Monitoring Reliability and Robustness of Agents for Dynamic Pricing in different (Re-)Commerce Markets

Monitoring der Zuverlässigkeit und Robustheit von Agenten zur dynamischen Bepreisung in unterschiedlichen (Re-)Commerce Märkten

Nikkel Mollenhauer

Universitätsbachelorarbeit
zur Erlangung des akademischen Grades

Bachelor of Science
(B. Sc.)

im Studiengang IT-Systems Engineering
eingereicht am 30. Juni 2022 am Fachgebiet
Enterprise Platform and Integration Concepts
der Digital-Engineering-Fakultät der Universität Potsdam

Gutachter	Dr. Rainer Schlosser
Betreuer	Johannes Huegle
	Alexander Kastius

Abstract

Sustainable recommerce markets are growing faster than ever. However, businesses now face the challenge of having to price the same item three times: A price for the new item, one for its refurbished version as well as the price at which items are bought back from customers. Since these prices are heavily influenced by each other, traditional pricing methods become less effective. To solve this dynamic pricing problem, a simulation framework was built, which can be used to train artificial vendors to set optimal prices using Reinforcement-Learning algorithms. Before employing these trained agents in real markets, they must be tested on their reliability and robustness, as even the smallest mistake by the agent can lead to high costs for the business. This thesis introduces a number of ways that agents can be monitored. We come to the conclusion that the most effective tools are...

...todo

Zusammenfassung

Nachhaltige Recommerce-Märkte befinden sich in stetigem Wachstum. Dies stellt Unternehmen jedoch vor die neuartige Herausforderung, dasselbe Produkt mehrfach bepreisen zu müssen: Preise sowohl für die neue und generalüberholte Version sowie ein Ankaufpreis gebrauchter Ware müssen gesetzt werden. Da diese Preise voneinander abhängig sind, greifen traditionelle Methoden der Preisfindung schlechter. Zur Lösung dieses dynamischen Bepreisungsproblems wurde eine Simulationsplattform gebaut, auf der mithilfe von Reinforcement-Learning Algorithmen künstliche Verkäufer für den Einsatz in realen Märkten trainiert werden können. Bevor dies jedoch geschehen kann müssen die trainierten Modelle auf Zuverlässigkeit und Robustheit überprüft werden, da bereits der kleinste Fehler zu hohen Verlusten des Unternehmens führen kann. Diese Arbeit führt Methoden und Tools ein, die zu einem solchen Monitoring verwendet werden können. Es stellt sich heraus, dass die effektivsten Tools dabei...

Schönerer Name für ärtificial vendors"?

Übersetze ich die beiden Begriffe, oder lasse ich sie englisch?

...todo

Contents

1 Introduction

This thesis builds upon the bachelors project "Online Marketplace Simulation: A Testbed for Self-Learning Agents" of the Enterprise Platform and Integration Concepts research group at the Hasso-Plattner-Institute. Therefore, the project will be referenced and all examples and experiments will have been conducted using its framework.

1.1 Objective of the Thesis

This thesis introduces ways to monitor the *Reliability* and *Robustness* of different agents (rule-based as well as trained using various Reinforcement-Learning (RL) approaches) tasked with dynamically pricing products in a Circular Economy marketplace. Since the terms *Reliability* and *Robustness* can be interpreted differently depending on context and personal experience, we will define our usage in the Reliability and Robustness section. Following the term definitions, we will give a short introduction ans explanation of what a Circular Economy market is (The Circular Economy model) as well as what Reinforcement Learning is and how we employ the technique in our framework (Using the simulated marketplace to train agents).

1.1.1 Reliability and Robustness

1. *Reliability*: With *Reliability*, we describe the ability of an agent to be able to transfer knowledge of a certain type of marketplace and/or against a certain opponent over to a different scenario. If Agent A performs well against Agent B on marketplace M, does it perform the same against Agent C on marketplace M, or against Agent B on marketplace N?

2. *Robustness*: *Robustness* is the property that describes how well an agent performs over a longer period of time. In a real-world marketplace, consistency is key to success, so finding profitability outliers and their causes are a central part of evaluating an agent's Robustness.

1.2 Introduction to the Recommerce platform

1.2.1 The Circular Economy model

The main goal of the aforementioned bachelors project was to develop a realistic online marketplace that simulates a Circular Economy. A market is most commonly referred to as being a "Circular Economy" if it includes the three activities of reduce, reuse and recycle [KRH17]. This means that while in a classical Linear Economy market each product is being sold once at its *new price* and after use being thrown away, in a Circular Economy, recycling and thereby waste reduction is a major focus. In our project, we modelled this by adding two additional price channels, *re-buy price* and *used price*, to the pre-existing *new price* of a product.

The *re-buy price* is defined as the price a vendor is willing to pay a customer to buy back a used product, while the *used price* is defined as the price the vendor sets for products they previously bought back and now want to sell alongside new products (whose price is defined by the *new price*).

In our framework, we implemented a number of "market blueprints" both for classic Linear Economy markets, as well as for Circular Economy markets both with and without rebuy-prices enabled. All market types offer various configurations for the number of competitors: Monopoly, Duopoly and Oligopoly scenarios can be configured, with the Duopoly and Oligopoly scenarios offering rulebased comeptitors specifically built for the respective market scenario.

How big is the difference between Duopoly and Oligopoly actually?

From now on, when mentioning the general *market* or *marketplace*, we are referencing the Circular Economy marketplace with rebuy prices.

1.2.2 Using the simulated marketplace to train agents

After the initial market was modelled the goal was to train agents using different reinforcement-learning algorithms to dynamically set prices on this marketplace, both in monopolistic scenarios as well as in competition with rule-based vendors which set prices following a strict set of pre-defined rules. These rules can range from simply undercutting the lowest competitor's price to more advanced techniques such as price-fixing and -gouging. Furthermore, functionality was added that allows for different Reinforcement learning algorithms to be trained against each other on the same marketplace, as well as functionality for so-called *self-play*, where an agent plays against itself, or more precisely, against its own policy.

Is the following sentence a footnote?

find out if we have such competitors

Reinforcement-learning agents are trained through a process of trial-and-error. They interact with the market through an observable state and an action which influences the following state. Figure 1.1 illustrates the RL-model in the context

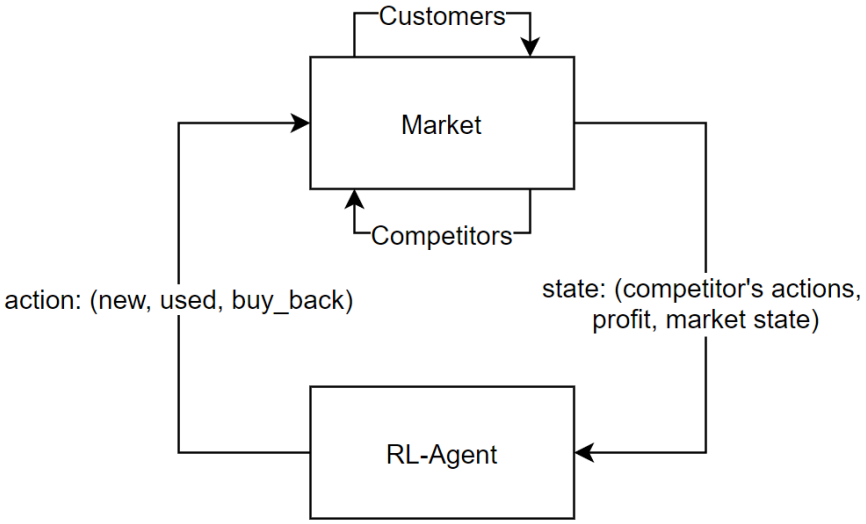

Figure 1.1: The standard reinforcement-learning model in the context of our market.

of our market. The goal of the agent is to maximize the so-called reinforcement signal, which in our case is the profit the agent made during the last cycle, since we want to train agents to maximize profits on real markets. By observing which prices lead to which reinforcement signal, the agents get more profitable over the course of training.

Create a *nicer* diagram with the three prices

2 Related Work

2.1 Approaches to evaluating RL-agents

2.1.1 ...on the fly (while training)

2.1.2 ...after training has finished

3 What makes a good agent?

In this section we want to take a look at what defines an objectively "good" agent, focusing on the dynamic pricing aspect and how our recommerce marketplaces were built to simulate real markets as realistically as possible. This section will be split into two parts - first, we will take a look at Reinforcement Learning Agents, to then try and transfer as much knowledge as possible to the rulebased competitors we specifically built for the framework.

3.1 Overview of market components

3.1.1 Focus on how agents make profit etc.

3.2 How realistic the market is/how realistic it can be

3.2.1 Restrictions for evaluation arising from this

4 Approaches to monitoring agents

In this section we will take a look at the different approaches we took to monitoring agents in our framework, explaining the reasons why we chose to implement specific features and how they help us in determining an agents strengths and weaknesses.

4.1 When to monitor what

Our workflow (which will be explained in more detail in The *recommerce* workflow) can generally be split into two parts when it comes to monitoring and evaluation of agents. In the future, when talking about the *workflow* we will be talking about the process of configuring and starting a training session, where a Reinforcement-Learning agent is being trained on a specific marketplace against competitors. The *workflow* also includes the subsequent collection of data used to evaluate the agent's performance. We are also introducing the term of the *complete agent* in this section, which will be used to refer to both Reinforcement-Learning agents that have been fully trained as well as rule-based agents which do not need training.

1. During training: Having data available as soon as possible without having to wait for a long training session to end is crucial to an efficient workflow. Our framework enables us to analyze and visualize data while a training session is still running . This enables us to filter out agents with sub-par performance prematurely. This can be done using user-defined threshholds or on the basis of previoulsly collected data (e.g. only keeping agents that are performing better than at least 50% of other agents after the same amount of training in comparable scenarios.)

2. On complete agents: After a training session has finished we have a complete and final set of data available for an agent, which enables us to perform more thorough and reliable tests. These can include simulating runs of a marketplace to gather data on the agent's performance in different scenarios and against different competitors, or running a static analysis of the agent's policy in different market states. The tools available for trained agents are in the same way also usable on rule-based agents.

Implement this feature. It should at least be able to temporarily halt the training to start an intermediate monitoring session

Also implement this feature. Allow users to set rules for when a training session should be terminated if the agent is not performing well at a certain point. Also, it should be able to give the program a set of datapoints and have it set rules if possible

In the following sections, we will take a look at all of the tools our framework provides for monitoring agents, distinguishing between the two general types of monitoring mentioned above. The goal of this section is to give a short overview of each tool, how and why they were implemented and what value they offer to the framework as a whole and to the analysis of agent reliability and robustness in particular. We will also discuss features that are currently not available, explaining how they could benefit the entire workflow or enrich the overall experience.

4.2 Monitoring during a training session

When talking about monitoring agents during a training session, we are of course talking about Reinforcement-Learning agents, since Rule-Based agents always perform the same and cannot be trained. Monitoring agents while they are still being trained enables us to be more closely connected to the training process. Ultimately, the goal of such monitoring tools is to be able to predict the estimated "quality" of the final trained agent as reliably as possible while the training is still going. Users must however be careful when interpreting the results of monitoring tools that work on "incomplete" agents, we will go more into this in Interpreting the results.

Is this a footnote? This info should be obvious, but maybe should be mentioned for completeness?

4.2.1 Tensorboard

The *Tensorboard* is an external tool from the from the Reinforcement-Learning library *Tensorflow*[1]. With just a few lines of code a training session can be connected to a Tensorboard. We are then able to pass any number of parameters and metrics we deem interesting or useful to the Tensorboard, which then offers visualizations for each of them, updating live as the training progresses. Though the Tensorboard does not interpret data in any way, it is an immensely useful tool for quickly and easily recording data and offering a first rough comparison of competitors in the market.

Question for the tutors: Do I give an example of such code?

Create some sample diagrams. Perhaps these can be re-used in the workflow section, so they could perhaps be moved to the Appendix for reference?

4.2.2 Live-monitoring

Unlike the Tensorboard, the monitoring tools summarised under the term *live-monitoring* were completely and from the ground up built by our team. For most of the visualizations, the *matplotlib*[2] library was used. Live-monitoring aims to be

1 https://www.tensorflow.org/
2 https://matplotlib.org/stable/index.html

more configurable and in-depth with the metrics it offers than the aforementioned Tensorboard. During a training session, "intermediate" models, as we will call them, are being saved after a set number of episodes. These models contain the current policy of the agent and can be used the same as models of complete agents, the only difference being the quality of the agent, as those with a lower amount of trained episodes generally perform worse. These intermediate models can then be used by a range of monitoring tools available to us. Since the models only contain the current policy of an agent but not the history of states and actions preceding the model, we need to run separate simulations on these models to be able to analyze and evaluate them. For this, we utilize our *agent-monitoring* toolset, which will be explained in more detail in the Agent-monitoring section.

make the current live-monitoring so that graphs are actually being created while the training is running, not like it is now with graphs only being created afterwards

Have we introduced the concept of episodes before? Should a Glossary be introduced, or do we do this stuff in the introductions?

Does this need a citation?

4.3 Monitoring complete agents

The following tools offer a wide range of functionalities for monitoring complete agents. We can use these tools to both determine the reliability and robustness of one certain agent, but also to compare different agents, either during a joined monitoring session or by comparing results of monitoring runs on identically parametrized market situations.

4.3.1 Agent-monitoring

Using the *Agent-monitoring* toolset, users can configure a custom market simulation, using the following parameters:

1. Episodes: This parameter decides how many independent simulations are run in sequence. At the start of each episode, the market state will be reset. Within an episode, each vendor runs through 50 timesteps, during each of which a price is set (depending on the chosen economy type, this can include a rebuy price for used items) and a set number of customers interact with the vendors.

2. Plot interval : A number of diagram types enable the user to view averaged or aggregated data over a period of time. The plot interval parameter decides the size of these intervals. Smaller intervals mean more accurate but also more convoluted data points. Computational time also increases with a smaller interval size.

Currently, the plot interval is not used by anything, since the statistics-plots have been "removed" in favor of the tensorboard plots

3. Marketplace: Using this parameter, the user can set the marketplace on which the monitoring session will be run. There are marketplaces available for each combination of the following features:

   a) Marketplace type: Linear Economy, Circular Economy, Circular Economy with rebuy-prices

   b) Market environment: Monopoly, Duopoly, Oligopoly

4. Agents: Depending on the chosen marketplace, only a select number of valid agents can be chosen to be monitored, as each agent is built to interact with a specific type of marketplace. First off, all agents belong to one of the two major categories: *Reinforcement Learning agent* or *Rulebased agent*.

It is important to note that while multiple agents are being monitored simultaneously, each agent operates on a market of its own. This means that while each agent receives the same initial market state at the start of an episode, and all agents play against the same competitors, monitored agents do not play against each other. The goal of the *Agent-monitoring* tool is to be able to compare different agents playing under the same circumstances to determine which ones might be "better" than others. This would not be achievable with monitored agents playing against each other, as the assumptions about the market state would differ for each agent.

During each episode and for each vendor, all actions and market events are being recorded using *watchers*. At the end of a monitoring session, the collected data is evaluated in different visual formats. First of all, all data that would be available to see in the *tensorboard* during a training session is visualized using simple line graphs .

Do I keep this new word? If yes, it should be explained at least a little bit!

Are these the ones with the probability distributions?

4.3.2 Exampleprinter

The *Exampleprinter* is a tool meant for quickly evaluating an agent in-depth. When run, each action the monitored agent takes is being recorded, in addition to market states and events, such as the number of customers arriving and the amount of products thrown away. For certain market scenarios such as the *CircularEconomyRebuyPriceDuopoly*, which simulates a circular economy model with rebuy prices in which two vendors compete against each other, these actions and events are also being summarised as an animated graphic, where each time-step is being illustrated.

Add in a graphic here. Perhaps have two of the time-steps to "simulate" the animation

4.3.3 Policyanalyzer

The *Policyanalyzer* is our only tool which does not simulate a run of the market scenario. Instead, the tool can be used to monitor an agent's reaction to different market events. The user can decide on up to two different features to give as an input, such as the competitor's new and used prices, and the Policyanalyzer will feed all possible input combinations to the agent and record its reactions.

Diagram

4.4 Features for the future

This section is meant as a collection of ideas and approaches for tools that could further enhance the workflow, but which have not been implemented and therefore not been tested for their feasability and usefulness.

5 The *recommerce* workflow

The main goal of the bachelor's project is to provide a simple-to-use but powerful interface for training Reinforcement-Learning algorithms on highly configurable markets for users in both a research and a business context. To achieve this, multiple components had to be developed and connected to create the workflow we now provide. This section will go over the most important parts of the workflow, focusing on the way each of them supports the monitoring capabilities of the framework.

Better word for components

5.1 Configuring the run

Unarguably, the most important part of the whole workflow is its configuration. Without it, each simulation and training session would produce similar, if not the same results. By tweaking different parameters of a run, market dynamics can be changed and agent performance be influenced. The goal of our monitoring tools is to enable users to assess the extent to which each parameter influences certain characteristics of the training and/or monitoring session, and to enable them to make informed decisions for subsequent experiments.

Is this "zu wertend"?

Ultimately, all configuration is done using various .json files which contain key-value pairs of the different configurable items. We further differentiate between different groups of configurations, which means that hyperparameters influencing the market, such as maximum possible prices or storage costs, are being handled separate from parameters needed for Reinforcement-Learning Agents, such as their learning rates, allowing users to make informed decisions when tweaking parameters involving different parts of the framework.

Our main goals when building our configuration tools were to make sure that users are able to quickly and safely configure their experiments in a way that is also easily reproducible. To be as user-friendly as possible, a lot of validation logic has been implemented to be able to catch invalid configurations as early as possible, making sure that whenever a training session is started, it is confirmed that the configuration is valid an the training will not fail due to wrongly set parameters at any point. Since our project has also been deployed to a remotely accessible, high-performant machine, we decided on creating an approachable web-interface

for our configuration tool, which can now be used for both configuring, starting and evaluating experiments.

5.1.1 The webserver/Docker-API

Example screenshot of the webserver. Configuration and running page?

On our webserver, users can upload .json files containing any number of (valid) key-value pairs, or create configurations using a form that always makes sure that it contains only valid values. For example, if the user chooses a circular economy market scenario with rebuy prices enabled, the user will not be able to configure an agent that cannot operate on such a market.

Aside from uploading or configuring complete, ready-to-go configurations, users can also choose to upload or create incomplete configurations, which can then be combined with other configuration objects to create a complete, valid configuration. This allows users to create multiple incomplete configurations containing only select parameters and then test all permutations of these configurations to observe the effect the different parameter combinations have on the experiment.

Example of a mix and match of parameters

An example of one such approach can be found below:

The webserver then also offers the option of starting multiple runs of the same configuration simultaneously. Most of the times this is recommended, as singular runs are prone to misleading results due to the trained agents training with specific market states. By running the same configuration multiple times using different starting circumstances for both the agent that is to be trained as well as its competitors, a configuration's effect on the agent's performance can be disconnected from the pseudo-random market states which influence the vendor's decision making processes.

Is this perhaps too much interpretation for this section?

A configuration can be interpreted as producing "reliable" agents if any number of simulations all converge on similar agent performances (meaning similar mean rewards). The more singular runs diverge from the mean , the more the agent's performance is dependent on the initial market state, meaning that its performance is less stable, as it will produce unpredictable results on new, unknown market scenarios. A higher number of runs that produce agents which perform on similar levels of performance means that the specific configuration, to be axact the parameters responsible for the Reinforcement-Learning algorithm, produces "reliable" agents: No matter the state of the market, the agent can always be expected to perform on the same level that was observed during the training process, setting prices that lead to similar profits. Reliability is therefore one of the most important factors in determing an agent's overall quality, as the ultimate goal is always to be able to deploy the agent onto the real market, which will always differ from the simulated environments the agent experienced during training, simply due to the

mean or median? What do I use?

fact that real markets are being influenced by so many more parameters than one could model in our, or for that matter any simulation framework.

5.2 Choosing what to show the user during training

During the training process, we of course record all actions the different vendors take, both for the training agent and its competitors, be they also Reinforcement-Learning Agents or Rulebased agents. Market states and events are also being recorded, to be able to match them to the agent's actions later on. Users of course want to have an indicator of how the training run is going so far, both to inform themselves about the total progress of the training, but also about the quality of the agent they are training. We now have to decide what information is shown to the user, making sure that the user is well informed, whilst also holding back on displaying too much information at once to prevent confusion, as the underlying data and states are ever-evolving, leading to many datapoints being invalidated shortly after their creation.

In When to monitor what we introduced the idea of running monitoring sessions while a training session is still runnning. To enable this, we need to save so-called "intermediate" models of our agents at set intervals. These intermediate models contain the current policy of an agent, which we can then use in conjunction with our various monitoring tools to gather first information on the agent's performance, to then try and predict future characteristics. In some cases, we may find that the agent is developing in a wrong direction, leading to reduced profits or irrational policies. We can use this information to terminate a training session preemptively to save resources and time to start anew.

In the following section, we will present a number of datapoints that could be chosen to be shown to the user, together with an analysis on where their strengths in informing the user lie.

This section sort of doesn't fit in with the flow of the points above and below

5.2.1 Informing the user

The first choice and a save bet when looking at datapoints that will be shown to users is the current training progress. Each training run is defined in its length by a parameter that sets the number of episodes that should be simulated. Each episode consists of an independently initialized market state on which another configurable amount of steps is run. Within each step, the vendors take turns in observing the current market state and then setting their prices for the new, used, and depending on the chosen marketplace, a rebuy price.

5.3 After training/Complete agents/Why do we even monitor after training?

After training has completed, the most crucial phase of the workflow starts. As has been mentioned before, the end-goal of users using the simulation framework is to be able to deploy trained agents into real markets to set prices independently of human inputs or guidance. Starting from this premise, the goal of the training process is therefore to model the marketplace as realistically as possible, so that the trained agents can transfer the knowledge from their training to the real market "without noticing a difference". Before deploying agents to the real market, they need to be tested on their reliability and robustness, to make sure that their policies are not just effective under certain circumstances, but that they are also able to adapt to different market scenarios and behave accordingly.

5.3.1 Which datapoints prove to be/are most effective?

6 Interpreting the results

6.1 Graphs and diagrams are available...

6.1.1 ...comparing with other agents/models

6.1.2 ...which hyperparameters influence the results in what ways?

6.1.3 ...can we augment e.g. Grid-Search with our analysis?

6.1.4 -> Would need to make results "machine-readable" again

7 Conclusions & Outlook

Bibliography

[KRH17] Julian Kirchherr, Denise Reike and Marko Hekkert. **Conceptualizing the circular economy: An analysis of 114 definitions**. *Resources, Conservation and Recycling* 127 (2017), 221–232. ISSN: 0921-3449. DOI: https://doi.org/10.1016/j.resconrec.2017.09.005. URL: https://www.sciencedirect.com/science/article/pii/S0921344917302835 (see page 2).

Declaration of Authorship

I hereby declare that this thesis is my own unaided work. All direct or indirect sources used are acknowledged as references.

Potsdam, 23rd May 2022

Nikkel Mollenhauer