Reinforcement Learning for Pricing Strategy Optimization in the Insurance Industry

Elena Krasheninnikova[a], Javier García[b], Roberto Maestre[a], Fernando Fernández[b]

[a]*BBVA Data & Analytics*
[b]*Universidad Carlos III de Madrid*

Abstract

Pricing is a fundamental problem in the banking sector, and is closely related to a number of financial products such as credit scoring or insurance. In the insurance industry an important question arises, namely: how can insurance renewal prices be adjusted? Such an adjustment has two conflicting objectives. On the one hand, insurers are forced to retain existing customers, while on the other hand insurers are also forced to increase revenue. Intuitively, one might assume that revenue increases by offering high renewal prices, however this might also cause many customers to terminate their contracts. Contrarily, low renewal prices help retain most existing customers, but could negatively affect revenue. Therefore, adjusting renewal prices is a non-trivial problem for the insurance industry. In this paper, we propose a novel modelization of the renewal price adjustment problem as a sequential decision problem and, consequently, as a Markov decision process (MDP). In particular, this study analyzes two different strategies to carry out this adjustment. The first is about maximizing revenue analyzing the effect of this maximization on customer retention, while the second is about maximizing revenue subject to the client retention level not falling below a given threshold. The former case is related to MDPs with a single criterion to be optimized. The latter case is related to Constrained MDPs (CMDPs) with two criteria, where the first one is related to optimization, while the second is subject to a constraint. This paper also contributes with the resolution of these models by means of a model-free Reinforcement Learning algorithm. Results have been reported using real data from the insurance division of BBVA, one of the largest Spanish companies in the banking sector.

Keywords: Pricing strategy optimization, Reinforcement learning

1. Introduction

The main goal of pricing is to set prices at which a company will sell its products and services (Phillips, 2005; Hinterhuber, 2017). The margin between the price and the total cost of the product or service is the revenue. If a company wants to adjust the price taking into account customer characteristics, product features, market status, risk, price sensitivity, etc., the Pricing

Email addresses: elena.krasheninnikova@bbvadata.com (Elena Krasheninnikova), fjgpolo@inf.uc3m.es (Javier García), roberto.maestre@bbvadata.com (Roberto Maestre), ffernand@inf.uc3m.es (Fernando Fernández)

Strategy Optimization framework defines the techniques, methods and process to achieve this goal.

Insurance is one example of an industry which deals with pricing. In the insurance industry an important question arises, namely: how can renewal insurance prices of existing customers be adjusted? The answer to this question requires addressing two different conflicting objectives: (i) increase retention of the existing customers and (ii) increase revenue. Intuitively, high renewal insurance prices may increase revenue, however this may also cause many customers to terminate their contracts. A priority for an insurance company is to avoid losing customers, and customer retention is critical to sustaining growth and maximizing revenue. Getting customers through the door only leads to a loss of revenue, and insurance companies need a pricing strategy to effectively retain customers. One may think that this strategy is to offer low renewal prices. Undoubtedly, such a strategy will induce many customers to stay loyal, but it will also negatively affect revenue. Therefore, striking a balance between revenue growth and customer retention presents itself as a difficult challenge.

In this paper we consider the renewal price adjustment problem as a sequential decision process. In particular, this problem is considered as follows: The insurer company has a portfolio of customers. Then, when it is time to renew the customer prices, the insurer takes the first client from the portfolio and makes a decision: which renewal price to offer him/her taking into account the current situation of the company (i.e., current revenue, retention). Whether the customer accepts the renewal price or not, the insurer's decision leads to the company to a new situation (e.g., if the customer does not accept the renewal price, the insurer company will have one customer less and, hence, lower revenue). Additionally, we can know if the insurer's decision has been good or bad (e.g., if the insurer's decision increases the revenue we can consider that the decision has been good, and bad otherwise).Thus, one can take into account the utility of each particular decision. Then, given the new situation of the company, the insurer takes the next customer from the portfolio, makes a decision, and so on until the insurer makes a decision for each of the customers in the portfolio. In this way, the problem can be seen as a succession of situations (states), decisions (actions), and utilities (rewards). Therefore, we consider that Markov decision processes (MDP) provide a more appropriate modelization of this problem. Such modelization introduces two benefits: it takes into account the long-term effects of each decision, and it also takes into account the expected value of each decision. The decision of each particular customer should be optimized taking into account the sequential process involved and the optimization criteria selected (e.g., maximization of the revenue). Thus, we might suggest a new renewal price whose immediate reward is lower, but leads to more likely or more profitable rewards in the future. These considerations are taken into account automatically by the policies generated for an MDP.

Keeping these benefits in mind, in this paper we propose the use of Reinforcement Learning (RL) techniques to learn these policies in such MDPs. RL (Sutton & Barto, 1998) is a type of machine learning where the main goal is that of finding an action policy that makes the agent act optimally in a given environment, generally formulated as an MDP. The present study analyzes the use of two different strategies to adjust the renewal prices by RL. In the first strategy we formulate the proposed problem as an MDP with a single criterion to be optimized. In this MDP, we aim to maximize revenue, while analyzing the impact of said maximization on customer retention. However, with this problem we find two conflicting objectives which require a balancing strategy. For this reason, in the second strategy to be analyzed, the problem is formulated as a Constrained MDP (CMDP) with two criteria: the first is to be optimized, while the second is subject to a constraint. In this CMDP, we aim to maximize the revenue subject to the client retention level not falling below a given threshold. The latter formulation of the problem better models the issues currently facing insurance companies. Therefore, we raise the problem as a multi-objective problem, in contrast to other approaches that only focus on maximizing the revenue (Chizhov & Borisov, 2011; Rana & Oliveira, 2015). These two strategies will be applied to real data provided by the insurance division of BBVA, one of the largest Spanish companies in the banking sector.

Therefore, the main contribution of this paper is the novel modelization of the renewal price adjustment problem as an MDP and a CMDP, in which the particular situation of each customer but also the global situation of the company are taken into account for each decision. For each decision, the global situation of the company is updated as customers accept or reject the renewal prices, which will influence future decisions. Additionally, each decision is made paying attention to two objectives: the revenue, and customer retention. The second contribution is the resolution of these models by means of an existing model-free RL algorithm: VQQL (Fernández & Borrajo, 2000). The learned policies can be used by the insurers as a decision support system or a price recommenders that allows them to decide the renewal price for the customers. To the best of our knowledge, this is the first time that (i) this problem is modeled in this way and (ii) it is solved by means of model-free RL techniques.

The remainder of the paper is structured as follows. Section 2 summarises relevant related work. Section 3 introduces the background on RL needed to understand the proposed research. This includes basic notions from RL field. Section 4 provides a detailed description of the problem to be solved by RL (i.e., Pricing Strategy Optimization for renewal portfolios), and also describes the real data used in this research. Section 5 formulates the problem of adjusting the renewal prices as an MDP and as a CMDP. Such modeling requires defining of all the elements of an MDP: the state and action spaces, and the reward and the transition functions. Section 6 introduces the algorithm used for learning: the VQQL algorithm. Finally, Section 7 shows the learning capabilities of the algorithm and Section 8 concludes and introduces future research.

2. Related Work

From a machine learning perspective, pricing represents a supervised prediction problem (Lowe & Pryor, 1996; Wuthrich & Buser, 2016; Parodi, 2012; Spedicato et al., 2018; Yeo, 2009), i.e., a model is learned that receives as input the features of a particular service or product (e.g., name, category,. . .) and predicts its price. However, we consider that such approaches have two main drawbacks. On the one hand, they do not consider the existing situation of the company. The price of each service or product not only depends on its particular features but also on the global situation of the company at that moment. For example, in the case of the insurance industry, a customer's renewal price might depend on the number of customers that the company already has. If the company has many customers, it can take risks offering a high renewal price that may be rejected. However, if the company has few customers it cannot risk losing more of them, so it should offer a low renewal price that increases the probability of being accepted. On the other hand, they do not take into account that the global situation of company changes over time as decisions are performed, and that current decisions impact future decisions and opportunities. In other words, they do not take into account the *sequential nature of the problem*. Contrarily, in this paper we consider the renewal price adjustment problem as a sequential decision process.

This is not the first time that pricing problems are modeled as sequential decision making (Cesa-Bianchi et al., 2006; Blum & Hartline, 2005; Trovò et al., 2018). For instance, Cesa-

Bianchi et al. (2006) address a similar problem to the considered in this paper. In that problem, a vendor sells a product to a sequence of customers whom he attends one by one. To each customer, the seller offers the product at a price he selects. The customer then decides to buy the product or not. The goal of the seller is to achieve an income almost as large as if he knew the maximal price each customer is willing to pay for the product. Similarly, Blum & Hartline (2005) compute the best price to offer to bidders arriving one at a time in an auction. In all those cases, the problem is considered as a type of multi-armed bandit problem with only one state in which the reward perceived by the agent is the price offered to each customer. However, these problems do not take into account that decisions can also modify the state in which the system is. In contrast, we consider a decision taken in a state leads the system to a new state which has an impact in the future decisions to be made.

Reinforcement Learning has been also applied to pricing problems but the literature is limited and sparse (Chizhov & Borisov, 2011; Rana & Oliveira, 2015; Raju et al., 2006). For instance, Raju et al. (2006) use RL to determine dynamic prices in an electronic retail market. Instead, Rana & Oliveira (2015) use RL to dynamically adjust the price of interdependent perishable products or services. In both cases, a state is described as the amount of inventory available for each product or service, but such state description does not include customer features, nor the global situation in which the company is. In other cases, the customer features are taken into account, but not the global situation of the company (Kamishima & Akaho, 2011). In our case, a state describes the particular situation of a given customer and the global situation of the company as explained in Section 5. Including particular customer features in the state description allows personalized pricing for each customer, while including global features of the company allows taking into account the situation of the company in each decision.

Finally, all these papers use profit as the main metric of success, but do not consider other optimization criterion such as customer retention. In summary, previous work address a different problem from the one we are considering here. In this paper, we are focusing on the modelization of the renewal price adjustment problem as a sequential decision problem. In such a problem there is a portfolio of customers who are offered a new renewal price paying attention to two different objectives: revenue, and customer retention. In addition, for each decision not only the particular situation of each customer is taken into account, but also the global situation of the company.

3. Background on Reinforcement Learning

A RL environment is typically formalized by means of an MDP (Barto et al., 1995; Sutton & Barto, 1998; Wiering & van Otterlo, 2014). An MDP consists of a set of states S, a set of actions A available from each state, the reward function $R : S \times A \to \mathbb{R}$ which assigns numerical rewards to transitions, and transition probabilities $T : S \times A \times S \to [0, 1]$ that capture the dynamics of a system. In this paper, we consider S and A to be infinitely large bounded sets, i.e., we consider MDPs where both the states and actions are continuous. We also consider discrete-time MDPs, i.e., at each discrete time step n the learning agent perceives a state $s \in S$ and takes an action $a \in A$ that leads it to the next discrete time step $n + 1$, with $n \in \{1, 2, 3, \dots\}$. The goal is to learn a policy π, which maps each state to an action, such that the return $J(\pi)$ is maximized:

$$J(\pi) \quad \sum_{n=0}^{N} \gamma^n r_n \tag{1}$$

where r_n is the immediate reward received in step n, and γ is the discount factor and affects how much the future is taken into account (with $0 \leqslant \gamma \leqslant 1$). We assume that the interaction between the learning agent and the environment is broken into episodes, where N is a time instant at which a terminal state is reached, or a fixed length for a finite horizon problem. Traditional methods in RL, such as TD-learning (Sutton & Barto, 1998; Wiering & van Otterlo, 2014), typically try to estimate the return (sum of rewards) for each state s when a particular policy π is being performed. This is also called the value-function $V^{\pi}(s) \quad E[J(\pi)|s_0 \quad s]$. The value of performing an action a in a state s under policy π is represented as $Q^{\pi}(s,a) \quad E[J(\pi)|s_0 \quad s, a_0 \quad a]$- this value represents the estimated return, i.e. sum of rewards, the system will receive when it performs action a in the state s, and follows the policy π thereafter. The Q-function is also called the action-value function.

The Q-learning algorithm (Watkins, 1989) is one of the most widely used for computing the action-value function. Given any experience tuple of the type $\langle s_n, a_n, s_{n+1}, r_n \rangle$ - where s_n is the current state, a_n is the action taken in that state, s_{n+1} is the state achieved when executing a_n from s_n, and r_n is the immediate reward - it updates the Q-function following Equation (2)

$$Q^{\pi}(s_n, a_n) \leftarrow Q^{\pi}(s_n, a_n) + \alpha(r_n + \gamma \max_{a} Q^{\pi}(s_{n+1}, a) - Q^{\pi}(s_n, a_n)) \tag{2}$$

where γ is the discount factor, and α is a learning rate. To correctly approximate the Q-function, the Q-learning algorithm uses an exploration strategy (e.g., ϵ-greedy, softmax) as a balance between the exploration of random unexplored actions and the exploitation of the ongoing learned policy (Tijsma et al., 2016). For problems with a continuous action space, selecting the optimal action with respect to the Q-function is non-trivial, as it requires finding a global maximum for a function in a continuous space. One way to extend Q-learning to continuous state-action space, is to discretize the environment in order to reduce such space (Uther & Veloso, 1998; Lee & Lau, 2004; García et al., 2010). Many different discretization methods have been proposed, but in this paper we focus on the use of Vector Quantization (VQ) (Fernández & Borrajo, 2000; Fernández et al., 2005) to discretize the state space and the use of an uniform discretization for the action space (Section 6). In such discrete state-action space, it is usual to use a tabular representation of the Q-function. Such Q-tables have as many rows as states existing in the domain, and as many columns as actions that can be executed in each of these states. Each of the positions in the Q-table is the value of the Q-function for the corresponding state and action.

While in typical RL environments the objective space consists of only one dimension, in practical applications, there might be several possibly conflicting objectives requiring a strategy that mediates between them (Vamplew et al., 2011; Roijers et al., 2013). In multi-objective RL the objective space consists of two or more dimensions. Therefore, regular MDPs are generalized to multi-objective MDPs or MOMDPs. One of the consequences is that MOMDPs provide a vector of rewards instead of a scalar reward, i.e.,

$$\overrightarrow{R(s,a)} \quad (R_1(s,a), \ldots, R_q(s,a)) \tag{3}$$

where q represents the number of objectives. In the same way, the scalar Q-value turns into a Q-vector that store a Q-value for each objective, i.e.,

$$\overrightarrow{Q^{\pi}(s,a)} \quad (Q_1^{\pi}(s,a), \ldots, Q_q^{\pi}(s,a)) \tag{4}$$

where q represents the number of objectives. Multi-objective RL has been successfully used for urban traffic light control (Shengchao, 2010; Khamis & Gomaa, 2012), cognitive space communications (Ferreira et al., 2017), or robot control (Ahmadzadeh et al., 2014; Ansari et al., 2018).

A particular type of MOMDPs are the CMDPs (Geibel, 2006). In CMDPs the learning agent maximizes one of the objectives subject to constraints on the others as in Equation (5).

$$\begin{aligned} &\max Q_1^\pi(s,a) \\ &s.t.\ Q_i^\pi(s,a) \geqslant \theta_i,\ i \quad 2,\dots,q \end{aligned} \tag{5}$$

CMDPs can be used to model a wide variety of different real problems. For instance, it can be used to maximize the revenue on online advertising, while considering the constraint of budget limits (Du et al., 2017), or, in robot control, to maximize the probability of reaching a target location within a temporal deadline (Carpin et al., 2014). As explained in Section 5, CMDPs is also used to model the problem considered in this paper.

4. Pricing Strategy Optimization for Renewal Portfolios

This section provides a detailed description of the problem to be solved by RL. It also includes a description of the data used in this research.

4.1. Problem Description

Generalized Linear Models (GLM) is the standard in the pricing industry (Murphy et al., 2000) as it is easy to fit and interpret. GLMs are a rich class of statistical methods that include linear regression, logistic regression, log-linear models, poisson regression and multinomial response model (Agresti, 2015). From a mathematical point of view, pricing can be solved using GLM as a general approach. For instance, we might use standard linear regression to learn a model to predict the price for a given customer and product. We can define a GLM (Nelder & Wedderburn, 1972; Lee & Nelder, 2003) as a relation between dependent and independent variables which takes the following form:

$$\varphi \quad g(\mu) \quad \mathbf{X}\beta \tag{6}$$

where $g(\mu) \equiv \mathbb{E}(Y)$, g is the link function, $\mathbf{X}$ is the model matrix and β is the vector of unknown parameters, with $\mathbf{Y}$ being independent and distributed as some exponential family distribution. Note that the standard linear regression model is a special case of GLM where the link function, g, is the identity function.

While setting a price for a given customer and product can be solved with standard linear regression models, the pricing strategy optimization deals with two main components: on the one hand, the pricing model represented by Equation (6) and, on the other, the customer's degree of acceptance of a given price (sensitivity to the price).

Customer sensitivity represents a valuable piece of information to correctly adjust the price and simulate several probabilistic scenarios. As the grade of acceptance of a given price can be represented as a probability function, it can use a logistic regression model, i.e., a GLM with a logistic link function (Germán, 2007) to represent the probability of renewal for a given price. The distribution of the target variable (i.e., if the customer will renew the policy) is set to be the binomial distribution. The mean of the binomial distribution, that is, the prediction generated by the model, is the probability that the event will occur. Equation (7) represents the *logit* function (Kleinbaum & Klein, 2010):

$$\eta \quad \ln\left(\frac{\rho}{1-\rho}\right) \quad \mathbf{X}\beta \tag{7}$$

where η is the logit or log-odd function and ρ is the probability of renewal, $\mathbf{X}$ is the model matrix and β is the vector of unknown parameters. Knowing the GLM model parameters, β, we can obtain the probability of renewal directly from the above equation:

$$\rho \quad \frac{\mathrm{e}^{\mathbf{X}\beta}}{1+\mathrm{e}^{\mathbf{X}\beta}} \quad \frac{1}{1+\mathrm{e}^{\ \mathbf{X}\beta}} \tag{8}$$

The logistic regression model in Equation (8) predicts the probability acceptance for a given renewal ratio and customer. The renewal ratio, v, is the ratio at which customers renew the price of their insurance. This renewal ratio decrements the price when $v \in [0.0, 1.0)$ or increments the price when $v \in (1.0, 2.0]$. Therefore, when v is equal to 1.0 it represents neither incremental nor decremental, i.e., the same price of renewal.

As we saw, the optimization of a pricing strategy requires handling two main goal functions:

1. Revenue

$$f_1(\mathbb{v}) \quad \sum_{i \in \mathbb{v}} \mathcal{B}(\rho_i(v_i)) * price_i * v_i \tag{9}$$

   where $\mathcal{B}$ represents the Bernoulli distribution for probability ρ_i of acceptance of a given renewal ratio v_i for the i-th customer in the portfolio, and $price_i$ is the current price paid by the i-th customer in the portfolio to the insurance company with $f_1 \in \mathbb{R}$.

2. Retention

$$f_2(\mathbb{v}) \quad \frac{1}{N} \sum_{i \in \mathbb{v}} \rho_i(v_i) \tag{10}$$

   where ρ_i is the probability of acceptance of a given renewal ratio v_i for the i-th customer in the portfolio, and N is the total number of customers in the portfolio with $f_2 \in [0, 1]$.

Therefore, although GMLs are the standard in the industry to model the pricing problem as described in Equation 6, such modelization adopts a static view of the problem and treats it as a linear prediction problem. Furthermore, the addition of the probability of customer acceptance transforms the problem into a stochastic problem which makes it difficult to solve it as a linear problem. For these reasons, what we suggest in this paper is precisely that it is more appropriate to model the renewal price adjustment problem as a sequential stochastic decision problem as proposed in Section 5, in which the customers are attended one by one, and in which each decision leads the system to a new state. Additionally, we assume the underlying model dynamics are completely unknown for the learning agent. This makes necessary the use of other types of techniques such as those based on *model-free* RL. In this paper, the proposed approach is able to go *model-free* and learn to price from data generated from a dynamic environment. In fact, the proposed approach fits in better with what happen in the real life, in which the insurer offers renewal prices to a sequence of customers whom he attends one by one.

Finally, one of the important points is the consideration of the business constraints that prices may have. In our case, we consider such constraints C_i are defined for each customer i in the portfolio. It represents the ranges in which each renewal ratio can be chosen, i.e., $v_i \in [\mathrm{C}_{i0}, \mathrm{C}_{i1}]$. These ranges can be adjusted by taking into account the company's objectives, marketing campaigns, changes in legislation or even the competence of other insurance companies. As an example, the lower limit of these ranges can be reduced to offer lower renewal prices in response to a market of aggressive competition or marketing campaigns that aim to retain the highest number of customers.

4.2. Dataset Preparation: The Synthetic Dataset

The real data that we have from BBVA are a dataset of 83200 decisions with 11 features from a home insurance portfolio. We will use it to validate the proposed framework of usage of RL. However, during the research stage we will use a synthetic dataset in which only the main features of the real dataset are taken into account. The selection of these features has been proposed by insurer experts from BBVA who have discarded those features that were not relevant for the considered task. Additionally, the values for the customer features in the synthetic dataset are generated by drawing from a model fitted to the real data, ensuring that the synthetic and the original dataset have the same statistical properties, but without compromising the privacy of the original dataset. The final synthetic dataset has 7 features as shown in Table 1. Table 1 shows the feature names, the feature types (numeric or nominal), the mean value and standard deviations of the numeric features (with the notation $A \pm B$, being A the mean and B the standard deviation), and a brief description of the corresponding feature.

Table 1: Synthetic dataset

Feature	**Type**	**Mean**	**Description**
price	Numeric	207.1 ± 89.1	Insurance price
tier	Nominal	$-$	Financial segment of customer
v	Numeric	1.1 ± 0.1	Renewal ratio
$\rho(v)$	Numeric	0.7 ± 0.2	Probability of customer acceptance
C	Numeric	$[0.9 \pm 0.1; 1.1 \pm 0.1]$	Range renewal ratio
gain	Numeric	224.7 ± 97.9	$price * v$
class	Nominal	$-$	Price acceptance $\sim \mathcal{B}(\rho(v))$

Therefore, in this synthetic dataset we consider five main features for each customer: *price*, *tier*, v, ρ, and C. Feature *price* is the price given from Equation 6, i.e., the insurance price. *tier* is a well known feature in the insurance sector that indicates a financial segment of a customer. Customer segmentation is usually performed through a clustering process considering several customer features (e.g., age, number of claims during the last year) Abolmakarem et al. (2016). In this way, in order to make a decision for a given customer, the company does not analyze each one of its particular characteristics, but simply the segment to which it belongs to. It is expected that all customers of the same group react in the same way to company decisions. Therefore, companies can even customize the decisions for each segment (e.g., marketing strategies, commercial plans, renewal prices). In this paper, the segment to which a customer belongs to is provided by the insurance company and it indicates a different customer risk. In particular, we consider four segments and, accordingly, we consider four values for this feature: *tier*1, *tier*2, *tier*3 and *tier*4, where the greater the value of *tier*, the greater the risk of the customer. Feature v is the renewal ratio offered to the customer and $\rho(v)$ is an artificially created probability of customer acceptance for that renewal ratio used for experimentation purposes. Finally, feature C are the business constraints, i.e., the range in which the renewal ratio v can be chosen for that particular customer. The features *gain* and *class* are derived from the previous features. In particular, *gain* is the new insurance price offered to the customer (i.e., $gain \quad price * v$) and *class* is the renewal acceptance class sampled from $\sim \mathcal{B}(\rho(v))$. Figure 1 graphically represents the synthetic dataset including these features. Note that, in Figure 1, higher prices are applied to customers with higher risk. Both variables *price* and ρ exhibit a non-lineal increment.

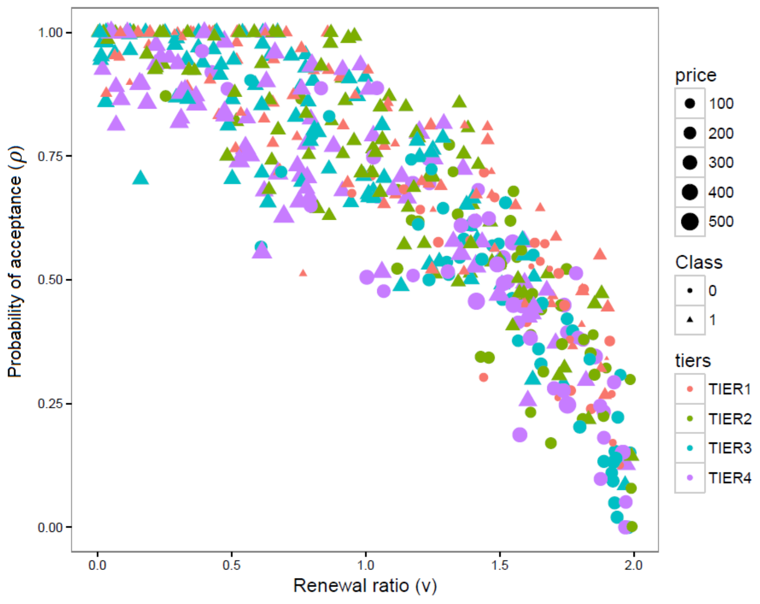

Figure 1: Synthetic dataset with *price*, *tier*, v, ρ and *class*

This synthetic dataset is used for two purposes. The first is to train the logistic regression function ρ described in Section 4.1. For training, the selected input attribute is the feature v in Table 1, and the outcome attribute is the feature *class*. If $class \quad 1$ it means that the customer accepts the new price, if $class \quad 0$ s/he doesn't. After training, the function ρ is able to predict for a given increment the grade of acceptance. The function ρ will be used to model the Pricing Strategy Optimization problem as an MDP as described in Section 5. Secondly, we assume that the customers in this synthetic dataset are the customers in our portfolio, and their features (*price* and *tier*) are part of the description of the state space. Then, each of these customers is considered in a different state as described in Section 5. Therefore, such customers are used to create the synthetic MDP described in Section 5, in which the agent travels for learning.

5. Mapping Pricing Strategy Optimization onto Reinforcement Learning

In this section, we model the renewal price adjustment problem as a non-deterministic MDP with continuous state and action spaces. In particular, we describe the mapping of the problem of renewal portfolios described in Section 4 onto a RL task. We consider this task as an episodic task, where each episode has as many steps as customers N are in the portfolio. Therefore, each one of these episodes can be considered as a real-life renewal period, in which the company renews one by one all the customers in its portfolio. For each step n, the learning agent receives a state s_n. In this paper, a state s_n is a tuple in the form $s_n \quad \langle f_1^n, f_2^n, tier_n, price_n \rangle$, where f_1^n and f_2^n are respectively the revenue and the retention at step n, and $tier_n$ and $price_n$ are respectively the customer segment and the insurance price for customer n in the portfolio. Therefore, each state s_n is composed of global features relating to the status of the insurance company (i.e., f_1 and f_2) and customer features (i.e., *tier* and *price*). From a RL point of view, it is also important to note that the customer segmentation can be considered as a way of compacting several customer features into only one feature already provided by the insurance company, *tier*, which allows reducing the state space. Therefore, the features *tier* and *price* perfectly describes the particular situation of a given customer. In respect to the global features, we have included in the state description those directly related to the considered objectives.

For the first state of each episode, s_1, the initial revenue and retention are computed as in Equations (11) and (12), i.e., we start from a situation where we have the profit generated by all the customers belonging to our portfolio (Equation (11)), but we also assume that we are going to lose some of these customers even offering them the same price, which is simulated by a renewal ratio of 1 (Equation (12)). This starting point is, in fact, the one that best fits the starting point in the real world.

$$f_1^1 \quad \sum_{i=1}^{N} price_i \tag{11}$$

$$f_2^1 \quad \frac{1}{N} \sum_{i=1}^{N} \rho_i(1) \tag{12}$$

At state s_n the agent performs an action a_n. In this paper, we consider action a_n is the renewal ratio v_i for the current customer n. The renewal ratios for each client are limited with the constraints applied to each customer, $[C_{n0}, C_{n1}]$, as explained in Section 4. This is to say that each customer has their own constraints for renewal ratios. Therefore, a_n is sampled from $[C_{n0}, C_{n1}]$.

After performing an action a_n in state s_n, the agent transits to a new state $s_{n+1} \quad \langle f_1^{n+1}$, $f_2^{n+1}, tier_{n+1}, price_{n+1} \rangle$. Therefore, a transition function is required in order to compute the values for f_1^{n+1} and f_2^{n+1}. These values are respectively computed using Equations (13) and (14) where $n \quad 1, \ldots, N$.

$$f_1^{n+1} \quad \sum_{i=1}^{n} \mathcal{B}(\rho_i(a_i)) * price_i * a_i + \sum_{i=n+1}^{N} price_i \tag{13}$$

$$f_2^{n+1} \quad \frac{1}{N} \left(\sum_{i=1}^{n} \rho_i(a_i) + \sum_{i=n+1}^{N} \rho_i(1) \right) \tag{14}$$

Both Equations (13) and (14) are divided into two main parts: the first takes into account the customers for whom a renewal ratio a_n has already been selected in the episode, and the second the customers in the portfolio for whom this renewal ratio has not yet been selected. For the latter, we assume a renewal ratio of 1. In this way, Equation (13) and (14) remain consistent with Equations (11) and (12). It is important to be aware of the fact that the transition function only affects the f_1 and f_2 part of the state space, i.e., to the global situation of the company, since no matter what action the learning agent takes in step n, $tier_{n+1}$ and $price_{n+1}$ will be the customer segment and the insurance price of the next customer in the portfolio.

Finally, when the learning agent performs an action a_n in a state s_n and moves to a state s_{n+1}, it also receives a reward signal r_n. Therefore, the definition of a reward function is also required. In this paper, we formulate a reward function for each of the two proposed strategies to optimize the renewal price: the maximization of the revenue, and the maximization of the revenue subject to the retention does not fall below a given threshold. The former strategy is related to an MDP with a single criterion to be optimized. In this case, the reward function is formulated as in Equation (15), i.e. at step n the reward is the difference between the value of revenue function in state s_{n+1} and state s_n.

$$r_n \quad (f_1^{n+1} - f_1^n) \tag{15}$$

Regarding the latter strategy, it is related to a CMDP with two criteria: the first is to be optimized, and the second is subject to a constraint (Equation (5)). With this type of problem it is common to formulate two reward functions: one for each objective. It is also common practice to include the cumulative reward of the constrained objective in the state description. Therefore, an additional negative reward $\Gamma < 0$ is given, when the process enters a state such that the accumulated reward of the constrained objective is below θ, i.e. when the constraint of the CMDP is violated (Geibel, 2006; Achiam et al., 2017). However, in our case there will be no need for an additional reward function for the constrained objective. The reason for this is that the retention function f_2 already belongs to the state description. Therefore, the MDP becomes a CMDP using the reward function in Equation (16) instead of that in Equation (15).

$$r_n \quad \begin{cases} (f_1^{n+1} - f_1^n), & \text{if } f_2^{n+1} \geqslant \theta. \\ \Gamma, & \text{otherwise.} \end{cases} \tag{16}$$

By the reward function in Equation (16), the agent receives a reward of $(f_1^{n+1} - f_1^n)$ if $f_2^{n+1} \geqslant \theta$, and a reward of $\Gamma < 0$ otherwise. In this way, the agent learns to avoid states in which $f_2 < \theta$, i.e., states in which the constrained objective is violated.

It is important to be aware of the fact that an interesting property of this model is that the competence is indirectly modeled by means of the probability of customer acceptance ρ used in Equations 12 and 14. The probability ρ is related to the prices offered by other companies. As an example, it is logical to assume that a given customer will have very low probability of accepting a renewal price of 50€, if other companies offer him/her a price of 10€. By contrast, this customer will have a high probability of accepting a renewal price of 10€, if other companies offer him/her a price of 50€. Therefore, the learning agent will adjust the renewal prices indirectly taking into consideration the competence by ρ in order to retain the highest number of customers.

6. The VQQL Algorithm

As explained in Section 5, the state and the action spaces of the considered problem are continuous. Therefore, in this paper, we propose to effectively discretize both spaces to obtain a finite set of states and actions. First, we uniformly discretize the action space. To this end, we divide uniformly the range of possible renewal ratios of each customer such that each one has a finite number of actions. Then, we use the VQQL algorithm to deal with the continuous state space and to learn a tabular representation of the Q-function (Fernández & Borrajo, 2000).

VQQL is based on the unsupervised discretization of the state space. Although any discretization of the state space breaks the Markov property, this has as its main consequence the introduction of non-determinism in the problem (Fernández & Borrajo, 2008). Therefore, in problems with continuous variables, it is crucial to choose a good discretization of the state space. If we discretize too finely, we lose the ability to generalize and increase both the size of the Q-table and the amount of training data required to learn the optimal policy. This is especially true when the task state is multi-dimensional, where the number of discrete states can be exponential in the state dimension. On the contrary, if we discretize the state space too coarsely, two states may be considered similar states when they are really completely different and, hence, they require completely different actions. For this reason, we use vector quantization methods, and more specifically, the generalized Lloyd algorithm (GLA), also called k-means. It has been previously successfully applied to discretize the state space in a multitude of problems (Fernández & Borrajo, 2000; Fernández et al., 2005; García et al., 2010). This method generates a set of centroids that together with the nearest neighbor rule define Voronoi regions. Each of these regions clusters a set of states of the original state space representation. The use of the k-means algorithm to solve the generalization problem in RL algorithms requires two consecutive phases:

- *Learning of the discretization of the state space.* The pseudocode of this phase is depicted in Algorithm 1. In this phase, we have to collect a sufficiently large number of experience tuples of the form $\langle s_n, a_n, s_{n+1}, r_n \rangle$ from the environment during exploration. To do this, the algorithm receives as input the number of episodes to interact with the environment, H, the number of steps of each of these episodes, N, and the number of centroids to generate, k. For each step n in episode h, the algorithm takes an exploratory action a_n in s_n and perceives the new state s_{n+1} and the reward r_n (line 5 of Algorithm 1). Then, it composes the tuple $\langle s_n, a_n, s_{n+1}, r_n \rangle$ which is added to the set C (line 6). Next, we use the states C_s in these tuples as input of the k-means algorithm to obtain a discretization, D, containing k centroids (lines 10-11). In this way, given a state s, $D(s)$ assigns s to the closest centroid in D.

- *Learning of the Q-function.* Once the discretization is designed (i.e., the environment is clustered into k different states), the Q-function must be learned, generating the Q table composed of k rows and a column for each action. The pseudocode of this phase is depicted in Algorithm 2. It receives as input a maximum number of learning episodes H, a number of steps per episode, N, the discretization learned in the previous phase, D, and the parameters α and γ for the Q-learning update equation. Then, it performs the following steps to learn the Q-function. First, it chooses an action following an exploration/exploitation strategy (e.g., ϵ-greedy) and receives the reward, r_n, obtaining an experience tuple $\langle s_n, a_n, s_{n+1}, r_n \rangle$ (lines 5-6 of Algorithm 2). Then, it uses the discretization D learned in the previous step to discretize the experience tuple, $\langle D(s_n), a_n, D(s_{n+1}), r_n \rangle$ (line 7). This discretized experience tuple is used to update the Q-function (line 8).

Algorithm 1: Learning of the discretization of the state space.

```
Data: $H, N, k$
Result: The discretization $D$
begin
    Set $h \quad 0, n \quad 0, C \quad \emptyset$;
    for $h < H$ do
        for $n < N$ do
            Take action $a_n$ in state $s_n$ (e.g., randomly), observe $r_n$ and $s_{n+1}$;
            $C \quad C \cup \langle s_n, a_n, s_{n+1}, r_n \rangle$;
            $n \quad n + 1$;
        $n \quad 0$;
        $h \quad h + 1$;
    Let $C_s$ be the set of states of $C$;
    Learn a discretization $D$ with $k$ centroids from $C_s$ using k-means;
    return $D$;
```

Algorithm 2: Learning of the Q-function.

```
Data: $H, N, D, \alpha, \gamma$
Result: The policy $\pi$ derived from $Q(s, a)$
begin
    Set $h \quad 0, n \quad 0, Q(s, a) \quad 0, \forall s \in S, \forall a \in A$;
    for $h < H$ do
        for $n < N$ do
            Take action $a_n$ in state $s_n$ (using e.g., $\epsilon - greedy$), observe $r_n$ and $s_{n+1}$;
            Obtain $\langle s_n, a_n, s_{n+1}, r_n \rangle$;
            Use $D$ to discretize $\langle s_n, a_n, s_{n+1}, r_n \rangle$ obtaining $\langle D(s_n), a_n, D(s_{n+1}), r_n \rangle$;
            $Q(D(s_n), a) \rightarrow Q(D(s_n), a_n) + \alpha(r_n + \gamma \max_{a'} Q(D(s_{n+1}), a') - Q(D_{s_n}, a_n))$;
            $n \quad n + 1$;
        $n \quad 0$;
        $h \quad h + 1$;
    return $\pi$ derived from $Q(s, a)$;
```

It is important to be aware of the fact that previous to the computation of the Euclidean distance between two states in the k-means algorithm, the features f_1, f_2 and price have been normalized in the range $[0, 1]$. Regarding the feature *tier*, it is split into three different features: *tier*1, *tier*2 and *tier*3. Each of these features takes the value of 0 if the corresponding customer is not within that risk segment and 1 otherwise. Note that if all these features are 0 it is assumed that the customer is within the risk segment *tier*4. Therefore, these three features allow us to identify to which of the four risk segments a customer belongs to. In this way, all features have the same scale for a fair comparison between them.

Both phases - the learning of the discretization and the learning of the Q-function - are applied within the VQQL model to the considered problem in Section 7.

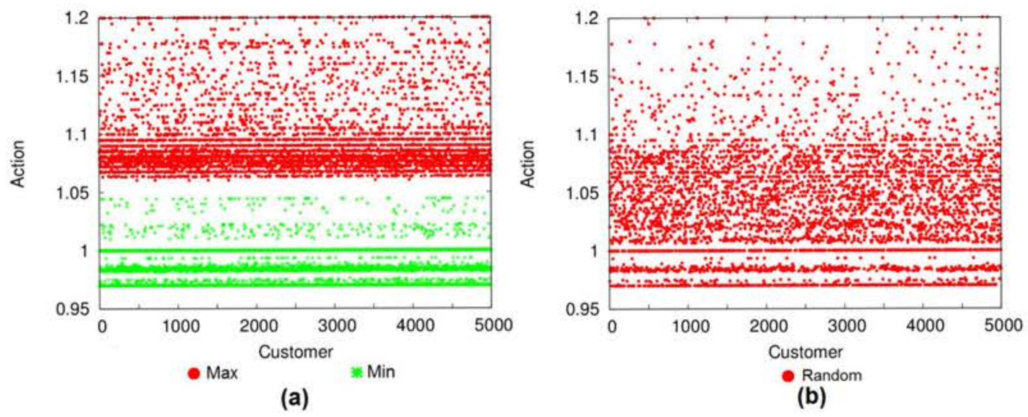

Figure 2: Performed actions with (a) the max and min baseline behaviors and (b) the random baseline behavior

7. Experimental Results

This section presents the experimental results collected from the use of the VQQL algorithm in the problem of renewal price adjustment. Prior to the application of this algorithm, the range of possible renewal ratios of each customer is uniformly divided that each individual has 10 possible actions (increments/decrements) including the value of 1 (when the price is the same for the next renewal). In this way, we have a finite and discrete space of actions.

Subsequently, the VQQL algorithm is applied. As explained in Section 6, VQQL has two consecutive phases: learning of the discretization and learning of the Q-function. This section is structured following this same division. First, we provide a detailed description about the learning of discretizations of different sizes (Section 7.1). Finally, we use these discretizations to learn a Q-function (Section 7.2). As regards the latter, we provide results using the two proposed strategies: maximization of the revenue, and maximization of the revenue subject to the retention is above a given threshold.

7.1. Learning of the Discretizations of the State Space

This section explains the process for discretizing the state space. Prior to this discretization, it is necessary to collect from the MDP described in Section 5 a significant number of experience tuples of the form $\langle s_n, a_n, s_{n+1}, r_n \rangle$. To this end, the state and action spaces need to be sufficiently explored. In this paper we propose the use of three different action selection strategies to carry out this exploration. Each one of these strategies corresponds to a different baseline behavior: the first is selecting the minimal renewal ratio for each customer in the portfolio. The second is selecting the maximal renewal ratio for each one, while the third consists of selecting a random renewal ratio. Using these exploration strategies, we have collected half a million tuples. Figure 2 shows the performed actions for each of these baseline behaviors.

In particular, Figure 2 (a) shows the performed actions for the max and the min baseline behaviors, and Figure 2 (b) shows the performed actions for the random baseline behavior. From Figure 2 (a) we can observe that the min baseline behavior selects for each client n action a_n C_{n0} in the range $[C_{n0}, C_{n1}]$, whereas the max baseline behavior selects action a_n C_{n1} in the same range. In contrast, from Figure 2 (b) we can observe that the random baseline behavior selected for each client n an action a_n randomly selected from the range $[C_{n0}, C_{n1}]$.

The performance of these baseline behaviors for each learning objective is shown in Figure 3 (a). Figure 3 (a) shows the evolution of f_1 and f_2 for each baseline behavior throughout an episode. The black point in Figure 3 (a) denotes the start situation at the beginning of an episode (computed as in Equations (11) and (12)). Therefore, all the baseline behaviors start from the same starting situation and, as the episode proceeds, the values of f_1 and f_2 are computed using Equations (13) and (14). In Figure 3, the max baseline behavior produces a decay in the retention, producing also a reduction in the revenue because of a loss of customers. The min baseline behavior allows for an increase in retention; in this case, revenue also decreases due to the reduced renewal ratio selected. The random baseline behavior can be considered a ”balance” behavior between the previous ones: it produces soft reductions in both the retention and the revenue. Using this figure, we can have an estimation of the minimum and maximum values for both retention and revenue. It does not seem possible to have a retention higher than 0.92, nor lower than 0.75. Revenue seems to range from 0.75 to 0.92.

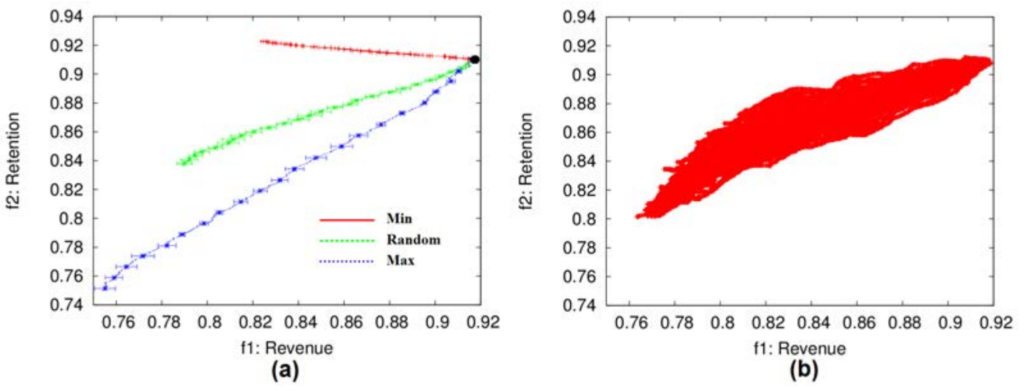

Figure 3: (a) Performance of the different baseline behaviors, and (b) the states visited during an exploration which uses alternatively these baseline behaviors. Figure (a) shows the average results for each learning processes together with their standard deviations computed from 10 different runs.

From Figure 3 (a) it is, however, easy to see that a random exploration is insufficient to efficiently explore the state space in this problem. Therefore, in this paper, to efficiently explore this space, we combine these three baseline behaviors during 100 episodes changing of baseline behavior every 100 steps. Figure 3 (b) shows the states visited during this exploration. As we can see, the use of this strategy allows us to explore a larger region of the state space. As a result of this exploration, it is produced a dataset of tuples in the form $\langle s_n, a_n, s_{n+1}, r_n \rangle$. The states s_n of this dataset are used as input of the k-means algorithm.

Finally, Figure 4 shows the centroids computed by the k-means algorithm for different (increasing) discretization sizes. These sizes range from k 128 to k 16384 clusters. Figure 4 shows, as the number of centroids increases, the corresponding point cloud is denser and appears more like the original in Figure 3 (b). In this way, the point cloud of the discretization of size 16384 is practically identical that in Figure 3 (b). All of these discretizations are used in the learning of the Q-functions as described in the next section. Therefore, Figure 4 shows the statistical advantages that VQ provides over other discretization approaches. In Figure 3 (b) only some regions of the bidimensional space have values, showing that not all combinations of the possible values of the Revenue and Retention exist in the state space. As it can be seen in Figure 4, VQ only generates centroids that represent minimally the states in Figure 3 (b), i.e., it only locates centroids in relevant areas of the state space. In contrast, other discretization approaches such as uniform clustering or tile coding (Sutton & Barto, 1998) do not take advantage of these statistical relationships between the state features, and discretize the whole state space without taking into account that some regions will never be visited. Additionally, while VQ only requires to provide as input the number k of centroids, in tile coding, for example, it is necessary to provide the number of partitions of each feature, the number of tilings, the offset between these tilings, Therefore, it is easier for VQ to define its learning parameters than it is for other discretization approaches, while maintaining of even increasing the performance results (Fernández & Borrajo, 2002).

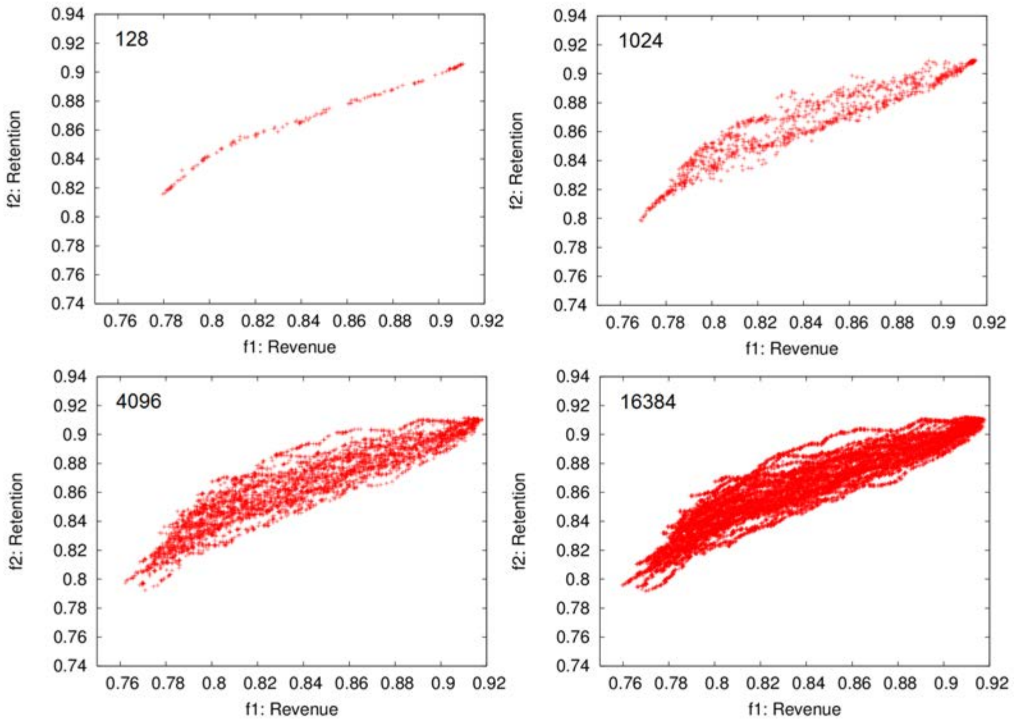

Figure 4: Centroids resulted from the k-means algorithm for $k = 128$, $k = 1024$, $k = 4096$ and $k = 16384$

7.2. *Learning of the Q-Function*

Once we have discretizations of different sizes for the state space, we can proceed with the second phase of the VQQL algorithm. In this section we learn different policies using the optimization strategies stated in the introduction: the maximization of the revenue, and the maximization of the revenue subject to the retention does not fall below a given threshold θ. As

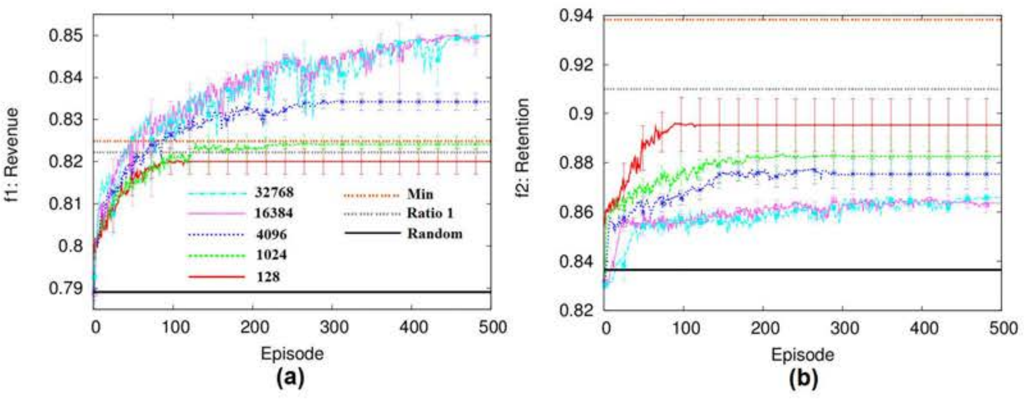

Figure 5: Evolution of (a) the revenue, f_1, and (b) the retention, f_2, for five learning processes (each one using a different discretization size) in the problem of maximization the revenue. The figure shows the average results for each learning processes together with their standard deviations computed from 10 different runs.

explained in Section 5, the first is related to an MDP, and the second to a CMDP. As also explained in Section 5 the difference between the two lies in the use of a different reward function: the reward function in Equation (15) for the first, and the reward function in Equation (16) for the second. In the following experiments, we use a fully greedy exploration strategy, i.e., the learner takes its current best action with probability 1. The parameter setting is as following: α 0.4 and γ 0.6. Next, we describe the results for each of the optimization strategies.

7.2.1. *Maximization of the Revenue*

First, we will focus on the first optimization strategy: the maximization of the revenue. Figure 5 shows the results for this strategy, i.e., the results when the reward function in Equation (15) is used. In particular, Figure 5 shows the evolution of the revenue, f_1, (Figure 5 (a)) and the retention, f_2, (Figure 5 (b)) during 500 episodes for five learning processes. For each one of these learning processes we use a different discretization size. These sizes range from 128 clusters (solid red line) to 32768 clusters (dashed light blue line). For each episode, Figure 5 shows the revenue and the retention at the last step N of the episode (i.e., once all the customers in the portfolio have been considered) by using the Equations (13) and (14). To test the performance of our pricing policies, we need to compare it with some baseline pricing policies. For this reason, Figure 5 also shows three horizontal lines, each one corresponding to the performance of a different baseline behavior: the first is concerned with selecting the minimum renewal ratio for each customer in the portfolio (dashed orange line), the second selects a renewal ratio for each customer of 1 (dashed grey line), while the third line selects a random renewal ratio (solid black line).

According to the results shown in Figure 5 we should highlight three main aspects. First, all learning processes improve the performance of f_1 and f_2 of the random baseline behavior (Figure 5 (a) and (b)). However, concerning f_1, only the learning processes associated respectively with the clusters of sizes 4096, 16384, and 32768 improve the performance of the min baseline behavior and the baseline behavior of ratio 1 (Figure 5 (a)). Regarding f_2, all learning processes are below the performance of these two baseline behaviors (Figure 5 (b)). Additionally, it is important to note that the min baseline behavior is an extreme behavior, i.e., it is not possible to retain more customers since we are offering them the lowest possible renewal prices. Therefore, in our portfolio, it is not possible to have an f_2 greater than 0.94. The second aspect to be highlighted from Figure 5 is that as the number of clusters increases, the performance of f_1 increases while the performance of f_2 decreases. Therefore, the higher the number of clusters is, the higher the renewal ratios will be, which has also the effect of many customers decide to leave. Finally, Figure 5 (a) also shows that the performance of f_1 does not improve for a number of clusters larger than 16384. With a number of clusters of 16384 a turning point has been reached, in which if the renewal ratios are increased, too many customers would decide to leave, and the revenue instead of increasing would decrease. This is the reason why we have tried with such so big values of k: we wanted to find the value of k from which no improvements could be made in f_1. The answer to this question is k 16386. Therefore, given our portfolio, it is not possible to gain a value of f_1 higher than 0.85.

Finally, Figure 6 shows two additional experiments designed to demonstrate that it is better to consider the proposed problem as an MDP. Figure 6 (a) shows the evolution of the revenue, f_1, during 500 episodes for two learning processes. The two learning processes use the same discretization size, 16384, but they use a different value for the discount factor γ: the first uses γ 0.6 (pink dashed line) and the other uses γ 0 (blue dashed line). The discount factor γ in RL determines the importance of future rewards. A discount factor of 0 means that the agent only cares about immediate rewards, while a discount factor of 0.6 represents a correct trade-off between present and future rewards. Figure 6 (a) shows that the learning process with γ 0.6 obtains a better performance than the learning process with γ 0. This means that the learning process with γ 0.6 is able to sacrifice immediate rewards in order to gain more long-term revenue, while the learning process with γ 0 only focuses on obtaining the greatest immediate rewards. This demonstrates that the problem is not about considering the customers individually and obtaining the greatest revenue for each decision, but it is about considering the problem as a sequential decision problem where the objective is to obtain the greatest total revenue. Finally, it is important to highlight that using values $\gamma > 0.6$, produce a similar performance than when using γ 0.6.

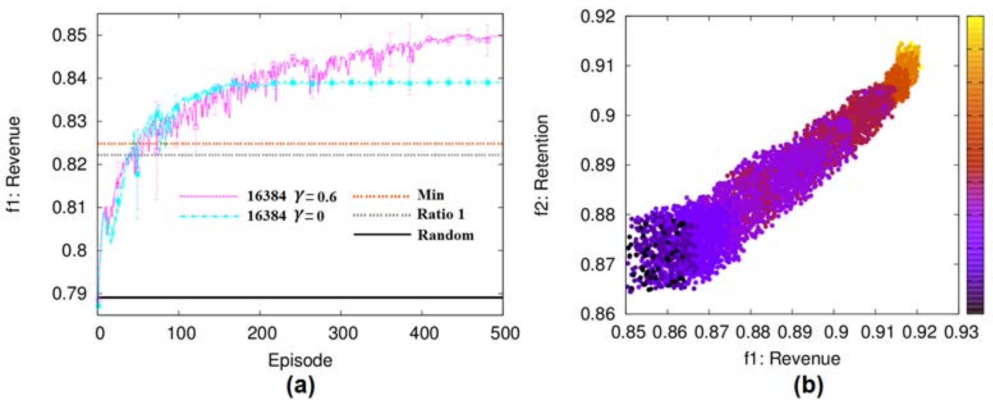

Figure 6: (a) Evolution of f_1 for two learning processes each one using a different γ value and a discretization size of 16384 and (b) renewal prices coloured from yellow (highest renewal prices) to dark blue (lowest renewal prices) depending on f_1 and f_2.

In addition, Figure 6 (b) demonstrates that the renewal price offered to each customer depends on the current situation of the company. In particular, it shows the states visited by the learned policy with γ 0.6. For each state, it shows the value of f_1 in the x-axis, and f_2 in the y-axis. Additionally, each state is colored in a different way depending on the action taken in that state: from yellow (corresponding to the highest renewal price), to dark blue (corresponding to the lowest renewal price). Figure 6 (b) shows that the highest renewal prices are offered when the situation of the company is better (the highest prices are offered when f_1 and f_2 are above 0.91). However, the lowest renewal prices are offered when the situation of the company is worse (the lowest prices are offered when f_1 is below 0.87 and f_2 is below 0.88). This means that when the company has a high revenue and retention it offers to the customers high renewal price. However, if the situation is worse, the company cannot afford to lose more customers, so it offers low renewal prices.

7.2.2. Maximization of the Revenue subject to Constraints

We will now focus on the second optimization strategy: the maximization of the revenue subject to the retention does not fall below a given threshold θ. Therefore, in the following experiments we will use the reward function in Equation (16) where the parameter Γ is set to Γ -1. Figure 7 shows the results using this reward function. For these experiments, we will use the discretization of size 16384, as this obtained the best possible value of f_1 in Figure 5.

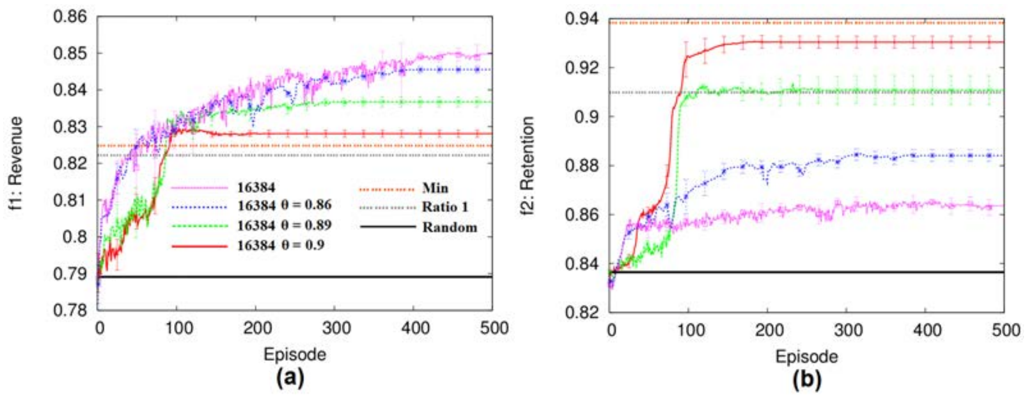

Figure 7: Evolution of (a) the revenue, f_1, and (b) the retention, f_2, for three learning processes (each one using a different threshold value) in the problem of maximization the revenue subject to the retention does not fall below a given threshold θ. The figure shows the average results of each learning processes together with their standard deviations computed from 10 different runs.

Figure 7 shows the evolution of f_1 and f_2 during 500 episodes for three different learning processes: the maximization of the revenue subject to the retention does not fall below θ 0.86 (dashed blue line), the maximization of the revenue subject to the retention does not fall below θ 0.89 (dashed green line), and the maximization of the revenue subject to the retention does not fall below θ 0.9 (solid red line). Figure 7 also shows an additional learning process: the learning process without constraints which use a discretization of size 16384 from Figure 5 (dashed pink line). It is important to note that this one is only included as a reference in the analysis. Finally, Figure 7 shows the same three baseline behaviors as in Figure 5.

In all cases, the imposed constraints are correctly fulfilled (dashed blue line, dashed green line and solid red line of Figure 7 (b)). However, the compliance of the constraints penalizes the performance of f_1: neither is able to reach the final performance of the learning process without constraints (dashed pink line in Figure 7 (a)). However, this loss of revenue is rewarded with a significant improvement in the performance of f_2 with respect to the learning process without constraints (Figure 7 (b)). Regarding the baseline behaviors, the three constrained learning processes improve the performance of f_1 for all the them (Figure 7 (a)), and they also improve the performance of f_2 for the random baseline behavior (Figure 7 (b)). Additionally, the learning processes with θ 0.89 and θ 0.9 improve the baseline behavior of ratio 1 (Figure 7 (b)).

Figure 8 (a) shows the evolution of the mean renewal ratio values for each episode of the learning processes in Figure 7. It can be seen that the unconstrained policy and the constrained one with θ 0.86 obtain a mean renewal ratio greater than 1 (pink and blue dashed lines in Figure 8 (a)). Therefore, the use of these two policies leads to an increase on average of the insurance prices. On the contrary, the constrained policies with θ 0.89 and θ 0.9 are below the mean renewal ratio of 1 (green dashed and solid red lines in Figure 8 (a)) and, hence, the use of these two policies leads to a decrease on the average of the insurance prices.

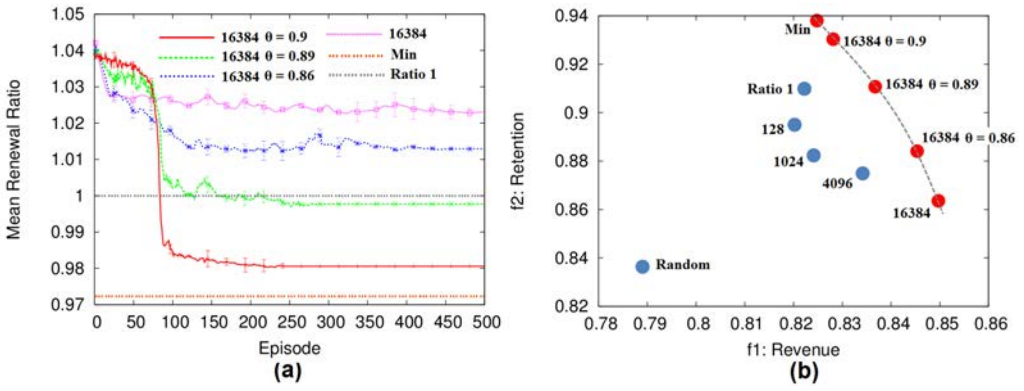

Figure 8: (a) Evolution of the mean renewal ratio values for the learning processes in Figure 7, and (b) performance of f_1 and f_2 for the policies in Figures 5 and 7. The figure shows the average results of each learning processes computed from 10 different runs. Additionally, Figure (a) also shows the standard deviations.

Finally, Figure 8 (b) shows the final performance for each objective, f_1 and f_2, of the policies shown in Figures 5 and 7. The Pareto comparison criterion can be used to compare the solutions in Figure 8 (b). Using this principle, one solution y^* strictly dominates (or ”is preferred to”) a solution y if the performance for each of the objectives of y^* is strictly better than the performance of the corresponding objective of y. In accordance with the Pareto principle, we can assume the blue points in Figure 8 (b) are strictly dominated by at least one red point. Additionally, we can assume the red points in Figure 8 (b) are in the Pareto front, since these points are not strictly dominated by any other solution (i.e., no other solution has, at the same time, a higher revenue and retention). In this problem, the Pareto front is composed of the solutions of the min baseline behavior, the constrained policies with θ 0.9, θ 0.89, and θ 0.86, and the unconstrained policy with a number of clusters of 16384.

At this point we will analyze the performance of the Pareto front solutions by taking as reference the performance of the baseline behavior of ratio 1. Additionally, this analysis will be carried out using the denormalized values for both f_1 and f_2, thus providing the reader with a deeper grasp and understanding of the resulting improvements. Additionally, we relied on the Mann-Whitney test (Corder & Foreman, 2014) to indicate a significant difference (using a significance level of 5%) between the performance of the baseline behavior of ratio 1 and the policies in the Pareto front. Figure 8 (b) shows that all the solutions in the Pareto front outperform the performance of f_1 794535€ of the baseline behavior of ratio 1 (i.e., regarding f_1, the performances of the Pareto front solutions are statistically significant better with a confidence level of 95% compared to the performance of the baseline behavior of ratio 1). The min baseline behavior has revenues of f_1 797065€ (i.e., an improvement of 0.3% compared to the baseline behavior of ratio 1). The constrained policies with θ 0.9, θ 0.89 and θ 0.86 have revenues of f_1 800235€, f_1 808573€ and f_1 817077€ respectively (0.7%, 1.8% and 2.8% improvement rate in f_1 with respect to the baseline behavior of ratio 1). Finally, the unconstrained policy has revenues of f_1 821151€ (i.e., representing a 3.3% increase in f_1 over the baseline behavior of ratio 1). Regarding the performance of the retention, the baseline behavior of ratio 1 obtains a final retention of f_2 91%. In this case, only the min baseline behavior (f_2 94%) and the constrained policy with θ 0.9 (f_2 93%) outperforms this retention value (i.e., regarding f_2, the performance of the baseline behavior of ratio 1 is statistically significant better with a confidence level of 95% compared to the performance of the min baseline behavior and the constrained policy with θ 0.9). Instead, there is no any significant statistical difference between the performance of f_2 of the baseline behavior of ratio 1 and the constrained policy of θ 0.89 (f_2 91.1%). Finally, the constrained policy with θ 0.86 (f_2 88%) and the unconstrained policy (f_2 86%) obtain slightly lower retention values.

8. Conclusions

The present paper describes the use of RL for the problem of renewal price optimization. At this point we can summarize the following principal conclusions:

(i) The problem can be successfully modeled as an MDP and a CMDP. In this paper, we have correctly modelled the proposed problem as an MDP and as a CMDP as detailed in Section 5. In fact, this is the main contribution of the paper: the novel modelization of the renewal price adjustment problem as an MDP and a CMDP. In this modelization we take both the global situation of the insurance company and the particular situation of each customer into consideration. Additionally, the global situation of the company is updated according to the actions performed. This allows us to consider the impact of current decisions on future decisions and opportunities. As mentioned earlier, as far as we now, this is the first time that this problem is modeled in this way. To accomplish this modelization, Section 5 describes all the elements of an MDP: the state and action spaces, and the reward and the transition functions. Regarding the CMDP, we take advantage of the fact that f_2 is part of the state description so that we do not have to define an additional reward function. For this reason, changing the problem from MDP to CMDP only requires changing the reward function used, i.e., use the reward function in Equation (16) instead of the reward function in Equation (15).

(ii) VQQL can be used to solve both the MDP and the CMDP. Formalizing the problem as an MDP allows the use of any model-free RL algorithm to solve it. However, in this paper we have opted for the well-known VQQL algorithm (Fernández & Borrajo, 2000). VQQL requires two consecutive phases: the learning of the discretization of the state space, and the learning of the Q-function. Regarding the former phase, we have to collect a sufficiently large number of experience tuples. Although in many problems it is sufficient to use a random exploration

to collect such tuples, Section 7.1 demonstrates that in this problem a random exploration only partially explores the state space. It is for this reason that we use alternatively three baseline behaviors to explore the state space: the max, min and random baseline behaviors. Section 7.1 shows that in this way the state space is explored more efficiently. Regarding the second phase of VQQL, the experiments in Section 7.2.1 demonstrate that a correct selection of the number of clusters is crucial to obtaining a good performance both for f_1 and f_2. Regarding the latter, a turning point is reached with a number of clusters of 16384, and revenue cannot be improved with current data from this point on. An increase in the renewal ratios would lead to a certain number of customers terminating their contracts, which, in turn, would affect revenue. Additionally, Section 7.2.1 also demonstrates that the renewal prices offered to the customers depend on the global situation in which the company is at that moment and not only on the particular situation of each customer. In our opinion, the latter reinforces the fact that it is better to consider the problem as an MDP as proposed in this paper. Finally, the experiments in Section 7.2.2 show that model-free RL algorithms such as VQQL can be successfully used to solve a CMDP.

(iii) Possibility of selecting the minimum acceptable retention value. The modelization of the problem as a CMDP allows the user to select the value of the threshold θ. Therefore, the problem now becomes one of maximising revenue, subject to the client retention level not falling below threshold θ (Section 7.2.2). This is a very important achievement because individual insurance companies can adjust the value of θ depending on their particular situation. In fact, modeling the problem as a CMDP better models what actually happens in the real world: insurance companies want to decide what level of customers they want to retain. However, it is necessary to keep in mind that the higher the value of θ is, the lower the margin for improvement of the revenue will be, as demonstrated in the experiments in Section 7.2.2.

(iv) Obtaining the Pareto front allows a posteriori *selection of the policy*. In Section 7.2.2 we have approximated correctly the Pareto front of the problem (Figure 8 (b)). This frontier allows *a posteriori* selection of the solution and encapsulates all the trade-offs among the multiple objectives. We cannot say which of the policies in the Pareto front is preferable. Nevertheless, the user is able to select a policy *a posteriori* depending on whether they prefer increased revenue over customer retention, or vice versa. If the most important objective is to achieve the maximum possible retention, then the policy corresponding to the min baseline behavior should be selected. Conversely, if the most important objective is to achieve the maximum possible revenue, then the policy of the unconstrained learning process with 16384 clusters should be selected. The rest of the policies in the Pareto front can be used to reach a trade-off between the two objectives.

Finally, a logical continuation of the present study would be the final deployment of the proposed ideas in the insurance company. The policies learned by RL can be used by the insurers as a decision support system that allows them to decide the renewal price for the customers. In this sense, one possibility would be to integrate such learned policies into a software tool that can be used easily by the insurers. Given a customer, the insurer provides to this tool the price the customer is currently paying, and the segment to which it belongs. We assume that both the revenue and the retention of the insurance company are already integrated in the tool. Note that this price, segment, revenue and retention represent the state in which the system is. Then, for this state the learned policy embedded in the tool gives as output the action, i.e., the renewal price that the insurer should offer to that customer. It is important to be aware of the fact that the output of the policy is simply a recommendation, i.e., the policy proposes a solution that the insurer may accept or change using any other approach. Human experts are suspicious of tools that provide solutions which cannot be changed, regardless of how sophisticated, intelligent or autonomic the tool is. Therefore, the final action to be performed is not decided by the policy but by the human expert. Anyway, the price recommended by the policy is a good starting point for the insurer to start the negotiation. Once the negotiation with the customer is finished, the insurer updates the policy by providing to the tool whether the customer has accepted or not, and the final renewal price offered. This information is used to perform a new learning step and the revenue and retention are recomputed as described in Section 5. In this way, the system keeps learning continuously and adaptively also in the real environment.

Acknowledgements

This work has been partially funded by the TIN2015-65686-C5 Spanish Ministerio de Economía y Competitividad project and FEDER funds. Javier García is partially supported by the Comunidad de Madrid (Spain) funds under the project 2016-T2/TIC-1712.

References

Abolmakarem, S., Abdi, F., & Damghani, K. K. (2016). Insurance customer segmentation using clustering approach. *International Journal of Knowledge Engineering and Data Mining*, *4*, 18–39.

Achiam, J., Held, D., Tamar, A., & Abbeel, P. (2017). Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (pp. 22–31).

Agresti, A. (2015). *Foundations of Linear and Generalized Linear Models*. Wiley Series in Probability and Statistics. Wiley.

Ahmadzadeh, S. R., Kormushev, P., & Caldwell, D. G. (2014). Multi-objective reinforcement learning for auv thruster failure recovery. In *2014 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)* (pp. 1–8).

Ansari, Y., Manti, M., Falotico, E., Cianchetti, M., & Laschi, C. (2018). Multiobjective optimization for stiffness and position control in a soft robot arm module. *IEEE Robotics and Automation Letters*, *3*, 108–115.

Barto, A. G., Bradtke, S. J., & Singh, S. P. (1995). Learning to act using real-time dynamic programming. *Artificial Intelligence*, *72*, 81–138.

Blum, A., & Hartline, J. D. (2005). Near-optimal online auctions. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms* SODA '05 (pp. 1156–1163). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Carpin, S., Pavone, M., & Sadler, B. M. (2014). Rapid multirobot deployment with time constraints. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, September 14-18, 2014* (pp. 1147–1154).

Cesa-Bianchi, N., Lugosi, G., & Stoltz, G. (2006). Regret minimization under partial monitoring. *Mathematics of Operations Research*, *31*, 562–580.

Chizhov, Y. A., & Borisov, A. N. (2011). Markov decision process in the problem of dynamic pricing policy. *Automatic Control and Computer Sciences*, *45*, 361–371.

Corder, G., & Foreman, D. (2014). *Nonparametric Statistics: A Step-by-Step Approach*. Wiley: Hoboken.

Du, M., Sassioui, R., Varisteas, G., State, R., Brorsson, M., & Cherkaoui, O. (2017). Improving real-time bidding using a constrained markov decision process. In G. Cong, W.-C. Peng, W. E. Zhang, C. Li, & A. Sun (Eds.), *Advanced Data Mining and Applications*. Springer International Publishing.

Fernández, F., & Borrajo, D. (2000). VQQL. Applying vector quantization to reinforcement learning. In *RoboCup-99: Robot Soccer World Cup III* (pp. 292–303). Springer Verlag volume 1856 of *Lecture Notes in Artificial Intelligence*.

Fernández, F., & Borrajo, D. (2002). On determinism handling while learning reduced state space representations. In *ECAI* (pp. 380–384). IOS Press.

Fernández, F., & Borrajo, D. (2008). Two steps reinforcement learning. *International Journal of Intelligent Systems*, *23*, 213–245.

Fernández, F., Borrajo, D., & Parker, L. (2005). A reinforcement learning algorithm in cooperative multi-robot domains. *Journal of Intelligent and Robotic Systems*, *43*, 161–174.

Ferreira, P. R., Paffenroth, R., Wyglinski, A. M., Hackett, T. M., Biln, S. G., Reinhart, R. C., & Mortensen, D. J. (2017). Multi-objective reinforcement learning-based deep neural networks for cognitive space communications. In *2017 Cognitive Communications for Aerospace Applications Workshop (CCAA)* (pp. 1–8).

García, J., López-Bueno, I., Fernández, F., & Borrajo, D. (2010). A comparative study of discretization approaches for state space generalization in the keepaway soccer task. In *Reinforcement Learning: Algorithms, Implementations and Aplications*. Nova Science Publishers.

Geibel, P. (2006). Reinforcement learning for MDPs with constraints. In *Proceedings of the 17th European Conference on Machine Learning, ECML 2005, Berlin, Germany, 18-22 September 2006* (pp. 646–653).

Germán, R. (2007). Lecture notes on generalized linear models.

Hinterhuber, A. (2017). Implementing pricing strategies. *Journal of Revenue and Pricing Management*, *17*, 51–80.

Kamishima, T., & Akaho, S. (2011). Personalized pricing recommender system: multi-stage epsilon-greedy approach. In *Proceedings of the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems* (pp. 57–64). ACM.

Khamis, M. A., & Gomaa, W. (2012). Enhanced multiagent multi-objective reinforcement learning for urban traffic light control. In *2012 11th International Conference on Machine Learning and Applications* (pp. 586–591). volume 1.

Kleinbaum, D. G., & Klein, M. (2010). *Logistic regression: a self-learning text*. (3rd ed.). New York: Springer.

Lee, I., & Lau, H. (2004). Adaptive state space partitioning for reinforcement learning. *Engineering Applications of Artificial Intelligence*, *17*, 577 – 588.

Lee, Y., & Nelder, J. (2003). Robust design via generalized linear models. *Journal Of Quality Technology*, *35*, 2–12.

Lowe, J., & Pryor, L. (1996). Neural networks v. glms in pricing general insurance. In *In General Insurance Convention*.

Murphy, K., Brockman, M., & Lee, P. (2000). Using generalized linear models to build dynamic pricing systems for personal lines insurance. In *Casualty Actuarial Society Forum* (pp. 107–139).

Nelder, J., & Wedderburn, R. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A (General)*, *135*, 370–384.

Parodi, P. (2012). Computational intelligence with applications to general insurance: a review: I the role of statistical learning. *Annals of Actuarial Science*, *6*, 307343.

Phillips, R. (2005). *Pricing and Revenue Optimization*. Stanford Business Books. Stanford University Press.

Raju, C. V. L., Narahari, Y., & Ravikumar, K. (2006). Learning dynamic prices in electronic retail markets with customer segmentation. *Annals of Operations Research*, *143*, 59–75.

Rana, R., & Oliveira, F. S. (2015). Dynamic pricing policies for interdependent perishable products or services using reinforcement learning. *Expert Systems with Applications*, *42*, 426–436.

Roijers, D. M., Vamplew, P., Whiteson, S., & Dazeley, R. (2013). A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, *48*, 67–113.

Shengchao, Y. (2010). Multi-objective reinforcement learning for traffic signal coordinate control. In *12th World Conference on Transport Research*.

Spedicato, G., Dutang, C., & Petrini, L. (2018). Machine learning methods to perform pricing optimization. a comparison with standard glms. *Variance Journal*, .

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement Learning : An Introduction*. MIT Press.

Tijsma, A. D., Drugan, M. M., & Wiering, M. A. (2016). Comparing exploration strategies for q-learning in random stochastic mazes. In *2016 IEEE Symposium Series on Computational Intelligence, SSCI 2016, Athens, Greece, December 6-9, 2016* (pp. 1–8).

Trovò, F., Paladino, S., Restelli, M., & Gatti, N. (2018). Improving multi-armed bandit algorithms in online pricing settings. *International Journal of Approximate Reasoning*, .

Uther, W., & Veloso, M. (1998). Tree based discretization for continuous state space reinforcement learning. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence* AAAI '98/IAAI '98 (pp. 769–774). Menlo Park, CA, USA: American Association for Artificial Intelligence.

Vamplew, P., Dazeley, R., Berry, A., Issabekov, R., & Dekker, E. (2011). Empirical evaluation methods for multiobjective reinforcement learning algorithms. *Machine Learning*, *84*, 51–80.

Watkins, C. (1989). *Learning from Delayed Rewards*. Ph.D. thesis King's College Cambridge, UK.

Wiering, M., & van Otterlo, M. (2014). *Reinforcement Learning: State-of-the-Art*. Springer Publishing Company, Incorporated.

Wuthrich, M. V., & Buser, C. (2016). *Data Analytics for Non-Life Insurance Pricing*. Swiss Finance Institute Research Paper Series 16-68 Swiss Finance Institute.

Yeo, A. C. (2009). Neural networks for automobile insurance pricing. In *Encyclopedia of Information Science and Technology, Second Edition*. IGI Global.