Research Article

Frédéric Logé*, Erwan Le Pennec, and Habiboulaye Amadou-Boubacar

Intelligent Questionnaires Using Approximate Dynamic Programming

https://doi.org/10.1515/icom-2020-0022

Abstract: Inefficient interaction such as long and/or repetitive questionnaires can be detrimental to user experience, which leads us to investigate the computation of an intelligent questionnaire for a prediction task. Given time and budget constraints (maximum q questions asked), this questionnaire will select adaptively the question sequence based on answers already given. Several use-cases with increased user and customer experience are given.

The problem is framed as a Markov Decision Process and solved numerically with approximate dynamic programming, exploiting the hierarchical and episodic structure of the problem. The approach, evaluated on toy models and classic supervised learning datasets, outperforms two baselines: a decision tree with budget constraint and a model with q best features systematically asked. The online problem, quite critical for deployment seems to pose no particular issue, under the right exploration strategy.

This setting is quite flexible and can incorporate easily initial available data and grouped questions.

Keywords: Planning, Questionnaire design, Approximate dynamic programming

CCS Concepts: Computing methodologies → Planning under uncertainty, Computing methodologies → Approximate dynamic programming methods

1 Introduction

In user interaction, less is often more. When asking questions, it is desirable to ask as few questions as possible or given a budget of questions asking the most interesting ones. We study the case of intelligent questionnaires in which the questions asked may depend on the previous answers. More precisely, we consider a set of p questions in a prediction context. Given a budget of q questions, we design an algorithm choosing sequentially the next question to be asked so that the predictive power is maximized after having q answers. We assume that we have observed the whole set of answers on a first dataset and that no prior knowledge is available.

This setting is quite general and comprises for example:

- Patient follow-up. Consider a patient who is hospitalized at home and fills in a daily checkup questionnaire asking for leg pain, a hurting chest or physical discomfort. The aim of the questionnaire being to check the status of the patient, we would like to ask the most pertinent questions as soon as possible and personalize the questions to their status.
- Prospective calls e. g. telemarketing. Instead of bluntly unrolling the same list of questions, we could adapt our series of questions in order to know as fast as possible if the person called would be interested or not in our product.
- Cold start issue with new customers. When subscribing to a new service, it is not uncommon to get asked some questions in order to personalize the service e. g. Netflix, web service provider. Assuming we already have a customer clustering at hand, we would like to find the new customer's cluster with as seamlessly as possible. One way to do so is to ask very few questions.
- Balancing acquisition cost with available information. Assuming the data is paid for, e. g. personal data sold, we would choose which information to pay for each people.
- Storing less data to make as good predictions. Considering that data storage has non-negligible cost, whether it be financial, facility-wise or environmental, we wish to only keep data which is essential to the prediction. One way to do so is to store a sparse matrix.

Adaptive questionnaires have been investigated through knowledge-based approaches in several fields amongst

***Corresponding author: Frédéric Logé,** Air Liquide R&D, Jouy-en-Josas, Paris, France; and CMAP, Polytechnique, Institut Polytechnique de Paris, Palaiseau, France, e-mail: frederic.logemunerel@gmail.com, ORCID: https://orcid.org/0000-0003-0941-7171
Erwan Le Pennec, CMAP, Polytechnique, Institut Polytechnique de Paris, Palaiseau, France; and XPop, Inria Saclay, Palaiseau, France, e-mail: erwan.le-pennec@polytechnique.edu
Habiboulaye Amadou-Boubacar, Air Liquide R&D, Jouy-en-Josas, Paris, France, e-mail: habiboulaye.amadou-boubacar@airliquide.com

which we find e-learning [14] and healthcare [9]. In [13] the authors investigated an approach relying on association rules for the prediction and question selection tasks and experimented their algorithm on Myers-Briggs tests.

Such a sequence of questions depending on the previous answers has a tree-like structure. A classical CART algorithm [6] with a tree of depth q provides a solution but is optimized in a top-down manner whereas we propose a bottom-up optimization. Furthermore, we allow much more flexibility than a single coordinate thresholding to choose the next question, or than association rules.

We formulate this problem as a sequential decision-making problem and represent it by a Markov Decision Process [16] where the state is the information currently available, actions are the questions we ask and as final rewards the prediction performance based on the final partial information. Take a look at Figure 1 to have an idea of the expected result. Such a modeling has been used for instance in [7] to tackle the game of 20 questions relying on a smart matrix factorization. This setting has also been used in active learning [15] and more recently in a health diagnosis problem using a reinforcement learning approach [5].

In our setting, we assume we have the full set of answers in a first dataset, hence we do not have an exploration issue and can thus focus on a planning approach [11]. We show how to use approximate dynamic programming [3, 4] to propose this adaptive sequence of questions so that it outperforms a fixed subset of q questions or a depth q CART decision tree. Several toy models and benchmark datasets will be used as means to validate our approach.

We start by presenting the methodology in section 2, followed by experimental results on toy models and benchmark datasets in section 3. In section 4 we investigate the possibility of bringing this methodology online, by testing it on one of the toy models. A general conclusion as well as ideas for improvements are presented in section 5.

2 Methodology

2.1 Setting

Consider $Y \in \mathcal{Y}$, $\dim(\mathcal{Y}) \geq 1$, our variable of interest and $X \in \mathcal{X}$, $\dim(\mathcal{X}) = p$, the variable vector which can be used to predict Y and can be collected via survey element-by-element. Since we collect X in order to predict Y we would like to build an intelligent questionnaire which would collect elements of X which are the most useful for the prediction task. As such, this questionnaire will take into account the realizations of the elements of X that were already requested and check which new feature could be the most useful for our task. This process is repeated q times, $q < p$, akin to a questionnaire with budget constraint. We write $\tilde{\mathcal{X}}$ the space of partially-known feature vector $\tilde{X}$, where to each dimension is added the element "unknown", encoding the fact that an element has not yet been queried.

We aim at designing an Intelligent Questionnaire algorithm π^*, which to any element of $\tilde{\mathcal{X}}$ assigns the best next question to ask, formally defined as follows:

$$\forall \tilde{x} \in \tilde{\mathcal{X}} \qquad \pi^*(\tilde{x}) = \arg\max_{\pi \in \Pi} \mathbb{E}_{\pi,(X,Y)}[score(\tilde{X}_q, Y) | \tilde{X} = \tilde{x}] \tag{1}$$

where the function *score* measures how accurately we can predict Y based on $\tilde{X}_q$, which is the partially-known feature vector obtained when the algorithm is stopped. Π stands for the set of functions mapping $\tilde{\mathcal{X}}$ to $\{1, \dots, p\}$ i. e. the set of algorithms recommending a question to ask based on partial information on X.

We propose the following score function:

$$\forall (\tilde{x}, y) \in \tilde{\mathcal{X}} \times \mathcal{Y} \qquad score(\tilde{x}, y) = -\mathbf{R}(\hat{m}(\tilde{x}), y) \tag{2}$$

where $\hat{m}$ is a prediction function of the target based on partial information and $\mathbf{R}$ is an individual risk measure, such as the squared error in regression or the log-loss in classification. The methodology that follows is reliant on function *score* and therefore the quality of predictor $\hat{m}$.

2.2 Markov Decision Process

The questionnaire process can be modeled by a Markov Decision Process (MDP, [16]) $\mathcal{M} = (\tilde{\mathcal{X}}, \mathcal{A}, T, R)$ where $\tilde{\mathcal{X}}$ is the state space, where the partially-known feature vector $\tilde{X}$ lives and $\mathcal{A} = \{1, \dots, p\}$ is the action space, the indexes of feature elements we can collect. T is the transition function such that for any initial feature vector $\tilde{x}$, any action j and any other feature vector $\tilde{x}'$, $T(\tilde{x}, j, \tilde{x}')$ denotes the probability of observing feature vector $\tilde{x}'$ when asking for element j and starting from feature vector $\tilde{x}$.

Finally, R is the reward function defined for any $\tilde{x}$ such that the algorithm is stopped i. e. any $\tilde{x}$ with q elements filled, such that $R(\tilde{x})$ follows the conditional distribution

$$\mathbb{P}_{(X,Y)}[score(\tilde{x}, Y) | \tilde{X} = \tilde{x}]. \tag{3}$$

For feature vectors with less than q elements, the reward function equals 0.

The MDP starts with initial state $\tilde{X}_0$ = "unknown"^{p}, we then ask question A_0, coordinate A_0 is revealed (state $\tilde{X}_1$), we then ask question A_1, reach state $\tilde{X}_2$, and so on and so forth, until a terminal state is reached. In our case, we will stop when q questions will have been asked.

In one of our experiments (Coronary Heart Disease dataset), we extend the setting to the case where initial information is available, allowing us to personalize the initial question. We also consider that to a given action, multiple features may be revealed. Other extensions are discussed in the last section.

Please note that it is straightforward to consider the case where initial information is known, and we still would like to reveal q new informations. Also, one might consider a specific action-question interaction, for instance: one might consider that some action reveal groups of features, because those are somewhat related and packed together during the interaction. To do so, one would only need to specify a binary matrix indicating the relationship between an action and the feature revealed: 1 if it is, 0 otherwise. We applied this formalism to one of the benchmark datasets, see Figure 2, for which we considered that gender information was available at the beginning and we considered that action 6 consisted in a short heart rate monitoring, hence providing us directly with both systolic and diastolic blood pressure levels.

2.3 Proposed Solution

We assume that we have at our disposal the set $\{(x^{(i)}, y^{(i)}), i \in \{1, \dots, n\}\}$, consisting of n independent and identically distributed instances from variables (X, Y) i. e. full questionnaires. From there we can fit prediction function $\hat{m}$, create the set of observed transitions and the set of rewards for terminal states, since we define function *score*. Based on those datasets we propose to learn π^* through the following state-action value functions:

$$\forall (\tilde{x}, a) \in \tilde{\mathcal{X}} \times \mathcal{A}$$
$$Q_\pi(\tilde{x}, a) = \mathbb{E}_{\pi,(X,Y)}[score(\tilde{X}_q, Y) | \tilde{X}_t = \tilde{x}, A_t = a]. \quad (4)$$

The problem being episodic (q steps) and hierarchical, we can use approximate dynamic programming [4] to learn the value functions in a backward fashion as presented in Figure 1.

Based on the calibrated neural networks $\{\hat{f}_j, j \in \{0, \dots, q-1\}\}$ we would apply the Intelligent Questionnaire as presented in algorithm 2.

The objective at the end is to get something resembling the decision-making procedure presented in Figure 1.

Algorithm 1: Learning value functions.

Input data $\{(x^{(i)}, y^{(i)}), i \in \{1, \dots, n\}\}$
Input function *score*
Learn f_{q-1} as $\hat{f}_{q-1}$, where

$$f_{q-1} : \tilde{\mathcal{X}} \times \mathcal{A} \to \mathbb{R}$$
$$(\tilde{x}, a) \mapsto \mathbb{E}_{(X,Y)}[score(\tilde{X}_q, Y) | \tilde{X}_{q-1} = \tilde{x}, A_{q-1} = a]$$

for $j \in \{q-1, q-2, \dots, 1\}$ **do**
  Learn f_{j-1} as $\hat{f}_{j-1}$, where

$$f_{j-1} : \tilde{\mathcal{X}} \times \mathcal{A} \to \mathbb{R}$$
$$(\tilde{x}, a) \mapsto \mathbb{E}_{(X,Y)}[\max_{a'} \hat{f}_j(\tilde{X}_j, a') | \tilde{X}_{j-1} = \tilde{x}, A_{j-1} = a]$$

end
Return set of networks : $\{\hat{f}_j, j \in \{0, \dots, q-1\}\}$

Algorithm 2: Intelligent Questionnaire algorithm.

Interacting user knows (x, y)
Initialize $\tilde{x} \leftarrow$ "unknown"^{p}
for $j \in \{1, \dots, q\}$ **do**
  `/* select next action/question to ask */`
  $a_j \leftarrow \arg\max_{a \in \mathcal{A}} \hat{f}_{j-1}(\tilde{x}, a)$
  `/* retrieve corresponding element */`
  $\tilde{x}_{a_j} \leftarrow x_{a_j}$
end
Return prediction of y : $\hat{m}(\tilde{x})$

2.4 Online Setting

Let us now assume that we are going to collect questionnaires in a smart way i. e. starting with no data and with the constraint on the number of questions authorized. Specifically, we assume that this process starts with a run-in period where T_0 observations of couples $(\tilde{X}_q, Y)$ are collected following an initial exploration strategy π_0 e. g. random subset of q questions. Then, we estimate prediction function $\hat{m}$ and Q-value estimates $\{\hat{f}_j, j \in \{0, \dots, q-1\}\}$ based on available data. From there, we will follow strategy π_1 to collect new observations, balancing between data diversity (exploration) and estimated optimal decisions based on running estimates (exploitation). In this process, illustrated in Figure 3, we write $\mathcal{D}$ the dataset augmented step by step.

Let us now propose candidate strategies for π_1. Consider $\tilde{x}$ our partial information available after l features col-

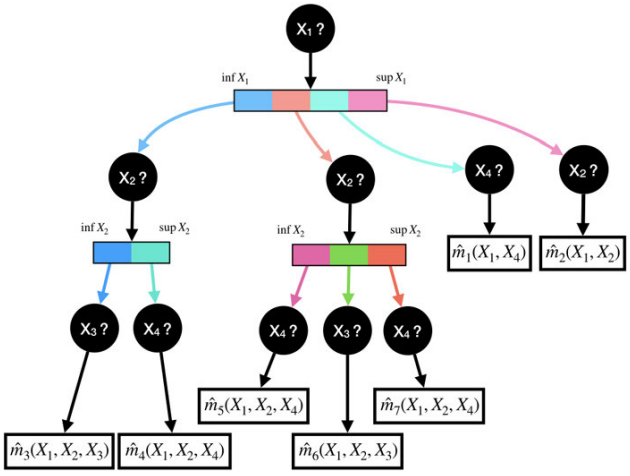

Figure 1: Intelligent Questionnaire expected. Actions chosen are represented by black circles, transitions are colored depending on revealed values and final predictions are indicated by functions $\hat{m}_\cdot$ in terminal leaves. In this example, the algorithm splitted the range of values for feature X_1 in four bins. On the second layer, the algorithm splitted differently the range of values of feature X_2 depending on the observed value for X_1 (two and three bins respectively). This process remembers all values provided before, partitions space $\mathcal{X}$ and instead of assigning a single value for prediction, as a decision tree would, it uses a prediction function based on retrieved information. The objective is therefore not necessarily to group observations homogeneous in Y, but rather the ones following similar underlying models of $Y|\tilde{X}$.

Algorithm 3: Online Intelligent Questionnaire algorithm.

Input (T_0, T) : run-in phase and total durations
Input π_0 : strategy followed during run-in phase
Input π_1 : strategy followed after run-in phase
for $t \in \{1, \dots, T\}$ **do**
  if $t > T_0$ **then**
    Estimate $(\hat{m}, \{\hat{f}_j\}_{j=0}^{q-1})$ based on $\mathcal{D}$
  end
  Follow strategy $\pi_{\mathbf{1}\{t>T_0\}}$ for individual t
  Store couple $(\tilde{x}_q, y)$ observed in $\mathcal{D}$
end

lected. We propose first the softmax policy:

$$a_{l+1} \sim \text{Multinomial}\left(\frac{\exp\{\beta\, \hat{f}_l(\tilde{x}, .)\}}{\sum_{k=1}^{p} \exp\{\beta\, \hat{f}_l(\tilde{x}, k)\}}\right) \qquad (5)$$

with control parameter $\beta, \beta \geq 0$. Depending on the value of β, this policy either picks the best estimated action (large values of β) or picks uniformly at random ($\beta = 0$). This policy makes the trade-off by drawing actions at random, favoring smoothly actions which are estimated optimal.

Another policy is the Upper-Confidence Bound (UCB) strategy which suggests taking:

$$a_{l+1} = \arg\max_{j=1,\dots p} \hat{f}_l(\tilde{x}, j) + \alpha\sqrt{\log(|\mathcal{D}|)/n_j} \qquad (6)$$

where $|\mathcal{D}|$ is the number of observations in $\mathcal{D}$ and n_j denotes the number of observations in $\mathcal{D}$ where features of the following set are known: $j \cup \{k : \tilde{x}_k \neq \text{“unknown”}\}$. This policy makes the trade-off by estimating an upper confidence level on the true Q-values by adding to their estimate a bound uncertainty which depends on the number of similar data already collected.

3 Experiments

To evaluate the methodology presented above, we used three toy models we built as well as three standard supervised learning benchmark datasets. The toy models were built in order to ensure that the methodology achieves proper performances against baselines and therefore validate quantitatively its interest. Amongst the three datasets we considered, there is the Boston Housing, the AMES and the Coronary Heart Disease (CHD) dataset. The first two, although not practically realistic for the intended use, serve as quantitative evaluation of the methodology performance on real-life data. The CHD dataset however is quite close to the motivation of this paper.

Amongst the six problems, four of them were regression problems, evaluated with Root Mean Squared Error (RMSE) metric. The two others were binary classification problems, evaluated with Area Under the Curve (AUC) metric.

For each of those problems, we built an Intelligent Questionnaire algorithm with a budget of $q = 3$ features to uncover based on training data. The training data was split in three equal parts: one to train the final predictor $\hat{m}$, one to train the Intelligent Questionnaire, and finally one to validate the training. We used `R` package `keras` to calibrate the feed-forward neural networks, relying on `rmsprop` optimizer, learning rate reduction on plateau as well as early stopping.[1] The overall performance results, obtained on 10 train-test splits are compiled in Table 1, followed by a focus on one of the Intelligent Questionnaire obtained on a benchmark dataset.

1 Problem specific details, such as network dimensions can be found at https://github.com/FredericLoge/SmartQuestions

3.1 Toy Models

Three toy models were considered in order to test our approach. In each case we simulated in total 6000 samples, 67 % of which are used for training and the rest for testing. As predictor functions, we used random forests with 100 trees, as implemented in the R package `randomForest`. For models #2 and #3, we will write ε a standard Gaussian noise generated independently of features X. For model #1 we considered the inaccuracy score function and for models #2 and #3 we used the squared prediction error.

3.1.1 Model #1, Set of Rules with Binary Features

We consider $p = 8$ mutually independent binary features

$$X_j \sim \mathcal{B}(0.5) \quad \forall j \in \{1, \dots, p\}.$$

Let $E(X)$ denote the union of arbitrarily chosen events:

$$\begin{aligned} E(X) \doteq &\{X_1 = X_2 = X_8 = 0\} \cup \{X_6 = 0, X_2 = X_3 = 1\} \\ &\cup \{X_8 = X_1 = X_3 = 1\} \cup \{X_4 = X_5 = X_6 = 0\} \\ &\cup \{X_3 = X_4 = X_2 = 1\} \cup \{X_4 = X_8 = X_1 = 1\} \\ &\cup \{X_3 = X_5 = X_7 = 0\}. \end{aligned}$$

We define $Y|X \doteq \mathbb{1}\{\bar{E}(X)\}$. The target is therefore defined deterministically based on X.

3.1.2 Model #2, Set of Rules with Binary and Continuous Features

We consider $p = 6$ mutually independent features

$$\forall j \in \{1, \dots, p-1\} \quad X_j \sim \mathcal{B}(0.5), \quad X_p \sim \mathcal{U}[0,1].$$

From there

$$Y|X = \mathbb{1}\{E_1(X) \cap \bar{E}_2(X)\} + 2\mathbb{1}\{E_2(X)\} + 0.2\varepsilon$$

with

$$\begin{aligned} E_1(X) \doteq &\{X_1 = 0, \{X_2 = 0 \cup X_6 > .7\}\} \cup \{X_4 = X_5 = 0, X_6 > .4\} \\ &\cup \{X_1 = X_3 = 0, X_6 > .8\}, \\ E_2(X) \doteq &\{X_1 = 1, \{X_3 = 1 \cup X_6 > .7\}\} \cup \{X_3 = X_5 = 1, X_6 > .6\}. \end{aligned}$$

In this toy model we add some stochasticity in the target and we consider mixed-type features.

3.1.3 Model #3: Regression with Continuous Features

We consider $p = 8$ mutually independent features

$$X_j \sim \mathcal{N}(0,1) \quad \forall j \in \{1, \dots, p\}.$$

From there,

$$Y|X = (X_2 + X_3)\mathbb{1}\{X_1 < 0\} + (X_4 + X_5)\mathbb{1}\{X_1 \geq 0\} + \sqrt{2}\varepsilon.$$

In this model, the target is a linear regression of the covariates, whose parameters depend on whether the first coordinate is strictly positive or not.

3.2 Benchmark Datasets

Three benchmark datasets where considered: the Boston Housing dataset [10], the more recent AMES dataset [8] and the Coronary Heart Disease dataset [2, 1]. The first two contain house prices and characteristics jointly. Because of the relatively low sample sizes we relied on linear regression models as prediction functions, rather than non-parametric models. For the Coronary Heart Disease problem, having relatively large sample size we used extreme gradient boosting with validation split for early stopping, relying on R package `xgboost`.

3.2.1 Boston Housing Dataset

This dataset contains 506 observations (one per suburb) and 13 variables, amongst which: the median value of owner-occupied homes (the target), crime rate, average number of rooms, pupil-teacher ratio.

3.2.2 House Prices Dataset, AMES

This dataset consists of 2930 observations of house value (log-scaled) and 81 characteristics such as overall quality, year of construction, surface information. The set of features was brought down to the following ten variables: *OverallQual*, *GrLivArea*, *YearBuilt*, *GarageCars*, *TotalBsmtSF*, *GarageArea*, *X1stFlrSF*, *FullBath*, *YearRemodAdd*, *LotArea*.

3.2.3 Coronary Heart Disease, CHD

This dataset contains 4238 observations of patients: sociodemographic information (e. g. gender), medical information (e. g. diabetes), medical examination (e. g. glucose)

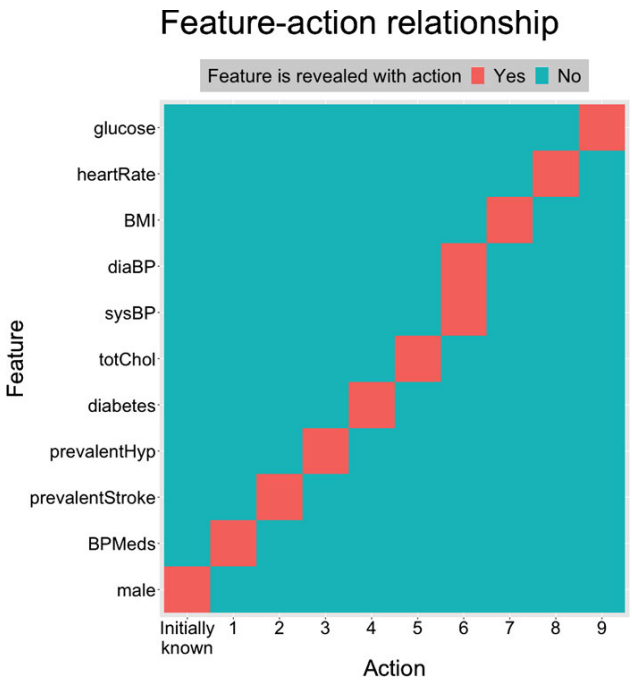

Figure 2: Feature-action relationship matrix for CHD problem: gender is assumed to be available initially, as it is considered to be a zero cost variables. Systolic and diastolic blood pressure are collected through a blood pressure measurement (action 6), all the others are collected with their own specific measurement / question e. g. blood glucose measurement for glucose, heart rate monitoring, asking for diabetes history.

and finally whether the patient developed Coronary Heart Disease during the following ten year period. For this problem, we assumed gender as an already-known feature and one-to-one relationship between actions and features except for one: action 6 reveals diastolic and systolic blood pressure simultaneously, see Figure 2.

3.3 Results

For each problem, we replicate the train/test split at random 10 times, calibrate our Intelligent Questionnaire as well as three baselines on the training set and evaluate on the test set. Toy model #1 and CHD being classification problems we used the AUC metric for comparison, whilst we used RMSE for all other problems.

In Table 1 we report the test set performance average and its standard deviation in parenthesis. The oracle corresponds to the model using all p features and best q subset relies on the fixed subset of features which performs best on the training set. CART algorithm with maximum depth q is a decision tree calibrated using `R` package `Rpart`. On all problems, whether it be toy models or benchmark datasets, the Intelligent Questionnaire outperforms both best q subset and the decision tree with maximum depth. For toy models, optimal performance bounds are provided as element of comparison.

In Figure 3 are represented the question sequences asked by the Intelligent Questionnaire on a test set of the AMES problem. The thickness of the arrows indicate the proportions of cases making the transitions. The best q subset was found to be *(OverallQual, GrLiveArea, YearBuilt)*. Those variables still matter a lot in the Intelligent Questionnaire, but it seems to be more interesting, depending on *OverallQual* observed to ask for *GarageArea* or *YearBuilt*. This questionnaire manages which information is more important to get depending on previously recorded values, as expected. Note that without initial information, the first feature asked for is systematically the same. In the CHD problem however, we witnessed that depending on the gender, known initially, the first question asked varied.

Table 1: Average and standard deviation of prediction performance on test sets, on 10 different train/test splits. On each problem, our approach performed better than the best q subset and the CART decision tree, getting close to the oracle predictor.

Problem	Metric	Bound	Oracle	Intelligent Questionnaire	Best q subset	CART, maxdepth = q
Toy models						
#1	AUC	1	1 (0)	0.87 (0.01)	0.75 (0.02)	0.81 (0.01)
#2	RMSE	0.2	0.3 (0.01)	0.37 (0.01)	0.46 (0.01)	0.41 (0.01)
#3	RMSE	$\sqrt{2}$	1.56 (0.02)	1.57 (0.03)	1.83 (0.03)	1.84 (0.02)
Benchmark datasets						
Boston Housing	RMSE		4.99 (0.57)	4.92 (0.54)	5.33 (0.54)	5.16 (0.64)
AMES Housing	RMSE		4.3 (0.22)	4.56 (0.14)	4.67 (0.19)	5.62 (0.15)
CHD	AUC		69.73 (1.92)	62.38 (3.2)	61.04 (4.35)	60 (3.41)

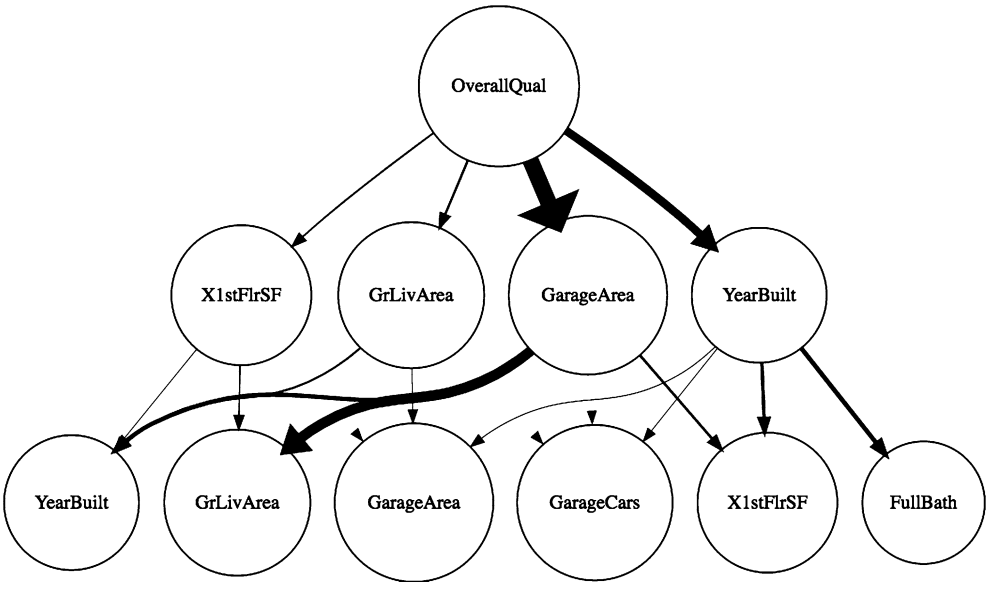

Figure 3: Diagram showing the paths taken during the application of the Intelligent Questionnaire on the test set on AMES dataset. Note: contrary to a decision tree which explicits the directions, this graph is only a representation of which features where asked, not why.

4 Reinforcement Learning on Toy Model

In practice, it is interesting to be able to update sequentially our model whilst using it, as presented in section 3. In this section we investigate our ability to quickly find a proper questionnaire on one of the toy models introduced earlier using a Reinforcement Learning approach [17]. The associated exploration-exploitation dilemma is the following: choosing questions enhancing our knowledge of $\mathbb{P}(X, Y)$ (*exploration*) versus choosing questions enhancing predictive power of Y given the partial information requested (*exploitation*). A related example is the 20 questions problem studied in [7]. Different formulations of the data acquisition setting (not always labelled, partially-known features) were proposed by [15].

4.1 Specifics

We considered toy model #3 presented in preceding section: consider $p = 8$ mutually independent random gaussians features

$$X_j \sim \text{Gaussian}(0,1) \quad \forall j \in \{1,\dots,p\}.$$

The target variable is defined by the simple linear model

$$Y = \begin{cases} X_1 + X_2 + \sigma\varepsilon & \text{if } X_3 < 0 \\ X_4 + X_5 + \sigma\varepsilon & \text{otherwise} \end{cases}$$

where $\varepsilon \sim \text{Gaussian}(0,1)$ is the noise random variable, independent of X. Parameter σ, which controls the noise level in the model will be set to $\sigma^2 = 0.4$ in order to have a Signal-to-Noise ratio of 5. We consider the problem of gradually sampling data using the questionnaire i. e. gathering only $q = 3$ elements of feature vector X at a time.

We start with 32 samples for each q-question combination and then at each iteration we sample a total of $32 \cdot \binom{p}{q} = 1792$ elements, accordingly with exploration policies. At every iteration, half of the samples are kept for learning function m and half for the neural network Q-value approximation. To approximate m we will rely on random forest algorithm. At each iteration, we will also follow greedily the current questionnaire on a free-test dataset of 2000 samples. This will allow us to track the performance improvements of the Intelligent Questionnaire constructed and see which exploration method performs well and under which conditions.

We considered the softmax and UCB exploration policies, presented in section 2.4, setting, quite arbitrarily two different exploration parameters. For all but one of the softmax approaches, we used the square root absolute prediction error instead of the squared error, because the latter has a highly-skewed distribution towards zero, which seems to be causing issues in the neural network training. It however may be that some parameters in the training on the neural network should be changed for the approach to perform properly.

The template neural network used for computing $(\hat{f}_2, \hat{f}_1)$ had the following characteristics: fully-connected neural network, with four hidden layers of 24 neurons, input of size 24 (the first 16 representing the partially-known vector, the last 8 indicating which action is selected) and with output size 1. Note that an alternative architecture could be to take as input 16 elements for the partially-known vector and in output 8 elements, one per action. Our architecture is slightly simpler to work with during training.

In order to obtain our results, we repeated each scenario i. e. exploration policy in 10 experiments with set seed values.

4.2 Results

In order to analyze and compare the algorithms performances we represent here the Root Mean Squared Error (RMSE) of prediction observed on test and acquired samples on figures 4 and 5 respectively.

Examining Figure 4, which represents the RMSE distribution over time steps and for the different policies, we can make several observations: (a) UCB methods take a longer time to reduce RMSE in contrast with softmax exploration, (b) UCB methods also present overall more variability than softmax experiment do, (c) the reward distribution matters, as the softmax policy relying on original squared errors is the only on which seems to stuck to a floor level, two times larger than the lower bound ($\sigma = \sqrt{0.4}$).

One can note that, because of the particular parameters, in the case of softmax the less-exploratory option ($\beta = 12$) is preferable to more exploratory one ($\beta = 6$). However, if we were to increase the parameter β too much, we would tend towards a purely greedy strategy i. e. picking the best estimated option which does not take into consideration the uncertainty of our estimates and risks focusing too much on sub-optimal actions. In the case of UCB, we can see that in this case it was preferable to pick a larger exploration bonus in order to make the solution converge faster.

We followed the analysis by checking which action was selected first in the learnt questionnaire and we observed that the softmax approach allows to converge quickly to picking question 3 first for almost all experiments around step 15, whilst this proportion is around 75 % for UCB(100) and 50 % for UCB (50).

Let us now analyze how the different exploration policies behaved on the acquired samples based on Figure 5. It appears that the RMSE on acquired samples drops only

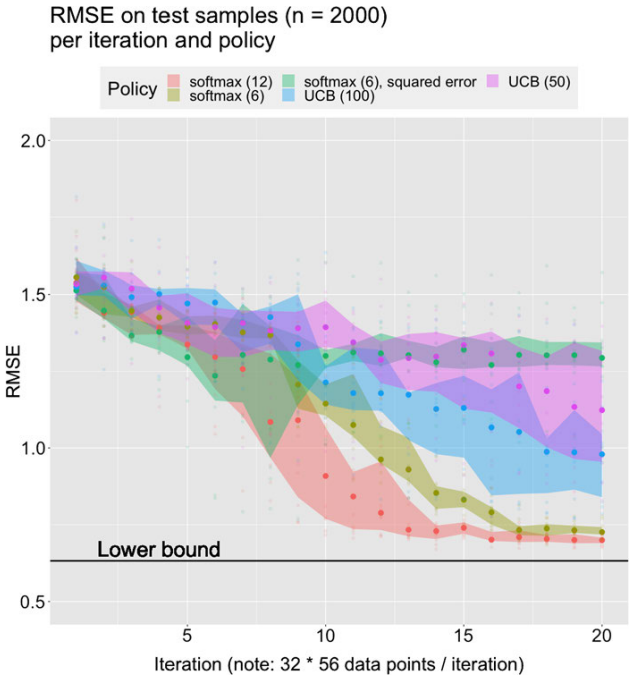

Figure 4: RMSE on test samples (n = 2000) per iteration and policy. The median RMSE is represented by the bold points, and the 1^{st} and 3^{rd} quantiles define the boundaries of the ribbon. The best possible performance is represented by the lower bound horizontal line, at level σ.

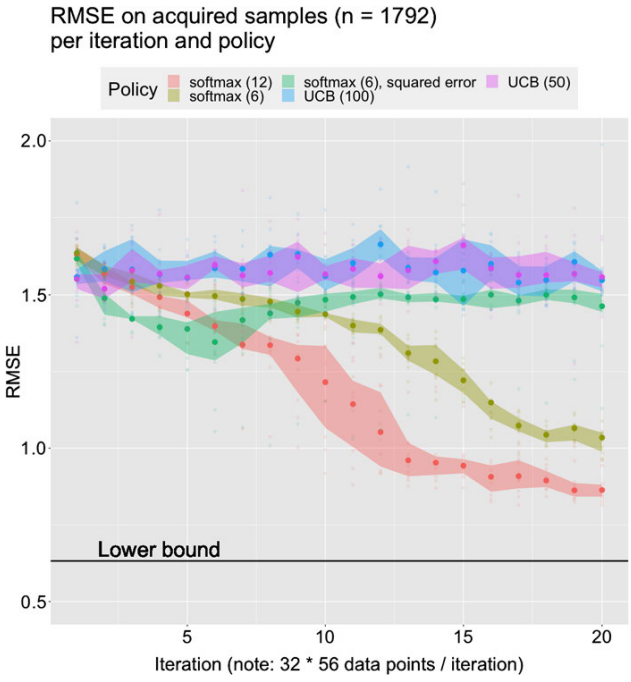

Figure 5: RMSE on acquired samples (n = 1792) per iteration and policy. The median RMSE is represented by the bold points, and the 1^{st} and 3^{rd} quantiles define the boundaries of the ribbon. The best possible performance is represented by the lower bound horizontal line, at level σ.

for the softmax policies, indicating that the samples collected were close to optimal selections. This finding was confirmed by checking the proportion of triplets sampled $(1, 2, 3)$ and $(3, 4, 5)$.

4.3 Suggestions

In this experiment, it appears that the softmax exploration policy seems to be more easy to deal with compared to UCB from an exploration perspective: it allows for much more exploration, without waiting for next steps; as such the learning happens more quickly and when we finally differentiate in terms of Q-values it clearly stands out, whereas in UCB it is not so clear, performs poorly in comparison in terms of cumulative regret as well as final regret. Also it seems more intuitive to compare Q-values amongst themselves rather than with an exploration bonus of a completely different nature. Also, it seems that reward distribution (at the last layer of the questionnaire) does matter, comparing the case where the squared error was taken instead of the root absolute error. Something we thoughtfully omitted, is that the reward function changes throughout iterations, since it is defined based on our prediction $\hat{m}$ of m, which is updated as more data is actually available. We are fortunate enough, or in a sufficiently simple setting at least, for this approach to be working. Now, it seems reasonable that we would have some idea of the variance of our prediction, which opens the possibility to learn not the exact error but a lower upper bound on it (optimism-based approach).

Finally, we could consider a more permissive version of the game, where we can actually ask more than three questions, but are penalized for it. That way, we could potentially learn much more, and it would open the way to more diverse exploration strategies.

5 Conclusion

In this work we built an adaptive predictive-questionnaire under constraint over the number of questions, motivated by the important balance between data acquisition and user experience. As this is a sequential decision-making problem we used a Markov Decision Process to model it. Furthermore, its episodic and hierarchical structure allowed us to apply an approximate dynamic programming approach to learn the best adaptive questionnaire based on available data on couple (X, Y), which is the standard setting in supervised learning. Regarding the evaluation of this approach, we have shown on three toy models as well as three classic benchmark datasets that our approach outperforms (a) a decision tree submitted to the same budget constraint of questions (b) classic models based on the most informative subset of questions. Option (b) being non adaptive and option (a) being limited to one-dimensional splits of data, our approach allows for much more flexibility. The application on the third dataset, which is the closest to our target application, showed that this approach can integrate easily some initially-known features and how actions unveil features. Finally, we showed on a toy model that our approach can be pursued in an online setting, which is interesting for practical applications. As a continuation of this work, we have a few ideas for further research.

Scaling with (p, q)

In our approach we relied on an approximation of state-value functions based on neural networks, without considering much on the dimension parameters. It is apparent that the higher $\binom{p}{q}$ is, the more difficult the problem becomes with blunt overall search. The exploration approaches from active & reinforcement learning will surely be handy to help identify q-sized subsets which are uninformative and handle this potential dimension issue.

Stopping Criterion

In the algorithm constructed, we assumed a budget constraint over the features requested. This approach makes sense in some applications as it reduces globally the number of requests from us to the user and it simplifies some computational aspects. We could also consider the case where we would stop asking questions as soon as we believe we have gathered enough information on X in order to predict Y to a satisfying level.

Performance Criterion

As we defined it, the performance of our algorithm is quantified through predictive power. From the user experience perspective, it might be worth taking into account the cost of each question/action: in the Coronary Heart Disease problem, some medical exams and checks might have higher costs than others and as such be favored differently. Bringing this work a step closer to Human-Computer Interaction, we could even consider choosing the appropriate question format e. g. radio-button choice versus numeric input. Overall this is about balancing predictive power and information retrieval cost, keeping in mind that such cost is related to user experience.

Truthful Responses

Throughout this work we assumed that when an element of X is asked for it is revealed exactly. As such, the same element is never asked for twice and we completely trust the response given. In some applications, the order and overall position of a question in a survey affects the answer given, which can sometimes have very important consequences, see [12]. Therefore, the answer given should be treated as a noisy version of the truth, where the noise actually depends on the order followed and previous answers that were given. This would lead us to model the interaction through a Partially Observable MDP [11].

Towards Human Computer Interaction, Online Evaluation

The framework we worked on suffers from the same issues one might find in the supervised learning setting, which assumes that independent and identically distributed pairs of random variables (X, Y) are observed: the data collection phase is often overlooked. Our objective in next steps with this project is to deploy such an algorithm on a tele-monitoring use case related to Air Liquide Home Care programs, where the difficulties linked to the questionnaire answering on tablets could be challenging for elderly patients. This requires going through regulatory protocols for access to data for algorithm training and piloting in a secure mode to carry out an online evaluation of the method. Finally, although this framework is presented for questionnaires in this paper, it is quite generic and therefore adaptable to other situations. For instance, it could be quite useful for developing dialog-based interfaces to intelligent systems.

References

[1] Framingham Heart study dataset. https://www.kaggle.com/amanajmera1/framingham-heart-study-dataset. Accessed: 2020-05-01.

[2] Framingham Heart Study, Three Generations of Dedication. https://framinghamheartstudy.org. Accessed: 2020-05-01.

[3] Bellman, R. *Dynamic Programming*, 1 ed. Princeton University Press, Princeton, NJ, USA, 1957.

[4] Bertsekas, D. P., and Tsitsiklis, J. N. *Neuro-dynamic programming*. Athena Scientific, 1996.

[5] Besson, R., Pennec, E. L., Allassonniere, S., Stirnemann, J., Spaggiari, E., and Neuraz, A. A model-based reinforcement learning approach for a rare disease diagnostic task. *arXiv preprint arXiv:1811.10112* (2018).

[6] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. *Classification and regression trees*. CRC press, 1984.

[7] Chen, Y., Chen, B., Duan, X., Lou, J.-G., Wang, Y., Zhu, W., and Cao, Y. Learning-to-ask: Knowledge acquisition via 20 questions. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2018), pp. 1216–1225.

[8] De Cock, D. Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project. *Journal of Statistics Education 19*, 3 (2011).

[9] Dunlop, M. D. Ontology-Driven, Adaptive, Medical Questionnaires for Patients with Mild Learning Disabilities. In *Artificial Intelligence XXXVI: 39th SGAI International Conference on Artificial Intelligence, AI 2019, Cambridge, UK, December 17–19, 2019, Proceedings* (2019), Springer, p. 107.

[10] Harrison Jr, D., and Rubinfeld, D. L. Hedonic housing prices and the demand for clean air.

[11] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and acting in partially observable stochastic domains. *Artificial intelligence 101*, 1-2 (1998), 99–134.

[12] Magelssen, M., Supphellen, M., Nortvedt, P., and Materstvedt, L. J. Attitudes towards assisted dying are influenced by question wording and order: a survey experiment. *BMC medical ethics 17*, 1 (2016), 24.

[13] Mwamikazi, E., Fournier-Viger, P., Moghrabi, C., Barhoumi, A., and Baudouin, R. An adaptive questionnaire for automatic identification of learning styles. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems* (2014), Springer, pp. 399–409.

[14] Nokelainen, P., Niemivirta, M., Kurhila, J., Miettinen, M., Silander, T., and Tirri, H. Implementation of an adaptive questionnaire. In *Proceedings of the ED-MEDIA Conference* (2001), pp. 1412–1413.

[15] Provost, F., Melville, P., and Saar-Tsechansky, M. Data acquisition and cost-effective predictive modeling: targeting offers for electronic commerce. In *Proceedings of the ninth international conference on Electronic commerce* (2007), pp. 389–398.

[16] Puterman, M. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 2014.

[17] Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Bionotes

Frédéric Logé
Air Liquide R&D, Jouy-en-Josas, Paris, France
CMAP, Polytechnique, Institut Polytechnique de Paris, Palaiseau, France
frederic.logemunerel@gmail.com

Trained in statistics, machine learning and programming, I am currently pursuing a PhD at the Centre de Mathématiques APpliquées (CMAP) of Ecole Polytechnique. My current research interest is in the application of Reinforcement Learning (RL) techniques to real-world problems. I am also interested in bandit problems and generative methods for data augmentation in RL.

Erwan Le Pennec
CMAP, Polytechnique, Institut Polytechnique de Paris, Palaiseau, France
XPop, Inria Saclay, Palaiseau, France
erwan.le-pennec@polytechnique.edu

I have been an associate professor in the Department of Applied Mathematics at École Polytechnique since September 2013. My research is carried out at the Centre de Mathématiques APpliquées (CMAP). I work there on data problems in signal processing, statistics and learning with a taste for applications. I am co-directing with Emmanuel Gobet the SIMPAS team (Signal IMage Probabibilités numériques et Apprentissage Statistique). I am also a member of the Inria Xpop team. I am in charge of many training programs offered by the school: PA of Data Science, MScT Data Science for Business and Data Science Starter Program of Polytechnique Executive Education. I also contributed to the creation of the M2 Datascience at the Institut Polytechnique de Paris.

Habiboulaye Amadou-Boubacar
Air Liquide R&D, Jouy-en-Josas, Paris, France
habiboulaye.amadou-boubacar@airliquide.com

I'm an International Expert in Data Science and Digital who had the good fortune to believe, at the beginning of the 21st century, that the advances in Machine Learning and Artificial Intelligence could be of great help in solving scientific problems for industrial applications. It is now a truism that the whole world will be impacted in profound ways by the revolution of Artificial Intelligence. Lucky enough to have jumped early into this exciting field, I completed my PhD in Machine Learning and Data Mining in 2006, and have ever since been addressing various data science challenges both in academic and industrial settings.