Adaptive predictive-questionnaire by approximate dynamic programming

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

Erwan Le Pennec
CMAP, Polytechnique, Institut Polytechnique de Paris, Palaiseau, FR
XPop, Inria Saclay, FR

Habiboulaye Amadou-Boubacar
Air Liquide R&D, Jouy-en-Josas, FR

ABSTRACT

As too much interaction can be detrimental to user experience, we investigate the computation of a smart 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.

CCS CONCEPTS

• **Computing methodologies → Planning under uncertainty**; **Approximate dynamic programming methods**.

KEYWORDS

Planning, Questionnaire design, Approximate dynamic programming

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 smart questionnaire 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 customer. 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 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. 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 exploitation issue and can thus focus on a planning approach [11]. We show how to use approximate dynamic programming [3, 4] to propose this adaptive

MuC'20 Workshops, Magdeburg, Deutschland

https://doi.org/10.18420/muc2020-ws111-264

sequence of questions so that it outperforms a fixed subset of q questions or a depth q CART decision tree.

The approach will be outlined in section 2, followed by experimental results on toy models and benchmark datasets in section 3. A general conclusion as well as ideas for improvements are presented in section 4.

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.

Consider $\{\mathcal{X}_j, j \in \{1, \ldots, p\}\}$ the set of one-dimensional spaces composing feature space $\mathcal{X}$ i.e. $\mathcal{X} \doteq \otimes_{j=1}^{p} \mathcal{X}_j$. Now consider $\tilde{\mathcal{X}} \doteq \otimes_{j=1}^{p} \left(\mathcal{X}_j \cup \{\text{“unknown”}\}\right)$ the space of partially-known feature vector $\tilde{X}$, where to each dimension is added the element “unknown”, encoding the fact this element has not yet been queried.

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

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

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 $\{1, \ldots, p\}$ i.e. the set of algorithms recommending a question to ask based on partial information on X.

We propose the following score function:

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

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 partially-known feature vector $\tilde{X}$ lives and $\mathcal{A} = \{1, \ldots, p\}$ is the action space, the indexes of feature elements we can collect. T is the transition function mapping $\tilde{\mathcal{X}} \times \mathcal{A} \times \tilde{\mathcal{X}}$ to $[0, 1]$ defined as

$$T(\tilde{x}, j, \tilde{x}') = \mathbb{P}_X\left(X_j = \tilde{x}'_j \mid \bigcap_{\{l: \tilde{x}_l \neq \text{“unknown”}\}} \{X_l = \tilde{x}_l\}\right) \quad (3)$$

for all $(\tilde{x}, j, \tilde{x}') \in \tilde{\mathcal{X}} \times \mathcal{A} \times \tilde{\mathcal{X}}$ such that $\tilde{x}_j =$ “unknown”, $\tilde{x}'_j \neq$ “unknown” and $\forall l \in \{1, \ldots, p\} \setminus \{j\}, \tilde{x}_l = \tilde{x}'_l$. In the case where $\tilde{x}_j \neq$ “unknown” and $\forall l \in \{1, \ldots, p\}, \tilde{x}_l = \tilde{x}'_l$, the question has already been asked, hence the transition probability is exactly one. In all other cases, the transition probability is exactly zero, because of incompatibilities.

R is the reward function defined for any $\tilde{x}$ such that the algorithm is stopped as

$$R(\tilde{x}) \sim \mathbb{P}_{(X,Y)}[score(\tilde{x}, Y)|\tilde{X} = \tilde{x}] \quad (4)$$

and equals 0 otherwise.

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.

2.3 Proposed solution

We assume that we have at our disposal the set $\{(x^{(i)}, y^{(i)}), i \in \{1, \ldots, n\}\}$, consisting of n independent and identically distributed instances from variables (X, Y). 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:

$$\forall (\tilde{x}, a) \in \tilde{\mathcal{X}} \times \mathcal{A}$$
$$Q_\pi(\tilde{x}, a) = \mathbb{E}_{\pi,(X,Y)}[score(\tilde{X}_q, Y)|\tilde{X}_t = \tilde{x}, A_t = a]. \quad (5)$$

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 $\{\hat{f}_j, j \in \{0, \ldots, q-1\}\}$ we would apply the Smart Questionnaire as presented in figure 2.

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 where regression problems, evaluated with Root Mean Squared Error (RMSE) metric. The two others where binary classification problems, evaluated with Area Under the Curve (AUC) metric.

For each of those problems, we built a Smart 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 Smart Questionnaire, and finally one to validate the Smart Questionnaire training. We

Learning value functions

Input data $\{(x^{(i)}, y^{(i)}), i \in \{1, \ldots, n\}\}$

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 $j \in \{q-1, q-2, \ldots, 1\}$

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 For

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

Figure 1: Learning value functions using approximate dynamic programming.

Smart Questionnaire algorithm

Interacting user knows (x, y)

Initialize $\tilde{x} \leftarrow$ "unknown"^{p}

For $j \in \{1, \ldots, q\}$

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 For

Return prediction of y : $\hat{m}(\tilde{x})$

Figure 2: Smart Questionnaire algorithm: choosing questions with task-trained neural networks $\{\hat{f}_j, j \in \{0, \ldots, q-1\}\}$ and providing final prediction of the target with prediction function $\hat{m}$.

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 Smart Questionnaire obtained on a benchmark dataset.

[1]Problem specific details, such as network dimensions can be found on the following Github repository 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

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

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

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

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

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

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 [1, 2]. 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: socio-demographic information (e.g. gender), medical information (e.g. diabetes), medical examination (e.g. glucose) 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 3.

3.3 Results

For each problem, we replicate the train/test split at random 10 times, calibrate our Smart 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 Smart 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 4 are represented the question sequences asked by the Smart 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 Smart 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.

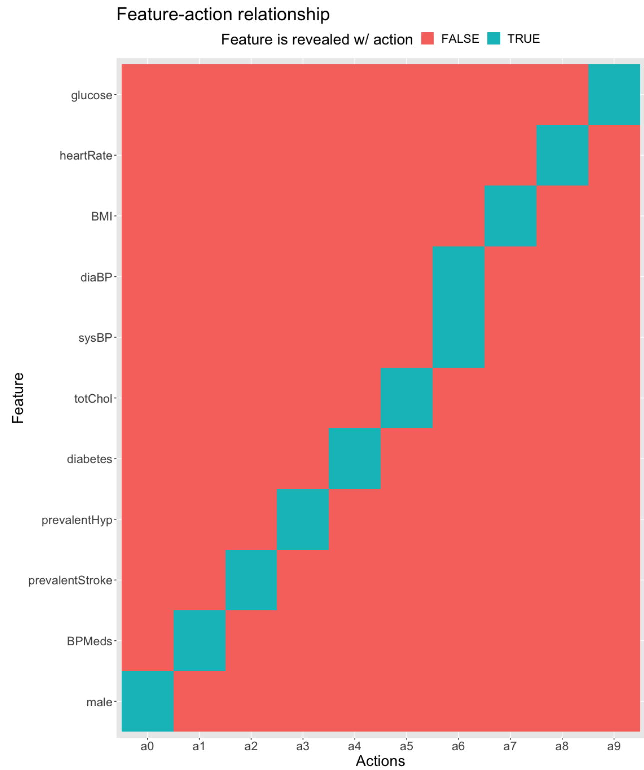

Figure 3: Feature-action relationship matrix for CHD problem: gender is assumed to be available initially, associated to action 0, as it is considered to be a zero cost variables. Systolic and diastolic blood pressure are collected through a blood pressure measurement (action 6), all the others are collected with their own specific measurement / question e.g. blood glucose measurement for glucose, heart rate monitoring, asking for diabetes history.

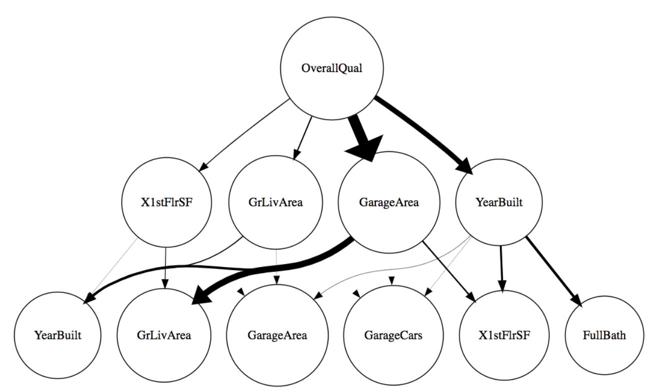

Figure 4: Diagram showing the paths taken during the application of the smart questionnaire on the test set on AMES dataset. Note: contrary to a decision tree which explicits the directions, this graph is only a representation of which features where asked, not why.

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	Smart 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)

4 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. Finally, 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. As a continuation of this work, we have a few ideas for further research.

Active & reinforcement learning. In this work we considered to have a dataset representative of couple (X, Y) already available and we aimed at setting the Smart Questionnaire algorithm once and for all. The problem could be extended to the case where observations can only be collected via the questionnaire, which could then be viewed as a Reinforcement Learning problem [17]. The associated exploration-exploitation dilemma would be 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].

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.

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.