Learning to Ask Medical Questions using Reinforcement Learning

Uri Shaham[1,2], **Tom Zahavy**[3], **Cesar Caraballo**[1], **Shiwani Mahajan**[1], **Daisy Massey**[1], and **Harlan Krumholz**[1]

[1]Center for Outcome Research and Evaluation, Yale University
[3]Technion
[2]Final Research

Abstract

We propose a novel reinforcement learning-based approach for adaptive and iterative feature selection. Given a masked vector of input features, a reinforcement learning agent iteratively selects certain features to be unmasked, and uses them to predict an outcome when it is sufficiently confident. The algorithm makes use of a novel environment setting, corresponding to a non-stationary Markov Decision Process. A key component of our approach is a guesser network, trained to predict the outcome from the selected features and parametrizing the reward function. Applying our method to a national survey dataset, we show that it not only outperforms strong baselines when requiring the prediction to be made based on a small number of input features, but is also highly more interpretable. Our code is publicly available at https://github.com/ushaham/adaptiveFS.

1. Introduction

Feature selection is an important topic in traditional machine learning (Li et al., 2018), which motivated a large number of widely adopted works, e.g., Lasso (Tibshirani, 1996). In various cases, the process of obtaining input measurements requires considerable effort (e.g., time, money, technology). For example, in medical datasets input features may correspond to lab tests, medical imaging results, or even answers to questionnaires, which are expensive and slow to produce. Allowing oneself to be able to accurately predict a response variable from a small set of input features is thus a desirable goal, which can be manifested in saving time, money, and sometimes even human lives.

As a running example, consider the case of a patient complaining to a family doctor about not feeling well. The doctor then asks the patient several questions about his current condition and medical background, and may also ask the patient to do some lab tests. Implicitly, the doctor is aiming at quickly collecting relevant details on the patient, that will allow her to have a clear understanding of the patients' medical status, and consequently decide on an appropriate action (e.g., medication prescription, admit to hospitalization etc.). The doctor would keep asking questions as long as this improves her understanding of the patient's medical status. Whenever the information acquired so far enables her to obtain a clear picture of the patient's status, she can make a decision about the next required steps. Making a (good) decision early in the process is beneficial, for example if the patient needs urgent treatment or when it is resourceful to acquire additional information (e.g., lab tests, medical diagnostics, or even doctor's time). Hence, a first challenge would be to navigate through the trade-off of gathering sufficient information while doing so as quickly as possible. Considering this example, it is also straightforward to realize that it would be highly sub-optimal to always select the same small subset of input features, regardless of the patient and the complaint. Indeed, when a 83 year old male complains about headache, we would expect the doctor to choose a different investigation path, comparing to a case of 7 year old girl complaining on stomachache. Put differently, it is often desirable to select the input features adaptively. Moreover, in cases like this it is also natural to select features iteratively, so that the k'th feature is selected after the values of the previous $k-1$ selected ones are known.

In this manuscript we aim at these desired attributes and propose a novel reinforcement learning (RL)-based approach for adaptive and iterative feature selection. In our RL framework, the state corresponds to the current agent's perception of the input sample, and the action space corresponds to the set of available input features. At each time step through an episode, a RL agent chooses a feature whose value is masked or unknown, and gets to observe the value of the specific feature. Once confident, the agent may choose to predict the label of the input sample and is rewarded based on the quality of the prediction. Throughout training, the agent learns to "ask" for the features which are most helpful for an accurate prediction of the label. The set of selected features may differ for each input sample, and the features are selected gradually and in adaptive fashion, so that each feature is selected based on the previously selected ones and their corresponding values. In this sense, the agent behaves in a more human-like fashion, comparing to standard feature selection models. Moreover, the trajectory (of selected features and their corresponding values) leading to each prediction is fully transparent and hence contributes to the model's interpretability.

We apply our approach to a national survey dataset, containing answers of patients to a large set of questions. We demonstrate that when limiting the prediction to be based on a small number of features, our approach outperforms strong off-the-shelf baselines, while also being more interpretable.

Our contributions are threefold: From a technical perspective, we apply reinforcement learning for adaptive feature selection, and propose a novel environment design to support it. Doing so, we propose a non-stationary Markov Decision Process (MDP) setting, in which the reward function is learned. From a medical perspective, we show how to design a human-like AI interface which adaptively selects information pieces in order to predict 4-year mortality. From an experimental perspective, we show that our approach outperforms strong off-the-shelf baselines, while also being more interpretable.

The remainder of this manuscript is organized as follows: In Section 2 we review related literature. The data cohort is described in Section 3. Our proposed approach is presented in Section 4. In Section 5 we report experimental results. Section 6 briefly concludes the manuscript.

Generalizable Insights about Machine Learning in the Context of Healthcare

This work exemplifies that interpretability of machine learning models does not have to come at a cost of lower prediction quality. In addition to interpretability, this work is focused around the efficiency (resource-wise) of the decision making process. We hope that the current work will serve as a step towards improving the pace and quality of decision making in the healthcare system. Oftentimes current decision-making systems are fed with unnecessary information, while other important information is missing. Our method may be helpful in making the decision process more efficient in terms of time and monetary resources. As a result, scarce resources might be allocated in a better way, and human lives may be saved.

2. Related Work

As the main motivation behind choosing a RL approach for the current adaptive feature selection task serve several recent applications of RL to the 20 questions game. In this game the player's goal is to predict the identity of an unknown character, where in each step of the game the player chooses a Yes/No question to ask and obtains the corresponding answer.

(Hu et al., 2018) define their action space as the set of possible questions and the state as a distribution over the possible characters. Their state update mechanism uses statistics of people's responses to questions. In addition, as a non-zero reward is obtained only at the end of the episode (corresponding to game win / lose), they augment their environment with a reward network, generating an intrinsic immediate reward at every time step, whose goal is to predict the true reward.

(Chen et al., 2018) use an Long Short Term Memory (LSTM) state update mechanism, so that the state space is the hidden space of the LSTM network. They use a naive-Bayes mechanism for making predictions at the end of each episode. In addition, their approach consists of two major elements: a DQN RL agent who plays the game and a knowledge acquisition mechanism, whose goal is to estimate answers distributions (regardless of the specific episode being played), which utilizes a matrix decomposition mechanism.

(Zhao and Eskenazi, 2016) propose a RL-based dialogue state tracking system, which they apply to the 20 questions game. Their approach combines reward signal with label supervision, which is related to our approach, as will be explained in Section 4.2.

Several works have focused on integrating feature selection methods with deep learning, e.g., (Li et al., 2016; Roy et al., 2015; Zhao et al., 2015; Louizos et al., 2017; Yamada et al., 2018; Balın et al., 2019). However, these approaches are neither adaptive nor iterative, which are core requirements in the setting we consider in this manuscript.

Lastly, the topic of intrinsic reward design (see, for example Zheng et al. (2018)), used also by (Hu et al., 2018), has drawn much interest in the RL community. It involves design of intrinsic reward functions, guiding the agent towards maximizing expected external reward (which may be sparse, or obtained at late time steps, for example). This has relations with our proposed approach, where we train and use use a guesser network to provide rewards to the RL agent, as will be explained in Section 4.2.

3. Cohort

In this section we describe the data cohort we apply our method to. Detailed instructions for data acquisition and preprocessing can be found in Appendix A.

3.1. Data Source

We used 10-year data (2002 to 2011) from the National Health Interview Survey (NHIS), which is an annual cross-sectional survey conducted by the National Center for Health Statistics (NCHS) that provides estimates on the health status, health-care access, and behaviors of the non-institutionalized US population[1]. The sample design of the NHIS follows a multistage area probability design, adjusting for non-response and allowing for nationally representative sampling of households and individuals. This sample design includes underrepresented groups. The NHIS questionnaire is divided into 4 cores: Household Composition core, Family core, Sample Child core, and Sample Adult core. The Household Composition core includes basic and relationship information about all individuals in the household. The Family Core file includes socio-demographic characteristics, health insurance coverage, basic indicators of health status, injuries, activity limitations, and access to and utilization of health care services separately for each family in the household. A random child and adult from each family are selected to gather more detailed information about them, constituting the Sample Child and the Sample Adult cores, respectively. In our study, we used the Sample Adult core files with variables supplemented from the other cores. The Household response rates ranged from 89.6% in 2002 to 79.5% in 2011. All survey participants provided informed consent before participation in the survey. This study received exemption from Yale University Institutional Review Board Committee because NHIS data are publicly available and de-identified.

The NHIS Linked Mortality files include data from all surveys between 1985 and 2014, linked to the National Death Index (NDI), with follow-up to the date of death or 31 December 2015[2]. An estimated 95.4% of baseline participants had the eligible mortality follow-up information. The NCHS at the Centers for Disease Control and Prevention used post-stratification re-weighting based on the U.S. population to account for ineligible follow-up[3]. The NDI file provides data on the mortality status, year of death, quarter of year, and cause of death (categorized into the following groups based on ICD-9 and ICD-10 codes – heart disease, cancer, chronic lower respiratory disease, cerebrovascular diseases, diabetes, pneumonia and influenza, Alzheimer's disease, kidney disease, and unintentional injuries).

3.2. Study Population

A total of 282,001 adults aged 18 years and above were interviewed between 2002-2011, of which we excluded those with missing information on their mortality follow-up ($n = 12,905$) resulting in a study sample of $n = 269,069$ individuals

[1] https://www.cdc.gov/nchs/nhis/about_nhis.htm

[2] https://www.cdc.gov/nchs/data-linkage/mortality-public.htm

[3] https://www.cdc.gov/nchs/data-linkage/mortality-methods.htm

3.3. Outcome Definition

Our outcome of interest was death within 4 years of the date of interview. We used 16 quarters of a year from the quarter of the interview to the quarter of death as our follow-up time, as the NHIS Linked Mortality files provides the year and quarter of death only.

3.4. Candidate Variables

Because some of the content of the NHIS questionnaire changed over the years depending on the data needs for current health topics, we decided to include questions that were consistent across all 10 years (1,022 variables out of 2,360 total). We then excluded those variables that were only asked to the participant conditional on a prior answer to another question (811 variables, labeled as "daughter variables"), those that had $> 80\%$ missingness (8 variables), and those that contained redundant information with other variables (for example, identifiers, repeated information; 45 variables). The final dataset had 211 variables, of which 188 were interview variables (candidate variables for the model) and 23 were identifiers and outcome variables. For the 10 continuous variables with missing values (ranging from 1%–27% missing rates), we used single-value imputation with median. For 11 categorical variables with missing values (ranging from 3%–78% missing rates) we created "missing" as a separate category. Finally, we added the daughter questions (85 variables) for the top 25 variables most correlated with outcome and for the top 25 variables identified using an XGBoost model, increasing the total number of candidate variables to 273 variables. Categorical variables were one-hot encoded.

4. Adaptive Feature Selection using Reinforcement Learning

4.1. Preliminaries

4.1.1. Reinforcement Learning

Reinforcement Learning (RL) is family of algorithms aimed at solving Markov Decision Processes (MDPs). A MDP is represented as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ is a set of states (also called state space), $\mathcal{A}$ is a set of actions (also called action space), $\mathcal{P}$ is a set of state transition rules

$$\mathcal{P}(s', s, a) = \text{Prob}(S_{t+1} = s' | S_t = s, A_t = a),$$

specifying the distribution over the state space for the next state given a current state and action, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a reward function and $0 < \gamma \leq 1$ is a discount factor.

The RL paradigm consists of two major elements: an agent and an environment. The agent follows a policy, defined as a function $\pi : \mathcal{S} \to \text{dist}(\mathcal{A})$ which maps each state to a distribution over the action space. Doing so, it interacts with the environment by choosing actions from $\mathcal{A}$ that may let it move between states. At each step t of the episode, being in state s_t, the agent chooses an action a_t from $\pi(s_t)$, obtains a (negative, zero or positive) reward $r_t = \mathcal{R}(s_t, a_t)$ and moves to state s_{t+1}, which is sampled from $\mathcal{P}(\cdot, s_t, a_t)$.

The agent's goal is to learn a policy that maximizes the expected reward

$$\arg\max_{\pi} \mathbb{E}[R_T] = \arg\max_{\pi} \mathbb{E}\left[\sum_{t=1}^{T} \gamma^t r_t \;\middle|\; a_t \sim \pi(s_t),\, s_{t+1} \sim \mathcal{P}(\cdot, s_t, a_t)\right],$$

where T is the length of the episode.

Rather than supervised learning, in which ground truth labels are known during training, such knowledge is absent in RL. Instead, the reward signal is the driving force of the learning. This weaker form of supervision signal makes RL systems take longer to train comparing to supervised learning algorithms. However, it is applicable to many scenarios where supervised learning is not an available approach. In recent years, RL has been an active research area in the machine learning community, and some dramatic successes demonstrated its potential, e.g., (Silver et al., 2016).

4.1.2. Q Learning and Deep Q Learning

In reinforcement learning, the Q value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ of a policy π corresponds to the expected reward given the current state and the current chosen action, where the agent follows π after taking the action. By definition, the Q function satisfies a recursive relation, known as Bellman equation (Sutton and Barto, 2018):

$$Q^\pi(s,a) = \mathcal{R}(s,a) + \mathbb{E}_{s' \sim \mathcal{P}(\cdot,s,a),\, a' \sim \pi(s')} \left[\gamma Q^\pi(s',a')\right].$$

A policy π^* maximizing Q^π for every $s \in \mathcal{S}$, $a \in \mathcal{A}$ yields the optimal Q function, denoted Q^*, whose corresponding Bellman equation is

$$Q^*(s,a) = \mathcal{R}(s,a) + \mathbb{E}_{s' \sim \mathcal{P}(\cdot,s,a)} \left[\gamma \max_{a'} Q^*(s',a')\right]. \tag{1}$$

In words, equation (1) means that the current expected reward equals the sum of the immediate reward and the best (over choice of action) possible expected reward of the next state.

In Q learning (Watkins and Dayan, 1992), the Q function is iteratively updated, in order to have the Bellman equation satisfied:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[\mathcal{R}(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right].$$

Deep Q network (DQN, (Mnih et al., 2015)) is arguably the first major milestone in utilizing deep neural networks for reinforcement learning. It is an instance of Fitted Q Iteration (Ernst et al., 2005), where the Q function is represented by a neural network (DQN - deep Q network), parametrized by θ. The net is trained to minimize the squared difference between the left and right hand sides of the Bellman Equation

$$L_{DQN}(\theta) = \left(\left(\mathcal{R}(s,a) + \gamma \max_{a'} Q(s',a';\hat{\theta})\right) - Q(s,a;\theta)\right)^2, \tag{2}$$

where $Q(\cdot,\cdot;\hat{\theta})$ is a target network, having identical architecture as the Q network, and whose parameter vector $\hat{\theta}$ is copied from the Q network parameter θ every certain number of training iterations. The method makes use of Experience Replay, which is a storage buffer holding historical instances of the form (s_t, a_t, r_t, s_{t+1}) which are experienced by the agent during the episode. At the end of each training episode, a minibatch of such instances is randomly sampled from the buffer and uses for gradient computation, following (2).

Double DQN (DDQN (Van Hasselt et al., 2016)) is an improvement of DQN, aiming to be less prone to overestimation of Q values. It differs from DQN in that the evaluation of the Q values and the selection of the best action uses different parameter vectors: the selection is done using the online parameter θ, while the evaluation uses the target parameter vector $\hat{\theta}$.

$$L_{DDQN}(\theta) = \left(\left(\mathcal{R}(s,a) + \gamma Q(s', \arg\max_a Q(s', a; \theta); \hat{\theta})\right) - Q(s,a;\theta)\right)^2 . \tag{3}$$

4.2. The Proposed Approach

In this section we describe our proposed approach for adaptive selection of small number of questions that will allow to accurately predict the outcome variable.

4.2.1. Rationale

Our questionnaire dataset D is a $n \times d$ matrix, where $D_{i,j}$ corresponds to the answer of patient i to question j. Each episode is performed on a single patient. Let x correspond to the (d-dimensional) feature vector of the patient and let $y \in \{0,1\}$ be the corresponding label. A key component in our design is a guesser function $G : \mathcal{S} \to [0,1]$ whose goal is to predict the outcome y from any state s, which corresponds to the unmasked entries of x. At the beginning of each episode, the patient's feature vector x is completely masked. In each time step throughout the episode, the agent chooses to unmask a certain entry of x. When ready, the guesser may choose to predict the outcome y from the unmasked features. When this is the case, the agent is rewarded according to the quality of the guesser's prediction. Therefore, it learns to select features that will allow the guesser to make an accurate guess. During training, two separate optimization procedures take place, where both the guesser and the agent are being trained. Our approach is depicted in Figure 1.

4.2.2. The MDP

Our MDP is defined as follows:

- State space: $\mathcal{S} = \mathbb{R}^{2d}$, where the first d entries correspond to the patient's answers to the d survey questions , and for $i = 1, \ldots, d$, the $d + i$ entry is set to 1 if question i was chosen and 0 otherwise.

- Action space: $\mathcal{A} = \{1, 2, \ldots, d+1\}$, where actions $1, \ldots, d$ refer to choosing the corresponding question and action $d+1$ refers to making a guess about the outcome variable. To prevent the agent from selecting the same feature more than once, we apply masking, as will be explained in Section 4.2.5.

- State transition rules: At the beginning of an episode, the initial state is simply a zero vector of length $2d$. At each time step throughout the episode, if the action refers to asking a new question ($1 \leq a \leq d$), state s' is obtained from state s by unmasking the a'th entry of s (i.e., setting $s'_a = x_a$) and marking that question a was asked (i.e., setting $s'_{a+d} = 1$). If the agent chooses to make a guess (i.e., $a = d+1$), the state remains unchanged and the episode terminates.

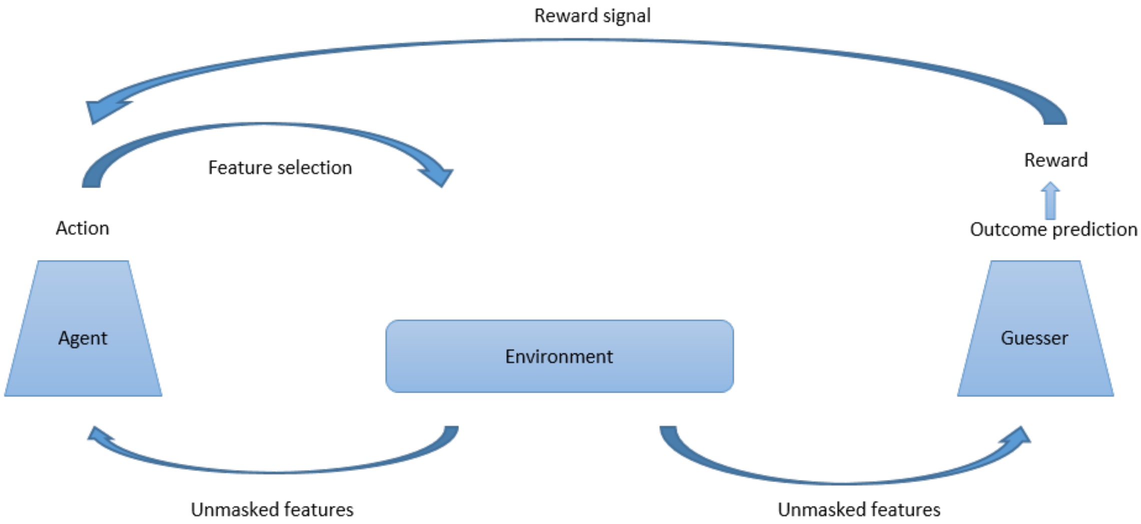


Figure 1: The proposed approach. The agent selects features to unmask. The guesser uses the unmasked features to predict the outcome and determines the agent's reward. The agent learns to select features that will allow accurate prediction.

- Reward function: For any question action ($1 \leq a \leq d$), the reward is a small random number: $\mathcal{R}(s, a) = 0.1 \cdot \text{Unif}(0, 1)$. For a guess action (i.e., $a = d + 1$), the reward corresponds to the probability the environment guesser G assigns to the true label, $\mathcal{R}(s, d + 1) = \text{Prob}(G(s) = y)$. Observe that the fact that the reward function is parametrized by the guesser network, which is trained as well along with the agent, makes the MDP non-stationary, as during the course of training the guesser's weights change and correspondingly so does the reward function. This non-stationarity of the MDP deviates from the majority of recent RL works, which consider a stationary setting. To cope with the challenges the non-stationary setting introduces, we alternate between training the guesser and the agent, as will be explained in Section 4.2.5.

4.2.3. The Environment

As mentioned above, we augment our environment with a guesser network, which is trained along with the RL agent. The guesser G maps a state s to $G(s)$, which is the probability assigned by the guesser to a positive outcome $p(y = 1|s)$. At the beginning of each episode, we reset the state so that only the age, gender and race features are visible and all other features are masked. At each step of the episode, an additional feature becomes visible, corresponding to the agent's chosen actions. The episode terminates whenever the number of steps exceeds the pre-defined number of steps, or earlier, if the agent chooses to make a guess about the patient's outcome. Whenever the agent chooses to make a prediction, the guesser network is called to predict the outcome from the current state (i.e., from the collection of all unmasked features). The probability that the guesser assigns to the correct class (which is known during training) uses as the reward which is passed on to the agent.

4.2.4. The Agent

We chose to use a DDQN agent. In our experiments this model performed at the same level or even outperformed more sophisticated recent models such as PPO (Schulman et al., 2017).

4.2.5. Additional Design Choices for Performance Improvements

Several implementational details allow to improve the performance of the algorithm. Below we briefly describe them.

Oversampling Our dataset is highly unbalanced: less than 5 percent of the patients had positive outcome (died within four years from the date of filling the questionnaire). To avoid bias toward the large class, in each training episode we sample a patient from one of the classes with equal probability (i.e., we over-sample the small class), so that roughly similar number of patients from each class are seen during training.

Alternating training To improve the training stability in our non-stationary setting, we trained the guesser and the agent intermittently, switching between them once in 1000 episodes. This way, during each such 1000 episodes period, when the agent is being trained, the guesser network remains fixed, so that the MDP is in fact ("locally") stationary.

Pre-training the guesser network We pre-train the guesser G as a classifier, where all features are visible. When setting-up the environment, we initialized the guesser network using the parameters of the pre-trained guesser.

Early Stopping Unlike typical RL works, we know the labels of the training data, which allows us to dedicate a portion of the data for validation set and apply an early stopping mechanism. Specifically, every 1000 training episodes we run our agent on the validation set and record its AUC. Training stops when no significant improvements of AUC occurs. In inference, we use the model with the highest AUC on the validation set.

Masking In order to ensure that the agent avoids selecting the same feature more than once, we apply a multiplicative mask to the agents' Q values so that Q values of features that were already selected are multiplied by zero and consequently will not be selected again.

ϵ-greedy sampling probabilities In order to explore new paths, it is a common practice to select the action that maximizes the Q values in equations (2) and (3) with probability $1 - \epsilon$, rather than with probability 1, and select a action uniformly at random with probability ϵ. Practitioners typically use a time-decay policy for ϵ. Here, instead of using uniform probabilities for action sampling, we sample each action a with probability which is proportional to the absolute correlation of the corresponding question with the target label over the training set. This heuristic helps choosing actions which are more informative about the target with higher probability.

Architectures Our best results were achieved using straightforward depth 3 MLP architectures for both the guesser and the Q network. We also experimented with a more sophisticated design, where the state update mechanism is a Long-Short-Term-Memory (LSTM (Hochreiter and Schmidhuber, 1997)) cell. In this design the input to the LSTM cell at each time step is an embedding of the identity of selected feature, concatenated to the actual value of the feature.

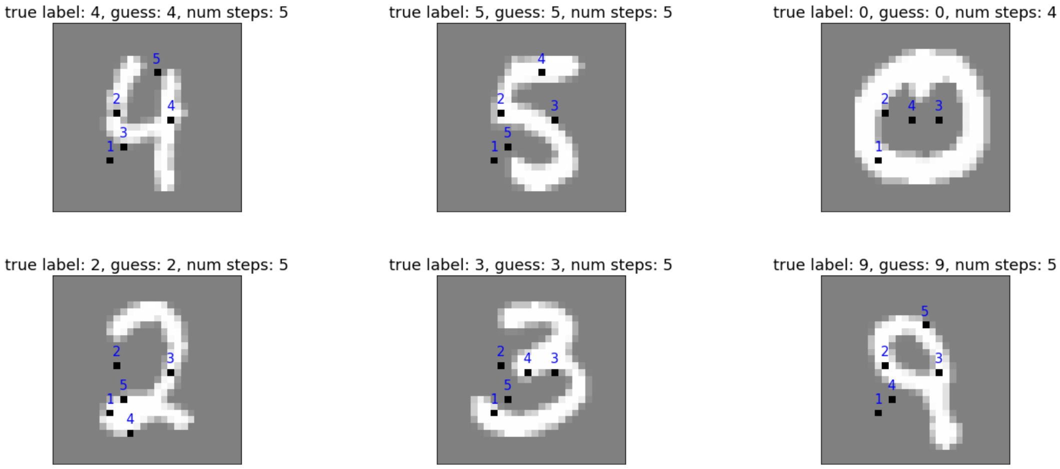


Figure 2: Demonstration of the proposed approach on the Mnist handwritten digit dataset. Unmasked features appear in black. The order in which the features were selected is indicated in blue.

5. Experimental Results

We begin this section by a visual demonstration of the feature selection process. We then report our primary results in predicting 4-year mortality from the national survey dataset, and comparing it to major off-the-shelf baselines. We provide results of ablation studies, justifying our main design choices, and of experimenting with our approach under off-policy regime. All results reported in this section are averaged over 5 identical trials with random splits of the dataset to train (67%) and test (33%) sets.

5.1. Demonstration on Mnist

For the purpose of demonstration, we applied our approach on the Mnist handwritten digits dataset, where each pixel is a feature. The goal is to predict the label of the handwritten digits based on at most five pixels. The agent was able to correctly recognize the handwritten digit from at most 5 pixels 56.9% of the time. Figure 2 shows examples of predictions made by the algorithm, and the corresponding selected features.

5.2. Main Results

Our goal is to obtain a good prediction for the outcome variable while considering a small number of features for each patient. As baselines, we choose to compare our result to two off-the-shelf classifiers: a Decision Tree (DT) and XGBoost (XGB) (Chen and Guestrin, 2016).

A decision tree is a fundamental and widely-used machine learning algorithm. It has several known disadvantages, such as its greedy training procedure and its simplistic modeling of the feature space (axis-aligned rectangles), which often prevent it from performing on par with the state-of-the-art models. However, in many cases it is nevertheless a strong performer. In our context, a DT has three attractive attributes which make it an appropriate baseline: first, specifying the depth of the tree, we can limit the number of features leading to each prediction made by the model. Second, different patients might correspond to different paths down the tree, so that different subsets of features may be used to obtain the predictions of different patients. Third, a DT is fully transparent, so that the predictions have high interpretability.

XGBoost is arguably considered as the state-of-the-art model for tabular data and is a popular choice by practitioners. Being an ensemble method, its interpretability is low, in the sense that it is difficult to provide a clear reasoning to the prediction made by the algorithm. Yet, given a trained model it is possible to obtain feature importance scores, describing how important each feature is to the predictions made by the model (see (Lundberg et al., 2018), for example). In addition, we use the feature importance scores in order to reduce the number of features prior to applying our proposed approach. The results reported in this manuscript were obtained by letting the agent select features out of the 50 most important features of a XGB model. We get similar results for the 100 most important features as well. The list of these 50 features appears in Appendix B.

Figure 3 shows the test AUC results of the proposed approach, comparing to DT and XGB. For every number of features k, the DT was developed up to depth k, the XGB model was trained on the subset of k most important features of a full XGB model (trained on all features), and the RL agent was trained to choose $k-3$ features, as it was forced to select the age, gender and race features as starting point. Our proposed RL approach consistently

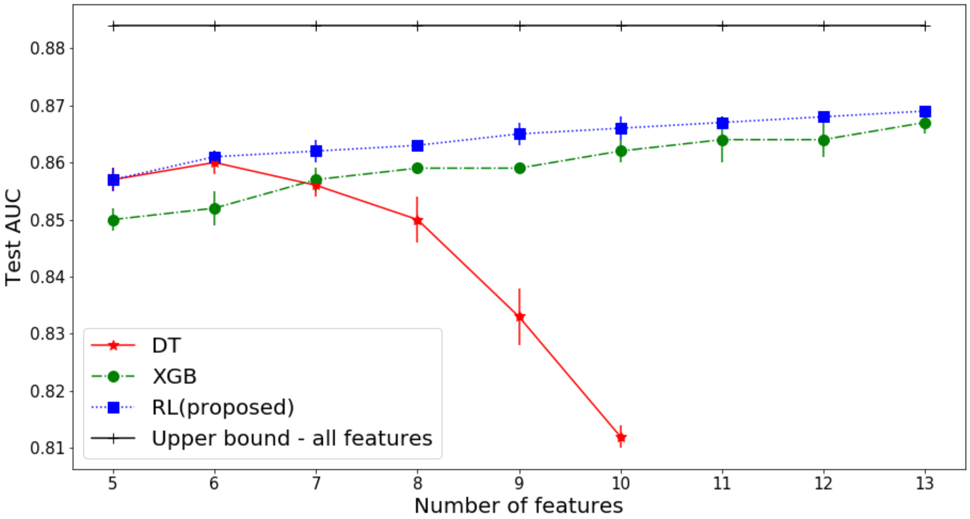
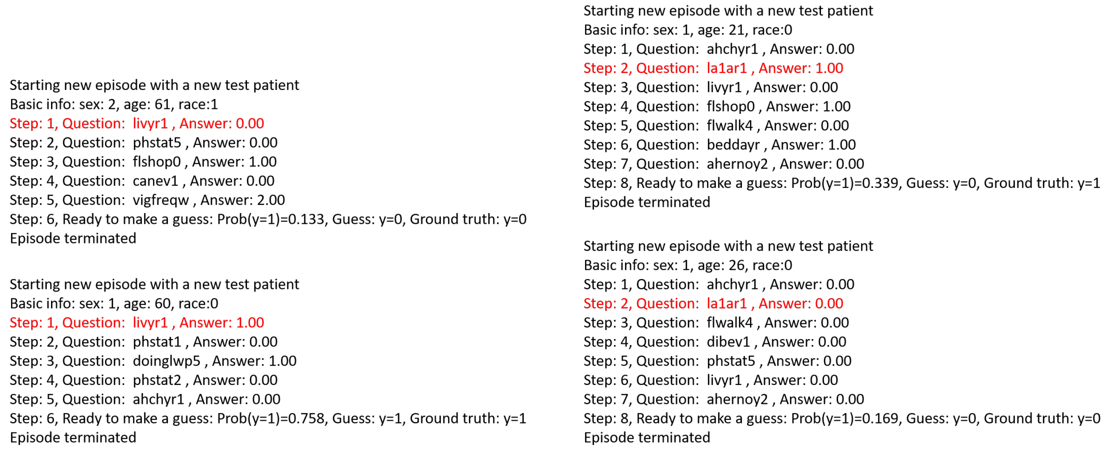


Figure 3: Test AUC performance of DT, XGB and our proposed RL approach. The upper bound was obtained using all features; the same performance was achieved by both XGB and a MLP network.

outperforms both the DT and the XGB models for all tested numbers of features. Moreover, the RL models improve monotonically as more features are allowed to be selected, as is also the case for the XGB models. The DT models, however, start to overfit for more than 6 features.

The advantage of the proposed approach over XGB manifests not only in terms of prediction accuracy, but also in terms of interpretability, through fact that one gets to observe the sequence of unmasked features leading to the each prediction. Figure 4 shows two case studies of the model predictions for patients from the test set. On the left hand side of Figure 4 the agent acts on the data of two patients from the same race, having similar age and different sex. For both patients the agent chooses to unmask a feature containing information about the patients' liver condition (marked in red). The bottom patient had a liver condition while the top patient did not have such condition. From the second step and on, the agent chooses to unmask different features for each patient, leading to a prediction of low probability for mortality for the top patient (whose unmasked features revealed is not in a poor physical condition, did not have cancer and does not need special equipment to go out), whereas the bottom patient (whose unmasked features revealed was not in a good physical condition and did not go to work last week) is assigned a high probability for 4-year mortality.



```
Starting new episode with a new test patient
Basic info: sex: 2, age: 61, race:1
Step: 1, Question:  livyr1 , Answer: 0.00
Step: 2, Question:  phstat5 , Answer: 0.00
Step: 3, Question:  flshop0 , Answer: 1.00
Step: 4, Question:  canev1 , Answer: 0.00
Step: 5, Question:  vigfreqw , Answer: 2.00
Step: 6, Ready to make a guess: Prob(y=1)=0.133, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 60, race:0
Step: 1, Question:  livyr1 , Answer: 1.00
Step: 2, Question:  phstat1 , Answer: 0.00
Step: 3, Question:  doinglwp5 , Answer: 1.00
Step: 4, Question:  phstat2 , Answer: 0.00
Step: 5, Question:  ahchyr1 , Answer: 0.00
Step: 6, Ready to make a guess: Prob(y=1)=0.758, Guess: y=1, Ground truth: y=1
Episode terminated
```

```
Starting new episode with a new test patient
Basic info: sex: 1, age: 21, race:0
Step: 1, Question:  ahchyr1 , Answer: 0.00
Step: 2, Question:  la1ar1 , Answer: 1.00
Step: 3, Question:  livyr1 , Answer: 0.00
Step: 4, Question:  flshop0 , Answer: 1.00
Step: 5, Question:  flwalk4 , Answer: 0.00
Step: 6, Question:  beddayr , Answer: 1.00
Step: 7, Question:  ahernoy2 , Answer: 0.00
Step: 8, Ready to make a guess: Prob(y=1)=0.339, Guess: y=0, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 26, race:0
Step: 1, Question:  ahchyr1 , Answer: 0.00
Step: 2, Question:  la1ar1 , Answer: 0.00
Step: 3, Question:  flwalk4 , Answer: 0.00
Step: 4, Question:  dibev1 , Answer: 0.00
Step: 5, Question:  phstat5 , Answer: 0.00
Step: 6, Question:  livyr1 , Answer: 0.00
Step: 7, Question:  ahernoy2 , Answer: 0.00
Step: 8, Ready to make a guess: Prob(y=1)=0.169, Guess: y=0, Ground truth: y=0
Episode terminated
```

Figure 4: Two case studies of the agent behavior.

In the second case study, on the right hand side of Figure 4, the agents acts on two young patients of the same race and sex. The second unmasked feature reveals that the top patient is limited in some way, while the bottom one is not. This leads to different unmasked features for each patient, revealing that the top patient also has necessity for special equipment and had bed days during the past 12 months, while for the bottom patient no potential negative medical conditions are recognized. The episodes end with a 4-year mortality probability assignment which is twice as high for the top patient (who indeed died within 4 years of the questionnaire). Additional examples for the question trajectories leading to the predictions of our approach are provided in Appendix C.

5.3. Off-Policy Experiments

Off-policy learning is an important area in RL. It corresponds to cases where the states the agent observes are not a consequence of the agent's policy. In cases like this there might be a decrease in the performance of the agent, as it was not trained on such states. Q learning is an off-policy learning method, as it updates the Q function independently of the policy it currently follows, .i.e., the updates are based on tuples (s_t, a_t, r_t, s_{t+1}) from past versions of the policy. Being an off-policy learner, DDQN handles such cases by design. To verify the performance of our proposed approach is stable under an off-policy regime, we considered a case where some features are given to us "for free", i.e., along with the age, gender and race of the patient we may also observe additional features, without the agent explicitly choosing to unmask these features. In order to investigate the performance of our proposed approach under such a scenario, we train the guesser and agent as usual, but modify our inference procedure, so when the environment restarts the state at the beginning of any episode, along with unmasking the age, gender and race features, it also unmasks a randomly chosen feature (selected randomly for each new test patient). Applying this test procedure for $k = 10$ features, we observed that the AUC over the test set decreased from 0.865 to 0.862. While a slight decrease is somewhat expected, the decrease is relatively minor, and the model seem to perform roughly on the same level as before.

5.4. Ablation studies

In this section we investigate the contribution of the oversampling, guesser pre-training and alternation of the training of the guesser and Q network. The results for $k = 10$ features appear in Table 1.

Configuration	Test AUC
Full approach (proposed)	0.865 (0.003)
No guesser pre-training	0.856 (0.003)
No oversampling	0.855 (0.002)
No alternation	0.834 (0.002)

Table 1: Ablation studies.

As can be seen, absence of any of the three elements causes a decrease in performance, comparing to the full approach.

5.5. Question Embedding

Figure 5 shows the embedding of the questions, obtained from training our proposed approach using the LSTM architecture. Interestingly, the plot manifests several intuitive relations between pairs of features. For example, the feature vectors of *la1ar1* (limited in any way) and *la1ar2* (not limited in any way) are in opposite directions, and so are *phstat1* (excellent health status) and *phstat5* (poor health status), as well as these of *livyr1* (told to have liver condition) and *livyr2* (was not told). On the other hand, the feature vectors of *hiscodi32* and *origin_i*, both corresponding to Hispanic origin, are close.

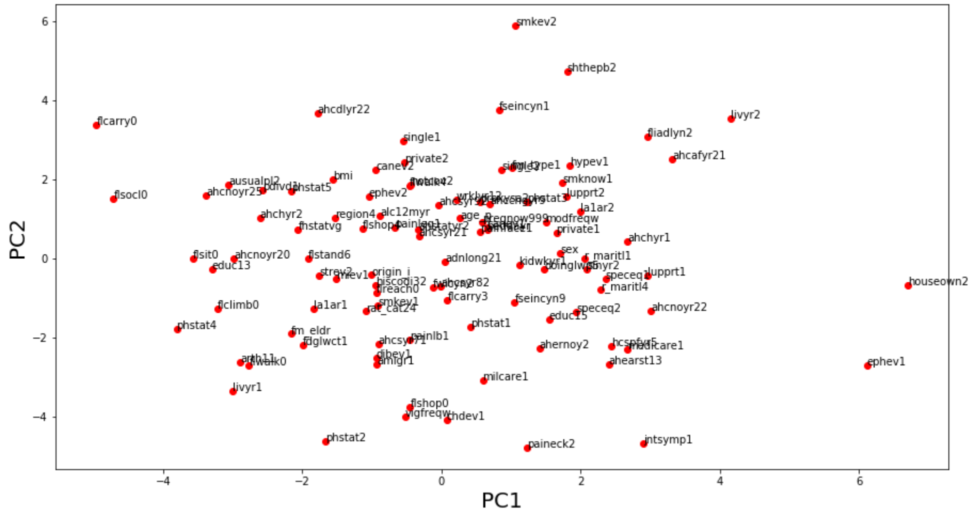


Figure 5: Question embedding using LSTM architecture. The two plotted dimensions are the first two principal axes of the 64-dimensional embedding.

5.6. Technical details

We used the sklearn implementation of DT, where for k features we built a full binary tree of depth k. We were not able to obtain better results using pruning. For XGBoost we used the python *xgboost* package, with ensemble size of 100. For each number k of features we manually tuned the tree depth to achieve the best performance. For both DT and XGB models we used class weights for training, such that the weight for each class was inversely proportional to its relative size.

For the proposed RL approach, we used the same hyperparameter setting for all numbers of features. The guesser architecture was a multi-layer perceptron (MLP) architecture, with three hidden layers of 250 PReLU units each and a softmax output layer. The Q network architecture had two hidden layers of 128 ReLU units each and sigmoid output layer. Weight penalty was added to the DQN loss. For both networks we used a learning rate decay policy, with initial value of 10^{-4} and division by 10 every 17500 training episodes, with minimal learning rate of 10^{-6}. We set the reward decay factor γ to 0.95.

6. Conclusions

In this manuscript we proposed a reinforcement learning-based approach for adaptive feature selection and applied it to a national survey dataset, where the goal is to predict 4-year mortality of patients. We demonstrated that our approach outperforms standard baseline models for the same task, while also being more interpretable than its closest competitor models. In the future we plan to use this approach as a basis for recommendation system and extend it to other types of medical data, such as images and natural language. In addition, we plan to incorporate feature costs and non-symmetric error costs into the model, by modifications of the reward function, and episode termination conditions.

References

Muhammed Fatih Balın, Abubakar Abid, and James Zou. Concrete autoencoders: Differentiable feature selection and reconstruction. In *International Conference on Machine Learning*, pages 444–453, 2019.

Lynn A Blewett, Julia A Rivera Drew, Risa Griffin, Miram L King, and Kari Williams. Ipums health surveys: National health interview survey, version 6.2. *Minneapolis: University of Minnesota*, 10:D070, 2016.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Yihong Chen, Bei Chen, Xuguang Duan, Jian-Guang Lou, Yue Wang, Wenwu Zhu, and Yong Cao. Learning-to-ask: Knowledge acquisition via 20 questions. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1216–1225, 2018.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Huang Hu, Xianchao Wu, Bingfeng Luo, Chongyang Tao, Can Xu, Wei Wu, and Zhan Chen. Playing 20 question game with policy-based reinforcement learning. *arXiv preprint arXiv:1808.07645*, 2018.

Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):94, 2018.

Yifeng Li, Chih-Yu Chen, and Wyeth W Wasserman. Deep feature selection: theory and application to identify enhancers and promoters. *Journal of Computational Biology*, 23 (5):322–336, 2016.

Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

Scott M Lundberg, Gabriel G Erion, and Su-In Lee. Consistent individualized feature attribution for tree ensembles. *arXiv preprint arXiv:1802.03888*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al.

Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Debaditya Roy, K Sri Rama Murty, and C Krishna Mohan. Feature selection using deep neural networks. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. Deep supervised feature selection using stochastic gates. *arXiv preprint arXiv:1810.04247*, 2018.

Lei Zhao, Qinghua Hu, and Wenwu Wang. Heterogeneous feature selection with multimodal deep neural networks and sparse group lasso. *IEEE Transactions on Multimedia*, 17(11):1936–1948, 2015.

Tiancheng Zhao and Maxine Eskenazi. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560*, 2016.

Zeyu Zheng, Junhyuk Oh, and Satinder Singh. On learning intrinsic rewards for policy gradient methods. In *Advances in Neural Information Processing Systems*, pages 4644–4654, 2018.

Appendix A. NHIS-NDI 2002-2011 Data Preprocessing Workflow

Step 1

1. We downloaded the publicly available data files for NHIS from the CDC website for years 2002 - 2011. We merged data from 3 separate files for each individual year - 1) sample adult file, 2) family file, and 3) person file. a.https://www.cdc.gov/nchs/nhis/data-questionnaires-documentation.htm
2. We added 3 variables from household files for years 2002-2004 (month of interview [2002, 2003], year of interview [2002, 2003], and region [2004]) since they were in the household files for those years but in the person file for years 2005-2011. We merged the 10 years of complete NHIS data for years 2002-2011.
3. We obtained the variance estimation for the entire 10 year pooled cohort from the Integrated Public Use Microdata Series https://nhis.ipums.org/nhis/ (Blewett et al., 2016).
4. We merged the NHIS data set with Public-use Linked Mortality Files that extract mortality information from the National Death Index https://www.cdc.gov/nchs/data-linkage/mortality-public.htm.
5. The resulting dataset contained $n = 282,001$ observations and $d = 2,360$ variables.

Step 2

1. We dropped the observations without mortality information ($n = 12,905$).
2. We created 3 new variables for defining the outcome – 1) interview quarter (iv-qtr), 2) interval to death (int-death), and 3) death within 4 years (dead-4y).
3. The resulting dataset contained $n = 269,096$ observations and $d = 2,363$ variables.

Step 3

1. Using the data questionnaire documentations for each year, we listed all the variables that had their name changed across the years, keeping the name of their most recent appearance.
2. Kept the variables that were consistent across the years.
3. The resulting dataset contained $n = 269,096$ observations and $d1,022$ variables.

Step 4

1. From the list of all consistent variables, each variable was reviewed by 2 different investigators independently (SM, DM, AA, or CC) and flagged as parent or daughter variable based on the data questionnaires documents and type of question, and kept only the parent variables. Altogether 264 variables were kept.
2. Identified variables with $> 80\%$ missing values not from the NDI variables (like cause of death) and dropped them. 8 variables were dropped.

3. We dropped variables that had their information contained under other variables and household/family identifiers. 45 variables were dropped.

4. The resulting dataset contained $n = 269,096$ observations and $d211$ variables.

Step 5

1. We re-coded variables for analysis using the following guidelines:

    - Categorical variables: We replaced missing values as a separate "999" category for 19 variables. Note: we reduced the number of categories for the following 4 variables by collapsing their values - income ratio (rat-cat2), education (educ1), usual place of care (ausualpl), and family structure (fm-strp).
    - Numeric variables: For numeric variables with values of 97, 98, and 99 (refused, don't know, missing) (9 variables), we created new variables for each of 97, 98, 99 categories (e.g. varx-97), and replaced those values in the original variable as missing (.).

2. Median single-value imputation for missing values for the continuous variables (10 variables).

3. The resulting dataset contained $n = 269,096$ observations and $d = 242$ variables.

Step 6

1. One-hot encoding for categorical interview variables (156 variables) in R. 867 One-hot encoded interview variables were generated.

2.

3. The resulting dataset contained $n = 269,096$ observations, $d = 932$ interview variables, 14 identifiers and 9 outcome variables, out of which we considered the 4 year mortality variable.

Appendix B. Input features

Table 2 shows the set of 50 most important features of the XGB model, trained on the full feature set. In all experiments in this manuscript features were selected from this set.

Appendix C. Examples

Starting new episode with a new test patient

Basic info: sex: 2, age: 85, race:0

Step: 1, Question: la1ar2 , Answer: 0.00

Step: 2, Ready to make a guess: Prob(y=1)=0.874, Guess: y=1, Ground truth: y=1

Episode terminated

Starting new episode with a new test patient

Feature name	Meaning
medicare1	Medicare coverage recode
la1ar1	Any limitation - all persons, all conditions
flwalk0	How difficult to walk 1/4 mile without special equipment
age-p	Age
flclimb0	How difficult to climb 10 steps without special equipment
doinglwp5	What was - - doing last week
la1ar2	Any limitation - all persons, all conditions
flcarry0	How difficult to lift/carry 10 lbs without special equipment
wrklyr12	Work for pay last year
pregnow999	Currently pregnant
smkev1	Ever smoked 100 cigarettes
lupprt1	Lost all upper and lower natural teeth
phstat5	Reported health status
speceq2	Have health problem that requires special equipment
flshop0	How difficult to go out to events without special equipment
flwalk4	How difficult to walk 1/4 mile without special equipment
fliadlyn2	Any family member need help with an IADL
smkev2	Ever smoked 100 cigarettes
educ15	Highest level of school completed
phstat4	Reported health status
eligpwic	Anyone age-eligible for the WIC program
canev1	Ever told by a doctor you had cancer
adnlong21	Time since last saw a dentist
vigfreqw	Freq vigorous activity (times per wk)
sex	Sex
livyr2	Told you had liver condition, past 12 m
private2	Private health insurance recode
ahchyr1	Received home care from health professional, past 12 m
ahcsyr71	Seen/talked to mental health professional, past 12 m
smknow1	Smoke freq: everyday/some days/not at all
origin-i	Hispanic Ethnicity
dibev1	Ever been told that you have diabetes
ephev1	Ever been told you had emphysema
miev1	Ever been told you had a heart attack
kidwkyr2	Told you had weak/failing kidneys, 12 m
phstat1	Reported health status
flsocl0	How difficult to participate in social activities without speci
phstat2	Reported health status
ahchyr2	Received home care from health professional, past 12 m
hiscodi32	Race/ethnicity recode
livyr1	Told you had liver condition, past 12 m
bmi	Body Mass Index (BMI)
amigr2	Had severe headache/migraine, past 3 m
rat-cat24	Ratio of family income to the poverty threshold
jntsymp1	Symptoms of joint pain/aching/stiffness past 30 d
houseown2	Home tenure status
doinglwp1	What was - - doing last week
beddayr	Number of bed days, past 12 months
ahernoy2	times in ER/ED, past 12 m
proxysa2	Sample adult status

Table 2: The pool of 50 most important features of an XGBoost model, out of which the methods selected features.

```
Basic info: sex: 2, age: 24, race:0
Step: 1, Question: la1ar2 , Answer: 1.00
Step: 2, Question: proxysa2 , Answer: 1.00
Step: 3, Ready to make a guess: Prob(y=1)=0.147, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 67, race:1
```

```
Step: 1, Question: la1ar2 , Answer: 1.00
Step: 2, Question: ephev1 , Answer: 0.00
Step: 3, Question: dibev1 , Answer: 0.00
Step: 4, Question: kidwkyr2 , Answer: 1.00
Step: 5, Question: proxysa2 , Answer: 1.00
Step: 6, Ready to make a guess: Prob(y=1)=0.299, Guess: y=0, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 20, race:0
Step: 1, Question: la1ar2 , Answer: 1.00
Step: 2, Question: proxysa2 , Answer: 1.00
Step: 3, Ready to make a guess: Prob(y=1)=0.143, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 48, race:0
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flsocl0 , Answer: 0.00
Step: 5, Question: ahernoy2 , Answer: 3.00
Step: 6, Question: jntsymp1 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.437, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 83, race:1
Step: 1, Ready to make a guess: Prob(y=1)=0.906, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 27, race:1
Step: 1, Ready to make a guess: Prob(y=1)=0.082, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 64, race:1
Step: 1, Question: smkev1 , Answer: 1.00
```

```
Step: 2, Question: kidwkyr2 , Answer: 1.00
Step: 3, Question: phstat1 , Answer: 0.00
Step: 4, Question: phstat2 , Answer: 0.00
Step: 5, Question: proxysa2 , Answer: 0.00
Step: 6, Question: flwalk0 , Answer: 0.00
Step: 7, Ready to make a guess: Prob(y=1)=0.817, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 51, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 1.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: houseown2 , Answer: 0.00
Step: 5, Ready to make a guess: Prob(y=1)=0.099, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 55, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: private2 , Answer: 0.00
Step: 5, Question: livyr2 , Answer: 1.00
Step: 6, Question: flwalk0 , Answer: 0.00
Step: 7, Ready to make a guess: Prob(y=1)=0.282, Guess: y=0, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 38, race:0
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flsocl0 , Answer: 1.00
Step: 5, Question: fliadlyn2 , Answer: 1.00
Step: 6, Question: jntsymp1 , Answer: 0.00
Step: 7, Ready to make a guess: Prob(y=1)=0.257, Guess: y=0, Ground truth: y=0
Episode terminated
```

```
Starting new episode with a new test patient
Basic info: sex: 1, age: 65, race:1
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flshop0 , Answer: 1.00
Step: 5, Question: dibev1 , Answer: 0.00
Step: 6, Question: amigr2 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.737, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 40, race:1
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: eligpwic , Answer: 1.00
Step: 5, Question: adnlong21 , Answer: 1.00
Step: 6, Ready to make a guess: Prob(y=1)=0.125, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 54, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: private2 , Answer: 0.00
Step: 5, Question: livyr2 , Answer: 0.00
Step: 6, Question: flwalk0 , Answer: 0.00
Step: 7, Ready to make a guess: Prob(y=1)=0.468, Guess: y=0, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 45, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 1.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flsocl0 , Answer: 1.00
Step: 5, Question: houseown2 , Answer: 0.00
```

```
Step: 6, Question: flwalk0 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.076, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 77, race:0
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: kidwkyr2 , Answer: 1.00
Step: 3, Question: phstat1 , Answer: 0.00
Step: 4, Question: vigfreqw , Answer: 95.00
Step: 5, Ready to make a guess: Prob(y=1)=0.906, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 80, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: kidwkyr2 , Answer: 1.00
Step: 3, Question: phstat1 , Answer: 0.00
Step: 4, Question: phstat5 , Answer: 0.00
Step: 5, Question: la1ar2 , Answer: 1.00
Step: 6, Question: flwalk0 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.708, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 33, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flwalk4 , Answer: 0.00
Step: 5, Question: flwalk0 , Answer: 1.00
Step: 6, Ready to make a guess: Prob(y=1)=0.114, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 57, race:1
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
```

```
Step: 4, Question: beddayr , Answer: 5.00
Step: 5, Question: dibev1 , Answer: 0.00
Step: 6, Question: amigr2 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.410, Guess: y=0, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 1, age: 42, race:0
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: phstat1 , Answer: 1.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flsocl0 , Answer: 1.00
Step: 5, Question: fliadlyn2 , Answer: 1.00
Step: 6, Question: flwalk0 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.191, Guess: y=0, Ground truth: y=0
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 79, race:1
Step: 1, Question: smkev1 , Answer: 0.00
Step: 2, Question: kidwkyr2 , Answer: 1.00
Step: 3, Question: phstat1 , Answer: 0.00
Step: 4, Question: private2 , Answer: 1.00
Step: 5, Question: la1ar2 , Answer: 1.00
Step: 6, Question: livyr2 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.645, Guess: y=1, Ground truth: y=1
Episode terminated

Starting new episode with a new test patient
Basic info: sex: 2, age: 47, race:1
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: phstat1 , Answer: 0.00
Step: 3, Question: kidwkyr2 , Answer: 1.00
Step: 4, Question: flsocl0 , Answer: 1.00
Step: 5, Question: dibev1 , Answer: 0.00
Step: 6, Question: amigr2 , Answer: 1.00
Step: 7, Ready to make a guess: Prob(y=1)=0.195, Guess: y=0, Ground truth: y=0
Episode terminated
```

```
Starting new episode with a new test patient
Basic info: sex: 1, age: 76, race:1
Step: 1, Question: smkev1 , Answer: 1.00
Step: 2, Question: kidwkyr2 , Answer: 1.00
Step: 3, Question: phstat1 , Answer: 0.00
Step: 4, Question: phstat5 , Answer: 0.00
Step: 5, Ready to make a guess: Prob(y=1)=0.839, Guess: y=1, Ground truth: y=1
Episode terminated
```