, 2002.

[17] W. Ketter, *Identification and Prediction of Economic Regimes to Guide Decision Making in Multi-Agent Marketplaces*. PhD thesis, University of Minnesota, 2007.

[18] James D. Hamilton, "A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle," *Econometrica*, vol. 57, no. 2, pp. 357–384, 1989.

[19] R. J. Hathaway and J. C. Bezdek, "Switching Regression Models and Fuzzy Clustering," *IEEE Transactions on Fuzzy Systems*, vol. 1, no. 3, pp. 195–204, 1993.

[20] Hung T. Nguyen and Berlin Wu and Vladik Kreinovich, "On Combining Statistical and Fuzzy Techniques: Detection of Business Cycles From

Uncertain Data," in *Proceedings of the International Conference on Information Technology (ICIT'99)*, pp. 69–74, 1999.

[21] W. Ketter, J. Collins, M. Gini, A. Gupta, and P. Schrater, "A Predictive Empirical Model for Pricing and Resource Allocation Decisions," in *Proceedings of the 9th International Conference on Electronic Commerce (ICEC'07)*, 2007.

[22] J. B. MacQueen, "Some methods of classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability (MSP'67)*, pp. 281–297, 1967.

[23] W. Ketter, J. Collins, M. Gini, A. Gupta, and P. Schrater, "Detecting and Forecasting Economic Regimes in Multi-Agent Automated Exchanges," *Decision Support Systems*, Forthcoming 2009.

[24] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[25] L. Zadeh, "Fuzzy Sets," *Information Control*, vol. 8, pp. 338–353, 1965.

[26] J. C. Dunn, "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters," *Journal of Cybernetics*, vol. 3, no. 3, pp. 32–57, 1973.

[27] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*. Norwell, MA, USA: Kluwer Academic Publishers, 1981.

[28] SICS AB, "TAC SCM 2007 Finals Results." Website, 2007. Available online, `http://www.sics.se/tac/page.php?id=70`; last visited July 21, 2008.

[29] V. Podobnik, A. Petric, and G. Jezic, *The CrocodileAgent: Research for Efficient Agent-Based Cross-Enterprise Processes*, vol. 4277 of *Lecture Notes in Computer Science*, pp. 752–762. Springer, 2006.

[30] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE - A FIPA-compliant agent framework," in *Proceedings of the Practical Applications of Intelligent Agents (PAAM'99)*, 1999.

[31] P. Vytelingum, R. K. Dash, M. He, and N. R. Jennings, "A Framework for Designing Strategies for Trading Agents," in *IJCAI Workshop on Trading Agent Design and Analysis (TADA'05)*, pp. 7–13, 2005.

[32] I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, eds., *Feature Extraction, Foundations and Applications*. Series Studies in Fuzziness and Soft Computing, Physica-Verlag, Springer, 2006.

[33] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[34] J. Andrews, M. Benisch, A. Sardinha, and N. Sadeh, "What Differentiates a Winning Agent: An Information Gain Based Analysis of TAC-SCM," in *AAAI Workshop on Trading Agent Design and Analysis (TADA'07)*, 2007.

[35] J. Andrews, M. Benisch, A. Sardinha, and N. Sadeh, *Lecture Notes on Business Information Processing*, ch. Using Information Gain to Analyze and Fine Tune the Performance of Supply Chain Trading Agents. Springer, 2008.

[36] The Mathworks, *Getting started with MATLAB 7*, 7.6 ed., March 2008.

[37] SICS, "Game History for tac3.sics.se." Website, 2004–2008. Available online, `http://tac3.sics.se:8080/tac3.sics.se/history/`; last visited September 1, 2008.

[38] SICS, "Game History for tac5.sics.se." Website, 2004–2008. Available online, `http://tac5.sics.se:8080/tac5.sics.se/history/`; last visited September 1, 2008.

[39] UMN, "Game History for tac01.cs.umn.edu." Website, 2004–2008. Available online, `http://tac01.cs.umn.edu:8080/tac01.cs.umn.edu/history/`; last visited September 1, 2008.

[40] UMN, "Game History for tac02.cs.umn.edu." Website, 2004–2008. Available online, `http://tac02.cs.umn.edu:8080/tac02.cs.umn.edu/history/`; last visited September 1, 2008.

[41] T. M. Mitchell, *Machine Learning*. McGraw-Hill Series in Computer Science, McGraw-Hill, 1997.

[42] SICS AB, "TAC SCM 2005 Semi-Finals Results." Website, 2005. Available online, `http://www.sics.se/tac/page.php?id=49`; last visited July 21, 2008.

[43] W. Ketter, J. Collins, M. Gini, A. Gupta, and P. Schrater, "Identifying and Forecasting Economic Regimes in TAC SCM," in *AMEC and TADA 2005, LNAI 3937* (H. L. Poutré, N. Sadeh, and S. Janson, eds.), pp. 113–125, Springer-Verlag Berlin Heidelberg, 2006.

[44] SICS, "Trading Agent Competition - FAQ." Website, 2008. Available online, http://www.sics.se/tac/page.php?id=22; last visited December 30, 2008.

[45] SICS, "Trading Agent Competition - TAC Agent Repository." Website, 2008. Available online, http://www.sics.se/tac/showagents.php; last visited December 30, 2008.