AutoPrognosis: Automated Clinical Prognostic Modeling via Bayesian Optimization with Structured Kernel Learning

Ahmed M. Alaa [1] **Mihaela van der Schaar** [1 2 3]

Abstract

Clinical prognostic models derived from large-scale healthcare data can inform critical diagnostic and therapeutic decisions. To enable off-the-shelf usage of machine learning (ML) in prognostic research, we developed AUTOPROGNOSIS: a system for automating the design of predictive modeling *pipelines* tailored for clinical prognosis. AUTOPROGNOSIS optimizes ensembles of pipeline configurations efficiently using a novel batched Bayesian optimization (BO) algorithm that learns a low-dimensional decomposition of the pipelines' high-dimensional hyperparameter space in concurrence with the BO procedure. This is achieved by modeling the pipelines' performances as a black-box function with a Gaussian process prior, and modeling the "similarities" between the pipelines' baseline algorithms via a sparse additive kernel with a Dirichlet prior. *Meta-learning* is used to warmstart BO with external data from "similar" patient cohorts by calibrating the priors using an algorithm that mimics the empirical Bayes method. The system automatically explains its predictions by presenting the clinicians with logical *association rules* that link patients' features to predicted risk strata. We demonstrate the utility of AUTOPROGNOSIS using 10 major patient cohorts representing various aspects of cardiovascular patient care.

1. Introduction

In clinical medicine, *prognosis* refers to the risk of future health outcomes in patients with given features. Prognostic research aims at building actionable predictive models that can inform clinicians about future course of patients' clinical conditions in order to guide screening and therapeutic decisions. With the recent abundance of data linkages, electronic health records, and bio-repositories, clinical researchers have become aware that the value conferred by big, heterogeneous clinical data can only be realized with prognostic models based on flexible machine learning (ML) approaches. There is, however, a concerning gap between the potential and actual utilization of ML in prognostic research; the reason being that clinicians with no expertise in data science find it hard to manually design and tune ML pipelines (Luo et al., 2017).

To fill this gap, we developed AUTOPROGNOSIS, an automated ML (AutoML) framework tailored for clinical prognostic modeling. AUTOPROGNOSIS takes as an input data from a patient cohort, and uses such data to automatically configure ML *pipelines*. Every ML pipeline comprises all stages of prognostic modeling: missing data imputation, feature preprocessing, prediction, and calibration. The system handles different types of clinical data, including longitudinal and survival (time-to-event) data, and automatically *explains* its predictions to the clinicians via an "interpreter" module which outputs clinically interpretable associations between patients' features and predicted risk strata. An overview of the system is provided in Figure 1.

The core component of AUTOPROGNOSIS is an algorithm for configuring ML pipelines using Bayesian optimization (BO) (Snoek et al., 2012). Our BO algorithm models the pipelines' performances as a black-box function, the input to which is a "pipeline configuration", i.e. a selection of algorithms and hyperparameter settings, and the output of which is the performance (predictive accuracy) achieved by such a configuration. We implement BO with a *Gaussian process* (GP) prior on the black-box function. To deal with the high-dimensionality of the pipeline configuration space, we capitalize on the fact that for a given dataset, **the performance of one ML algorithm may not be correlated with that of another algorithm**. For instance, it may be the case that the observed empirical performance of *logistic regression* on a given dataset does not tell us much information about how a *neural network* would perform on the same dataset. In such a case, both algorithms should not share the same GP prior, but should rather be

[1]University of California, Los Angeles, USA [2]University of Oxford, Oxford, UK [3]Alan Turing Institute, London, UK. Correspondence to: Ahmed M. Alaa <ahmedmalaa@ucla.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

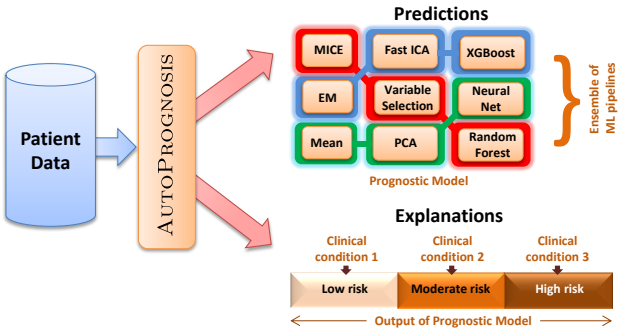

Figure 1. Illustration for exemplary outputs of AUTOPROGNOSIS.

modeled independently. Our BO **learns** such a decomposition of algorithms from data in order to break down the high-dimensional optimization problem into a set of lower-dimensional sub-problems. We model the decomposition of algorithms via an additive kernel with a Dirichlet prior on its structure, and learn the decomposition from data in concurrence with the BO iterations. We also propose a batched (parallelized) version of the BO procedure, along with a computationally efficient algorithm for maximizing the BO acquisition function.

AUTOPROGNOSIS follows a principled Bayesian approach in all of its components. The system implements post-hoc construction of pipeline ensembles via *Bayesian model averaging*, and implements a *meta-learning* algorithm that utilizes data from external cohorts of "similar" patients using an *empirical Bayes* method. In order to resolve the tension between accuracy and interpretability, which is crucial for clinical decision-making (Cabitza et al., 2017), the system presents the clinicians with a rule-based approximation for the learned ML pipeline by mining for logical associations between patients' features and the model's predicted risk strata using a *Bayesian associative classifier* (Agrawal et al., 1993; Kruschke, 2008).

We conclude the paper by conducting a set of experiments on multiple patient cohorts representing various aspects of cardiovascular patient care, and show that prognostic models learned by AUTOPROGNOSIS outperform widely used clinical risk scores and existing AutoML frameworks.

Related work: To the best of our knowledge, none of the existing AutoML frameworks, such as AUTO-WEKA (Kotthoff et al., 2016), AUTO-SKLEARN (Feurer et al., 2015), and TPOT (Olson & Moore, 2016) use principled GP-based BO to configure ML pipelines. All of the existing frameworks model the sparsity of the pipelines' hyperparameter space via frequentist tree-based structures. Both AUTO-WEKA and AUTO-SKLEARN use BO, but through tree-based heuristics, such as random forest models and tree Parzen estimators, whereas TPOT uses a tree-based genetic programming algorithm. Previous works have refrained from using principled GP-based BO because of its statistical and computational complexity in high-dimensional hyperparameter spaces. Our algorithm makes principled, high-dimensional GP-based BO possible by **learning** a sparse additive kernel decomposition for the GP prior. This approach confers many advantages as it captures the uncertainty about the sparsity structure of the GP prior, and allows for principled approaches for (Bayesian) meta-learning and ensemble construction that are organically connected to the BO procedure. In Section 5, we compare the performance of AUTOPROGNOSIS with that of AUTO-WEKA, AUTO-SKLEARN, and TPOT, demonstrating the superiority of our algorithm.

Various previous works have addressed the problem of high-dimensional GP-based BO. (Wang et al., 2013) identifies a low-dimensional effective subspace for the black-box function via random embedding. However, in the AutoML setup, this approach cannot incorporate our prior knowledge about dependencies between the different hyperparameters (we know the sets of hyperparameters that are "activated" upon selecting an algorithm (Hutter et al., 2011)). This prior knowledge was captured by the *Arc-kernel* proposed in (Swersky et al., 2014), and similarly in (Jenatton et al., 2017), where a BO algorithm for domains with tree-structured dependencies was proposed. Unfortunately, both methods require full prior knowledge of the dependencies between the hyperparameters, and hence cannot be used when jointly configuring hyperparameters across multiple algorithms, since the correlations of the performances of different algorithms are not known a priori. (Bergstra et al., 2011) proposed a naïve approach that defines an independent GP for every set of hyperparameters that belong to the same algorithm. Since it does not share any information between the different algorithms, this approach would require trying all combinations of algorithms in a pipeline exhaustively. (In our system, there are 4,800 possible pipelines.) Our model solves the problems above via a **data-driven** kernel decomposition, through which only relevant groups of hyperparameters share a common GP prior, thereby balancing the trade-off between "information sharing" among hyperparameters and statistical efficiency.

2. AUTOPROGNOSIS: A Practical System for Automated Clinical Prognostic Modeling

Consider a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ for a cohort of n patients, with x_i being patient i's features, and y_i being the patient's clinical endpoint. AUTOPROGNOSIS takes $\mathcal{D}$ as an input, and outputs an automatically configured *prognostic model* which predicts the patients' risks, along with "explanations" for the predicted risk strata. This Section provides an overview of the components of AUTOPROGNOSIS;

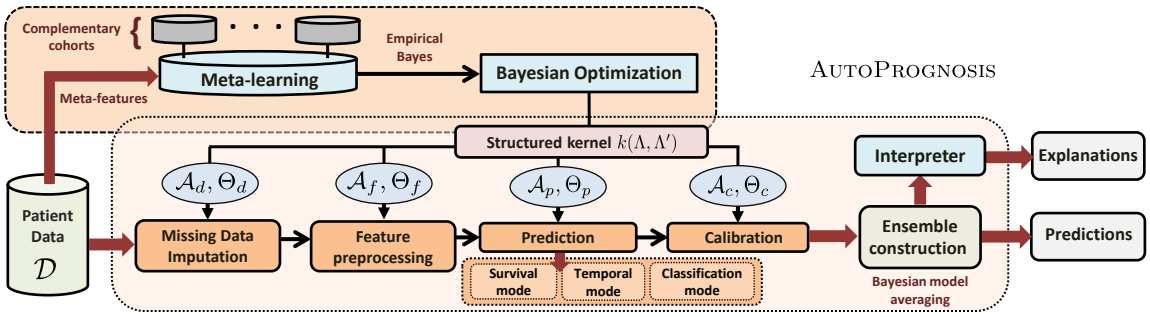

Figure 2. A schematic depiction of AUTOPROGNOSIS. Every ML pipeline comprises imputation, feature processing, prediction, and calibration algorithms. The ensemble construction and interpreter modules are included in the system as post-processing steps.

a schematic depiction of the system is shown in Figure 2.

The core component of AUTOPROGNOSIS is an algorithm that automatically configures ML pipelines, where every pipeline comprises algorithms for missing data imputation (□), feature preprocessing (♣), prediction (•), and calibration (★). Table 1 lists the baseline algorithms adopted by the system in all the stages of a pipeline. The imputation and calibration stages are particularly important for clinical prognostic modeling (Blaha, 2016), and are not supported in existing AutoML frameworks. The total number of hyperparameters in AUTOPROGNOSIS is 106, which is less than those of AUTO-WEKA (786) and AUTO-SKLEARN (110). The pipeline configuration algorithm uses **Bayesian optimization** to estimate the performance of different pipeline configurations in a **scalable** fashion by learning a **structured kernel decomposition** that identifies algorithms with *correlated* performance. Details of the Bayesian optimization algorithm are provided in Sections 3 and 5.

In order to cope with the diverse nature of clinical data and health outcomes, AUTOPROGNOSIS pipelines are enriched with three modes of operation: **(a) classification mode**, **(b) temporal mode**, and **(c) survival mode**. The **classification mode** handles datasets with binary clinical outcomes (Yoon et al., 2017). In this mode, the baseline predictive models include all algorithms in the `scikit-learn` library (Pedregosa et al., 2011), in addition to other powerful algorithms, such as XGBoost (Chen & Guestrin, 2016). The **temporal mode** handles longitudinal and time series data (Alaa et al., 2017) by applying the classification algorithms above on data residing in a sliding window within the time series, which we parametrize by the sequence time (Hripcsak et al., 2015). The **survival mode** handles *time-to-event* data, and involves all the classification algorithms above, in addition to survival models such as Cox proportional hazards model and survival forests (Ishwaran et al., 2008), and models for multiple competing risks (Fine & Gray, 1999).

The **meta-learning** module is a pre-processing step that is used to warmstart BO using data from external cohorts, whereas the **ensemble construction** and **interpreter** modules post-process the BO outputs. All of the three module run with a relatively low computational burden. Details of the three modules are provided in Sections 4 and 5.

3. Pipeline Configuration via Bayesian Optimization with Structured Kernels

Let $(\mathcal{A}_d, \mathcal{A}_f, \mathcal{A}_p, \mathcal{A}_c)$ be the sets of all missing data imputation, feature processing, prediction, and calibration algorithms considered in AUTOPROGNOSIS (Table 1), respectively. A **pipeline** P is a tuple of the form:

$$\boxed{P = (A_d, A_f, A_p, A_c)}$$

where $A_v \in \mathcal{A}_v$, $\forall v \in \{d, f, p, c\}$. The space of all possible pipelines is given by $\mathcal{P} = \mathcal{A}_d \times \mathcal{A}_f \times \mathcal{A}_p \times \mathcal{A}_c$. Thus, a pipeline is a selection of algorithms from the elements of Table 1. An exemplary pipeline can be specified as follows: $P = \{$**MICE**, **PCA**, **Random Forest**, **Sigmoid**$\}$. The total number of pipelines in AUTOPROGNOSIS is $|\mathcal{P}| = 4,800$.

The specification of a **pipeline configuration** is completed by determining the hyperparameters of its constituting algorithms. The space of hyperparameter configurations for a pipeline is $\Theta = \Theta_d \times \Theta_f \times \Theta_p \times \Theta_c$, where $\Theta_v = \cup_a \Theta_v^a$, for $v \in \{d, f, p, c\}$, with Θ_v^a being the space of hyperparameters associated with the a^{th} algorithm in $\mathcal{A}_v$. Thus, a pipeline configuration $P_\theta \in \mathcal{P}_\Theta$ is a selection of algorithms $P \in \mathcal{P}$, and hyperparameter settings $\theta \in \Theta$; $\mathcal{P}_\Theta$ is the space of all possible pipeline configurations.

3.1. The Pipeline Selection & Configuration Problem

The main goal of AUTOPROGNOSIS is to identify the best pipeline configuration $P^*_{\theta^*} \in \mathcal{P}_\Theta$ for a given patient cohort $\mathcal{D}$ via J-fold cross-validation as follows:

$$P^*_{\theta^*} \in \arg\max_{P_\theta \in \mathcal{P}_\Theta} \frac{1}{J} \sum_{i=1}^{J} \mathcal{L}(P_\theta; \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{valid}}), \qquad (1)$$

where $\mathcal{L}$ is a given accuracy metric (AUC-ROC, c-index, etc), $\mathcal{D}^{(i)}_{\text{train}}$ and $\mathcal{D}^{(i)}_{\text{valid}}$ are training and validation splits of $\mathcal{D}$

Pipeline Stage	Algorithms				
□ Data Imputation	□ missForest (2) □ Matrix completion (2)	□ Median (0) □ MICE (1)	□ Most-frequent (0) □ None (0)	□ Mean (0)	□ EM (1)
♣ Feature process.	♣ Feature agglo. (4) ♣ R. kitchen sinks (2)	♣ Kernel PCA (5) ♣ Nystroem (5)	♣ Polynomial (3) ♣ Linear SVM (3)	♣ Fast ICA (4) ♣ Select Rates (3)	♣ PCA (2) ♣ None (0)
• Prediction	• Bernoulli NB (2) • Gaussian NB (0) • Multinomial NB (2) • Ridge Class. (1)	• AdaBoost (4) • XGBoost (5) • R. Forest (5) • Bagging (4)	• Decision Tree (4) • Extr. R. Trees (5) • Neural Net. (5) • k-NN (1)	• Grad. Boost. (6) • Light GBM (5) • Log. Reg. (0) • Surv. Forest (5)	• LDA (4) • L. SVM (4) • GP (3) • Cox Reg. (0)
⋆ Calibration	⋆ Sigmoid (0)	⋆ Isotonic (0)	⋆ None (0)		

Table 1. List of algorithms included in every stage of the pipeline. Numbers in brackets correspond to the number of hyperparameters.

in the i^{th} fold. The optimization problem in (1) is dubbed the *Pipeline Selection and Configuration Problem* (PSCP). The PSCP can be thought of as a generalization for the *combined algorithm selection and hyperparameter optimization* (CASH) problem in (Feurer et al., 2015; Kotthoff et al., 2016), which maximizes an objective with respect to selections of single algorithms from the set $\mathcal{A}_p$, rather than selections of full-fledged pipelines from $\mathcal{P}_\Theta$.

3.2. Solving the PSCP via Bayesian Optimization

The objective in (1) has no analytic form, and hence we treat the PSCP as a *black-box* optimization problem. In particular, we assume that $\frac{1}{J}\sum_{i=1}^{J}\mathcal{L}(P_\theta; \mathcal{D}^{(i)}_{\text{train}}, \mathcal{D}^{(i)}_{\text{valid}})$ is a noisy version of a black-box function $f : \mathbf{\Lambda} \to \mathbb{R}$, were $\mathbf{\Lambda} = \Theta \times \mathcal{P}$, and use BO to search for the pipeline configuration $P^*_{\theta^*}$ that maximizes the black-box function $f(.)$ (Snoek et al., 2012). The BO algorithm specifies a Gaussian process (GP) prior on $f(.)$ as follows:

$$f \sim \mathcal{GP}(\mu(\Lambda), k(\Lambda, \Lambda')), \tag{2}$$

where $\mu(\Lambda)$ is the *mean function*, encoding the expected performance of different pipeline, and $k(\Lambda, \Lambda')$ is the *covariance kernel* (Rasmussen & Williams, 2006), encoding the similarity between the different pipelines.

3.3. Bayesian Optimization via Structured Kernels

The function f is defined over the D-dimensional space $\mathbf{\Lambda}$, where $D = \dim(\mathbf{\Lambda})$ is given by

$$D = \dim(\mathcal{P}) + \sum_{v\in\{d,f,p,c\}}\sum_{a\in\mathcal{A}_v}\dim(\Theta^a_v). \tag{3}$$

In AUTOPROGNOSIS, the domain $\mathbf{\Lambda}$ is high-dimensional, with $D = 106$. (The dimensionality of $\mathbf{\Lambda}$ can be calculated by summing up the number of pipeline stages and the number of hyperparameters in Table 1.) High-dimensionality renders standard GP-based BO infeasible as both the sample complexity of nonparametric estimation and the computational complexity of maximizing the acquisition function are exponential in D (Györfi et al., 2006; Kandasamy et al., 2015). For this reason, existing AutoML frameworks have refrained from using GP priors, and relied instead on scalable tree-based heuristics (Feurer et al., 2015; Kotthoff et al., 2016). Despite its superior performance, recent empirical findings have shown that plain-vanilla GP-based BO is feasible only for problems with $D \leq 10$ (Wang et al., 2013). Thus, the deployment of GP-based BO has been limited to hyperparameter optimization for single, predefined ML models via tools such as `Google`'s `Visier` and `HyperTune` (Golovin et al., 2017). AUTOPROGNOSIS overcomes this challenge by leveraging the structure of the PSCP problem as we show in what follows.

3.3.1. THE STRUCTURE OF THE PSCP PROBLEM

The key idea of our BO algorithm is that for a given dataset, **the performance of a given group of algorithms may not be *informative* of the performance of another group of algorithms**. Since the kernel $k(\Lambda, \Lambda')$ encodes the correlations between the performances of the different pipeline configurations, the underlying "informativeness" structure that relates the different hyperparameters can be expressed via the following **sparse additive kernel decomposition**:

$$k(\Lambda, \Lambda') = \sum_{m=1}^{M} k_m(\Lambda^{(m)}, \Lambda'^{(m)}), \tag{4}$$

where $\Lambda^{(m)} \in \mathbf{\Lambda}^{(m)}, \forall m \in \{1, \ldots, M\}$, with $\{\mathbf{\Lambda}^{(m)}\}_m$ being a set of *disjoint* subspaces of $\mathbf{\Lambda}$. (That is, $\cup_m \Lambda^{(m)} = \mathbf{\Lambda}$, and $\Lambda^{(m)} \cap \Lambda^{(m')} = \emptyset$.) The subspaces are assigned mutually exclusive subsets of the dimensions of $\mathbf{\Lambda}$, so that $\sum_m \dim(\Lambda^{(m)}) = D$. The structure of the kernel in (4) is **unknown** a priori, and needs to be **learned from data**. The kernel decomposition breaks down f as follows:

$$f(\Lambda) = \sum_{m=1}^{M} f_m(\Lambda^{(m)}). \tag{5}$$

The additively sparse structure in (4) gives rise to a statistically efficient BO procedure. That is, if f is γ-smooth, then our additive kernels reduce **sample complexity** from $\boldsymbol{O(n^{\frac{-\gamma}{2\gamma+D}})}$ to $\boldsymbol{O(n^{\frac{-\gamma}{2\gamma+D_m}})}$, where D_m is the maximum number of dimensions in any subspace (Raskutti et al.,

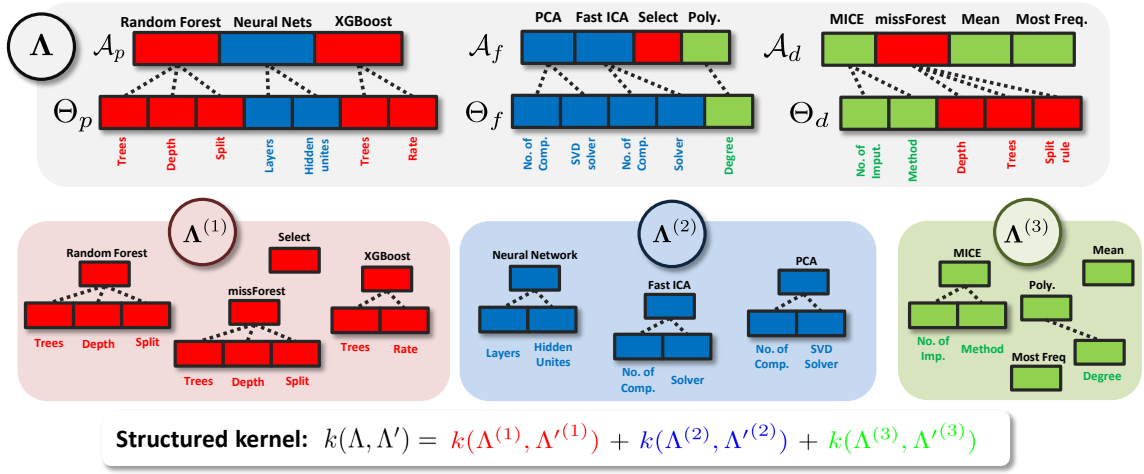

Figure 3. Illustration for a exemplary subspace decomposition $\{\mathbf{\Lambda}^{(m)}\}_{m=1}^{3}$.

2009; Yang et al., 2015). (Similar improvements hold for the cumulative regret (Kandasamy et al., 2015).)

Each subspace $\mathbf{\Lambda}^{(m)} \subset \mathbf{\Lambda}$ contains the hyperparameters of algorithms with correlated performances, whereas algorithms residing in two different subspaces $\mathbf{\Lambda}^{(m)}$ and $\mathbf{\Lambda}^{(m')}$ have uncorrelated performances. Since a hyperparameter in Θ is only *relevant* to $f(.)$ when the corresponding algorithm in $\mathcal{P}$ is selected (Hutter et al., 2009), then the decomposition $\{\mathbf{\Lambda}^{(m)}\}_m$ must ensure that all the hyperparameters of the same algorithm are bundled together in the same subspace. This a priori knowledge about the "conditional relevance" of the dimensions of $\mathbf{\Lambda}$ makes it easier to learn the kernel decomposition from data. Figure 3 provides an illustration for an exemplary subspace decomposition for the hyperparameters of a set of prediction, feature processing and imputation algorithms. Since the structured kernel in (4) is not fully specified a priori, we propose an algorithm to learn it from the data in the next Section.

3.3.2. Structured Kernel Learning

AutoPrognosis uses a Bayesian approach to learn the subspace decomposition $\{\mathbf{\Lambda}^{(m)}\}_m$ in concurrence with the BO procedure, where the following Dirichlet-Multinomial prior is placed on the structured kernel (Wang et al., 2017):

$$\alpha \sim \text{Dirichlet}(M,\gamma),\ \ z_{v,a} \sim \text{Multi}(\alpha), \tag{6}$$

$\forall a \in \mathcal{A}_v, v \in \{d,f,p,c\}$, where $\gamma = \{\gamma_m\}_m$ is the parameter of a Dirichlet prior, $\alpha = \{\alpha_m\}_m$ are the Multinomial mixing proportions, and $z_{v,a}$ is an indicator variable that determines the subspace to which the a^{th} algorithm in $\mathcal{A}_v$ belongs. The kernel decomposition in (4) is learned by updating the posterior distribution of $\{\mathbf{\Lambda}^{(m)}\}_m$ in every iteration of the BO procedure. The posterior distribution over the variables $\{z_{v,a}\}_{v,a}$ and α is given by:

$$\mathbb{P}(z,\alpha \,|\, \mathcal{H}_t,\gamma) \propto \mathbb{P}(\mathcal{H}_t \,|\, z)\,\mathbb{P}(z\,|\alpha)\,\mathbb{P}(\alpha,\gamma), \tag{7}$$

where $z = \{z_{v,a} : \forall a \in \mathcal{A}_v, \forall v \in \{d,f,p,c\}\}$, and $\mathcal{H}_t$ is the history of evaluations of the black-box function up to iteration t. Since the variables $\{z_{v,a}\}_{v,a}$ are sufficient statistics for the subspace decomposition, the posterior over $\{\mathbf{\Lambda}^{(m)}\}_m$ is fully specified by (7) marginalized over α, which can be evaluated using Gibbs sampling as follows:

$$\mathbb{P}(z_{v,a} = m \,|\, z/\{z_{v,a}\}, \mathcal{H}_t) \propto \mathbb{P}(\mathcal{H}_t \,|\, z)\,(|\mathcal{A}_v^{(m)}| + \gamma_m),$$

where $\mathbb{P}(\mathcal{H}_t \,|\, z)$ is the GP likelihood under the kernel induced by z. The Gibbs sampler is implemented via the Gumble-Max trick (Maddison et al., 2014) as follows:

$$\omega_m \overset{\text{i.i.d}}{\sim} \text{Gumbel}(0,1),\ m \in \{1,\ldots,M\}, \tag{8}$$
$$z_{v,a} \sim \arg\max_m \mathbb{P}(\mathcal{H}_t \,|\, z, z_{v,a} = m)(|\mathcal{A}_v^{(m)}| + \gamma_m) + \omega_m.$$

3.3.3. Exploration via Diverse Batch Selection

The BO procedure solves the PSCP problem by exploring the performances of a sequence of pipelines $\{P_{\theta^1}^1, P_{\theta^2}^2, \ldots\}$ until it (hopefully) converges to the optimal pipeline $P_{\theta^*}^*$. In every iteration t, BO picks a pipeline to evaluate using an *acquisition function* $A(P_\theta; \mathcal{H}_t)$ that balances between *exploration* and *exploitation*. AutoPrognosis deploys a 2-step batched (parallelized) exploration scheme that picks B pipelines for evaluation at every iteration t as follows:

Step 1: Select the frequentist kernel decomposition $\{\hat{\mathbf{\Lambda}}^{(m)}\}_m$ that maximizes the posterior $\mathbb{P}(z \,|\, \mathcal{H}_t)$.

Step 2: Select the B pipelines $\{P_\theta^b\}_{b=1}^B$ with the highest values for the acquisition function $\{A(P_\theta^b; \mathcal{H}_t)\}_{b=1}^B$, such that each pipeline P_θ^b, $b \in \{1,\ldots,B\}$, involves a **distinct prediction** algorithm from a **distinct subspace** in $\{\hat{\mathbf{\Lambda}}^{(m)}\}_m$.

We use the well-known *Upper Confidence Bound* (UCB) as acquisition function (Snoek et al., 2012). The decomposition in (5) offers an **exponential speed up** in the overall **computational complexity** of Step 2 since the UCB acquisition function is maximized separately for every (low-dimensional) component f_m; this reduces the number of computations from to $\mathcal{O}(n^{-D})$ to $\mathcal{O}(n^{-D_m})$. The batched implementation is advantageous since sequential evaluations of $f(.)$ are time consuming as it involves training the selected ML algorithms.

Step 2 in the algorithm above encourages exploration as follows. In every iteration t, we select a "diverse" batch of pipelines for which every pipeline is representative of a **distinct** subspace in $\{\hat{\mathbf{\Lambda}}^{(m)}\}_m$. The batch selection scheme above encourages diverse exploration without the need for sampling pipelines via a determinantal point process with an exponential complexity as in (Kathuria et al., 2016; Nikolov, 2015; Wang et al., 2017). We also devise an efficient **backward induction** algorithm that exploits the structure of a pipeline to maximize the acquisition function efficiently. (Details are provided in the supplement.)

4. Ensemble Construction & Meta-learning

In this Section, we discuss the details of the ensemble Construction and meta-learning modules; details of the interpreter module are provided in the next Section.

4.1. Post-hoc Ensemble Construction

The **frequentist** approach to pipeline configuration is to pick the pipeline with the best observed performance from the set $\{P^1_{\theta^1}, \ldots, P^t_{\theta^t}\}$ explored by the BO algorithm in Section 3.3.3. However, such an approach does not capture the uncertainty in the pipelines' performances, and wastefully throws away $t-1$ of the evaluated pipelines. On the contrary, AUTOPROGNOSIS makes use of all such pipelines via post-hoc **Bayesian model averaging**, where it creates an ensemble of weighted pipelines $\sum_i w_i P^i_{\theta^i}$. Model averaging is particularly useful in cohorts with small sample sizes, where large uncertainty about the pipelines' performances would render frequentist solutions unreliable.

The ensemble weight $w_i = \mathbb{P}(P^{i^*}_{\theta^{i^*}} = P^i_{\theta^i} \mid \mathcal{H}_t)$ is the posterior probability of P^i_θ being the best performing pipeline:

$$w_i = \sum_z \mathbb{P}(P^{i^*}_{\theta^{i^*}} = P^i_{\theta^i} \mid z, \mathcal{H}_t) \cdot \mathbb{P}(z \mid \mathcal{H}_t), \tag{9}$$

where i^* is the pipeline configuration with the best (true) generalization performance. The weights in (9) are computed by Monte Carlo sampling of kernel decompositions via the posterior $\mathbb{P}(z \mid \mathcal{H}_t)$, and then sampling the pipelines' performances from the posterior $f \mid z, \mathcal{H}_t$. Note that, unlike the ensemble builder of AUTOSKLEARN (Feurer et al., 2015), the weights in (9) account for correlations between different pipelines, and hence it penalizes combinations of "similar" pipelines even if they are performing well. Moreover, our post-hoc approach allows building ensembles without requiring extra hyperparameters: in AUTOWEKA, ensemble construction requires a 5-fold increase in the number of hyperparameters (Kotthoff et al., 2016).

4.2. Meta-learning via Empirical Bayes

The Bayesian model used for solving the PSCP problem in Section 3 can be summarized as follows:

$$f \sim \mathcal{GP}(\mu, k \mid z), \; z \sim \text{Multi}(\alpha), \; \alpha \sim \text{Dirichlet}(M, \gamma).$$

The speed of convergence of BO depends on the calibration of the prior's hyperparameters (M, γ, μ, k). An agnostic prior would require many iterations to converge to satisfactory pipeline configurations. To warmstart the BO procedure for a new cohort $\mathcal{D}$, we incorporate prior information obtained from previous runs of AUTOPROGNOSIS on a repository of K complementary cohorts $\{\mathcal{D}_1, \ldots, \mathcal{D}_K\}$. Our meta-learning approach combines $\{\mathcal{H}^1_{t_1}, \ldots, \mathcal{H}^M_{t_K}\}$ (optimizer runs on the K complementary cohorts) with the data in $\mathcal{D}$ to obtain an empirical Bayes estimate $(\hat{M}, \hat{\gamma}, \hat{\mu}, \hat{k})$.

Our approach to meta-learning works as follows. For every complementary dataset $\mathcal{D}_k$, we create a set of 55 **meta-features** $\mathcal{M}(\mathcal{D}_k)$, 40 of which are **statistical meta-features** (e.g. number of features, size of data, class imbalance, etc), and the remaining 15 are **clinical meta-features** (e.g. lab tests, vital signs, ICD-10 codes, diagnoses, etc). For every complementary dataset in $\mathcal{D}_j$, we optimize the hyperparameters $(\hat{M}_j, \hat{\gamma}_j, \hat{\mu}_j, \hat{k}_j)$ via marginal likelihood maximization. For a new cohort $\mathcal{D}$, we compute a set of weights $\{\eta_j\}_j$, with $\eta_j = \ell_j / \sum_k \ell_k$, where $\ell_j = \|\mathcal{M}(\mathcal{D}) - \mathcal{M}(\mathcal{D}_j)\|_1$, and calibrate its prior (M, γ, μ, k) by setting it to be the average of the estimates $(\hat{M}_j, \hat{\gamma}_j, \hat{\mu}_j, \hat{k}_j)$, weighted by $\{\eta_j\}_j$.

Existing methods for meta-learning focus only on identifying well-performing pipelines from other datasets, and use them for initializing the optimization procedure (Brazdil et al., 2008; Feurer et al., 2015). Conceptualizing meta-learning as an empirical Bayes calibration procedure allows the transfer of a much richer set of information across datasets. Through the method described above, AUTOPROGNOSIS can import information on the smoothness of the black-box function (k), the similarities among baseline algorithms (γ, M), and the expected pipelines' performances (μ). This improves not only the initialization of the BO procedure, but also the mechanism by which it explores the pipelines' design space.

5. Evaluation of AUTOPROGNOSIS

In this section, we assess the ability of AUTOPROGNOSIS to **automatically** make the right prognostic **modeling choices**

	MAGGIC	UK Biobank	UNOS-I	UNOS-II	SEER-I	SEER-II	SEER-III	SEER-IV	SEER-V	SEER-VI
AUTOPROGNOSIS										
vanilla	0.76 ± .004	0.71 ± .004	0.78 ± .002	0.65 ± .001	0.68 ± .002	0.66 ± .005	0.61 ± .001	0.69 ± .002	0.64 ± .002	0.65 ± .003
best predictor	Grad. Boost	XGBoost	AdaBoost	Rand. Forest	Cox PH	Cox PH	S. Forest	Cox PH	S. Forest	Cox PH
+ ensembles	0.77 ± .002	0.73 ± .003	0.80 ± .001	**0.66 ± .001**	0.68 ± .002	0.67 ± .003	0.62 ± .001	0.69 ± .002	0.66 ± .002	0.65 ± .002
+ meta-learning	0.77 ± .004	0.72 ± .004	0.79 ± .002	0.65 ± .002	0.72 ± .003	0.68 ± .003	0.64 ± .001	0.71 ± .003	0.69 ± .003	0.66 ± .002
full-fledged	**0.78 ± .004**	**0.74 ± .003**	**0.81 ± .001**	**0.66 ± .001**	**0.73 ± .003**	**0.69 ± .003**	**0.64 ± .001**	**0.72 ± .002**	**0.70 ± .003**	**0.67 ± .002**
AUTO-SKLEARN	0.76 ± .003	0.72 ± .004	0.77 ± .002	0.63 ± .002	0.67 ± .002	0.51 ± .005	0.60 ± .001	0.65 ± .004	0.64 ± .002	0.61 ± .003
AUTO-WEKA	0.75 ± .003	0.72 ± .005	0.78 ± .001	0.62 ± .002	0.66 ± .002	0.54 ± .004	0.59 ± .002	0.68 ± .003	0.63 ± .004	0.63 ± .002
TPOT	0.74 ± .006	0.68 ± .005	0.72 ± .003	0.61 ± .003	0.64 ± .003	0.59 ± .003	0.57 ± .002	0.67 ± .004	0.62 ± .005	0.61 ± .003
Clinical Score	0.70 ± .007	0.70 ± .003	0.62 ± .001	0.56 ± .001	—	—	—	—	—	—
Cox PH	0.75 ± .005	0.71 ± 0.002	0.70 ± .001	0.59 ± .001	0.71 ± .003	0.65 ± .004	0.62 ± .002	**0.72 ± .003**	0.68 ± .003	**0.67 ± .002**

Table 2. Performance of the different prognostic models in terms of the AUC-ROC with 5-fold cross-validation. Bold numbers correspond to the best result. The "best predictor" row lists the prediction algorithms picked by vanilla AUTOPROGNOSIS.

when confronted with a variety of clinical datasets with different **meta-features**.

5.1. Cardiovascular Disease Cohorts

We conducted experiments on 10 cardiovascular cohorts that correspond to the following aspects of patient care:

• **Preventive care:** We considered two major cohorts for preventive cardiology. The first is the Meta-analysis Global Group in Chronic heart failure database (**MAGGIC**), which holds data for 46,817 patients gathered from multiple clinical studies (Wong et al., 2014). The second cohort is the **UK Biobank**, which is a bio-repository with data for more than 500,000 volunteers in the UK (Sudlow et al., 2015).

• **Heart transplant wait-list management:** We extracted data from the United Network for Organ Sharing (UNOS) database, which holds information on all heart transplants conducted in the US between the years 1985 to 2015. Cohort **UNOS-I** is a pre-transplant population of 36,329 cardiac patients who were enrolled in a transplant wait-list.

• **Post-transplant follow-up:** Cohort **UNOS-II** is a post-transplant population of 60,400 patients in the US who underwent a transplant between the years 1985 to 2015.

• **Cardiovascular comorbidities:** We extracted 6 cohorts from the Surveillance, Epidemiology, and End Results (SEER) cancer registries, which cover approximately 28% of the US population (Yoo & Coughlin, 2018). We predict cardiac deaths in patients diagnosed with breast cancer (**SEER-I**), colorectal cancer (**SEER-II**), Leukemia (**SEER-III**), respiratory cancers (**SEER-IV**), digestive system cancer (**SEER-V**), and urinary system cancer (**SEER-VI**).

The first three groups of datasets (colored in red) were collected for cohorts of patients diagnosed with (or at risk for) cardiac diseases, and so they shared a set of meta-features, including a large number of cardiac risk factors, low censoring rate, and moderate class imbalance. The last group of datasets (colored in blue) was collected for cohorts of cancer patients for whom cardiac diseases are potential comorbidities. These datasets shared a different set of meta-features, including a small number of cardiac risk factors, high censoring rate, and severe class imbalance. Our experiments will demonstrate the ability of AUTOPROGNOSIS to adapt its modeling choices to these different clinical setups.

5.2. Performance of AUTOPROGNOSIS

Table 2 shows the performance of various competing prognostic modeling approaches evaluated in terms of the area under receiver operating characteristic curve (AUC-ROC) with 5-fold cross-validation[1]. We compared the performance of AUTOPROGNOSIS with the clinical risk scores used for predicting prognosis in each cohort (MAGGIC score in **MAGGIC** and **UNOS-I** (Wong et al., 2014), Framingham score in the **UK Biobank** (Schnabel et al., 2009), and IMPACT score in **UNOS-II** (Weiss et al., 2011)). We also compared with various AutoML frameworks, including AUTO-WEKA (Kotthoff et al., 2016), AUTO-SKLEARN (Feurer et al., 2015), and TPOT (Olson & Moore, 2016). Finally, we compared with a standard Cox proportional hazards (Cox PH) model, which is the model most commonly used in clinical prognostic research.

Table 2 demonstrates the superiority of AUTOPROGNOSIS to all the competing models on all the cohorts under consideration. This reflects the robustness of our system since the 10 cohorts had very different characteristics. In many experiments, the learned kernel decomposition reflected an intuitive clustering of algorithms by the similarity of their structure. For instance, Figure 4 shows one subspace in the frequentist decomposition learned by AUTOPROGNOSIS over the BO iterations for the **MAGGIC** cohorts. We can

[1]All algorithms were allowed to run for a maximum of 10 hours to ensure a fair comparison.

see that all ensemble methods in the imputation and prediction stages that use decision-trees as their base learners were lumped together in the same subspace.

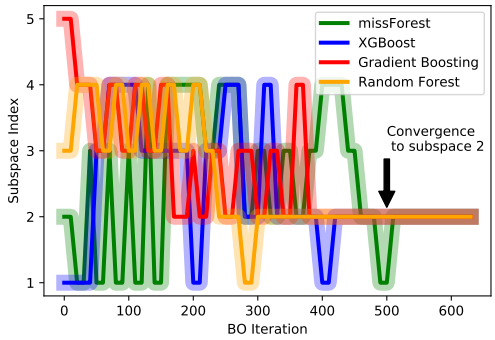

Figure 4. The learned kernel decomposition for **MAGGIC**.

5.3. The "Interpreter"

Albeit accurate, models built by AUTOPROGNOSIS would generally be hard for a clinician to "interpret". To address this issue, AUTOPROGNOSIS deploys an **interpreter** module (see Figure 2) that takes as an input the learned model for a given cohort, in addition to a set of actionable risk strata $\mathcal{R}$, and outputs an "explanation" for its predictions in terms of a set of logical *association rules* of the form:

$$C_1 \wedge C_2 \wedge \ldots \wedge C_{l(r)} \implies r, \forall r \in \mathcal{R}, \tag{10}$$

where $\{C_1, \ldots, C_{l(r)}\}$ is a set of Boolean conditions associated with risk stratum r. The association rules are obtained via a *Bayesian associative classifier* (Ma & Liu, 1998; Agrawal et al., 1993; Kruschke, 2008; Luo, 2016), with a prior over association rules, and a posterior computed based on target labels that correspond to the outputs of the learned model discretized via the strata in $\mathcal{R}$. The Bayesian approach allows incorporating prior knowledge (from clinical literature) about "likely" association rules.

We report one example for an explanation provided by the interpreter module based on our experiments on the **UK Biobank** cohort. For this cohort, the standard Framingham risk score exhibited an AUC-ROC of 0.705 for the overall cohort, but its AUC-ROC for patients with Type-2 Diabetes (T2D) was as low as 0.63. On the contrary, AUTOPROGNOSIS performed almost equally well in the two subgroups. The interpreter provided an explanation for the improved predictions through the following association rule:

Diabetic $\wedge$ **Lipid-lowering** $\wedge$ **(Age** $\geq$ **40)** $\implies$ **High risk**

None of these risk factors were included in the standard guidelines. That is, the interpreter indicates that a better stratification, with new risk factors such the usage of lipid-lowering drugs, is possible for diabetic patients. Clinicians can use the interpreter as a data-driven hypothesis generator that prompts new risk factors and strata for subsequent research.

5.4. Learning to Pick the Right Model and AUTOPROGNOSIS as a Clairvoyant

We split up Table 2 into 2 groups of columns: group 1 (left) contains cohorts obtained from cardiology studies, whereas group 2 (right) contains cohorts obtained from cancer studies, with cardiac secondary outcomes. As mentioned earlier, the two groups had different meta-features. We tracked the modeling choices made by vanilla AUTOPROGNOSIS (no ensembles or meta-learning) in both groups ("best predictor" row in Table 2). For all datasets in group 2, AUTOPROGNOSIS decided that survival modeling (using Cox PH model or survival forests) is the right model. This is because, with the high prevalence of censored time-to-event data, survival models are more data-efficient than operating on binarized survival labels and removing patients lost to follow-up. When given richer datasets with a large number of relevant features, low rates of censoring and moderate imbalance (group 1), AUTOPROGNOSIS spent more iterations navigating ML classifiers, and learned that an algorithm like AdaBoost is a better choice for a dataset like **UNOS-I**. Such a (non-intuitive) choice would have not been possibly identified by a clinical researcher; researchers typically use the Cox PH model, which on the **UNOS-I** cohort provides an inferior performance.

Meta-learning was implemented via leave-one-dataset-out validation: we run vanilla AUTOPROGNOSIS on all of the 10 cohorts, and then for every cohort, we use the other 9 cohorts as the complementary datasets used to implement the meta-learning algorithm. Since the pool of complementary cohorts contained 5 datasets for cardiovascular comorbidities, meta-learning was most useful for group 2 datasets as they all had very similar meta-features. With meta-learning, AUTOPROGNOSIS had a strong prior on survival models for group 2 datasets, and hence it converges quickly to a decision on using a survival model having observed the dataset's meta-features. Ensemble construction was most useful for the **MAGGIC** and **UNOS** cohorts, since those datasets had more complex hypotheses to learn.

Clinical researchers often ask the question: **when should I use machine learning for my prognostic study?** The answer depends on the nature of the dataset involved. As we have see in Table 2, a simple Cox model may in some cases be sufficient to issue accurate predictions. The meta-learning module in AUTOPROGNOSIS can act as a clairvoyant that tells whether ML models would add value to a given prognostic study **without even training any model**. That is, by looking at the "meta-learned" GP prior calibrated by a new dataset's meta-features, we can see

whether the prior assigns high scores to ML models compared to a simple Cox model, and hence decide on whether ML has gains to offer for such a dataset.

References

Agrawal, Rakesh, Imieliński, Tomasz, and Swami, Arun. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pp. 207–216. ACM, 1993.

Alaa, Ahmed M, Hu, Scott, and van der Schaar, Mihaela. Learning from clinical judgments: Semi-markov-modulated marked hawkes processes for risk prognosis. *International Conference on Machine Learning*, 2017.

Bergstra, James, Bardenet, Rémi, Kégl, B, and Bengio, Y. Implementations of algorithms for hyper-parameter optimization. In *NIPS Workshop on Bayesian optimization*, pp. 29, 2011.

Blaha, Michael J. The critical importance of risk score calibration, 2016.

Brazdil, Pavel, Carrier, Christophe Giraud, Soares, Carlos, and Vilalta, Ricardo. *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

Cabitza, Federico, Rasoini, Raffaele, and Gensini, Gian Franco. Unintended consequences of machine learning in medicine. *Jama*, 318(6):517–518, 2017.

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

Feurer, Matthias, Klein, Aaron, Eggensperger, Katharina, Springenberg, Jost, Blum, Manuel, and Hutter, Frank. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2962–2970, 2015.

Fine, Jason P and Gray, Robert J. A proportional hazards model for the subdistribution of a competing risk. *Journal of the American statistical association*, 94(446):496–509, 1999.

Golovin, Daniel, Solnik, Benjamin, Moitra, Subhodeep, Kochanski, Greg, Karro, John, and Sculley, D. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1487–1495. ACM, 2017.

Györfi, László, Kohler, Michael, Krzyzak, Adam, and Walk, Harro. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

Hripcsak, George, Albers, David J, and Perotte, Adler. Parameterizing time in electronic health record studies. *Journal of the American Medical Informatics Association*, 22(4):794–804, 2015.

Hutter, Frank, Hoos, Holger H, Leyton-Brown, Kevin, and Stützle, Thomas. Paramils: an automatic algorithm configuration framework. *Journal of Artificial Intelligence Research*, 36(1):267–306, 2009.

Hutter, Frank, Hoos, Holger H, and Leyton-Brown, Kevin. Sequential model-based optimization for general algorithm configuration. *LION*, 5:507–523, 2011.

Ishwaran, Hemant, Kogalur, Udaya B, Blackstone, Eugene H, and Lauer, Michael S. Random survival forests. *The annals of applied statistics*, pp. 841–860, 2008.

Jenatton, Rodolphe, Archambeau, Cedric, González, Javier, and Seeger, Matthias. Bayesian optimization with tree-structured dependencies. In *International Conference on Machine Learning*, pp. 1655–1664, 2017.

Kandasamy, Kirthevasan, Schneider, Jeff, and Póczos, Barnabás. High dimensional bayesian optimisation and bandits via additive models. In *International Conference on Machine Learning (ICML)*, pp. 295–304, 2015.

Kathuria, Tarun, Deshpande, Amit, and Kohli, Pushmeet. Batched gaussian process bandit optimization via determinantal point processes. In *Advances in Neural Information Processing Systems*, pp. 4206–4214, 2016.

Kotthoff, Lars, Thornton, Chris, Hoos, Holger H, Hutter, Frank, and Leyton-Brown, Kevin. Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka. *Journal of Machine Learning Research*, 17: 1–5, 2016.

Kruschke, John K. Bayesian approaches to associative learning: From passive to active learning. *Learning & behavior*, 36(3):210–226, 2008.

Luo, Gang. Automatically explaining machine learning prediction results: a demonstration on type 2 diabetes risk prediction. *Health information science and systems*, 4(1):2, 2016.

Luo, Gang, Stone, Bryan L, Johnson, Michael D, Tarczy-Hornoch, Peter, Wilcox, Adam B, Mooney, Sean D, Sheng, Xiaoming, Haug, Peter J, and Nkoy, Flory L. Automating construction of machine learning models with clinical big data: proposal rationale and methods. *JMIR research protocols*, 6(8), 2017.

Ma, Bing Liu Wynne Hsu Yiming and Liu, Bing. Integrating classification and association rule mining. In *Proceedings of the fourth international conference on knowledge discovery and data mining*, 1998.

Maddison, Chris J, Tarlow, Daniel, and Minka, Tom. A* sampling. In *Advances in Neural Information Processing Systems*, pp. 3086–3094, 2014.

Nikolov, Aleksandar. Randomized rounding for the largest simplex problem. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pp. 861–870. ACM, 2015.

Olson, Randal S and Moore, Jason H. Tpot: A tree-based pipeline optimization tool for automating machine learning. In *ICML Workshop on Automatic Machine Learning*, pp. 66–74, 2016.

Pedregosa, Fabian, Varoquaux, Gaël, Gramfort, Alexandre, Michel, Vincent, Thirion, Bertrand, Grisel, Olivier, Blondel, Mathieu, Prettenhofer, Peter, Weiss, Ron, Dubourg, Vincent, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

Raskutti, Garvesh, Yu, Bin, and Wainwright, Martin J. Lower bounds on minimax rates for nonparametric regression with additive sparsity and smoothness. In *Advances in Neural Information Processing Systems*, pp. 1563–1570, 2009.

Rasmussen, Carl Edward and Williams, Christopher KI. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Schnabel, Renate B, Sullivan, Lisa M, Levy, Daniel, Pencina, Michael J, Massaro, Joseph M, D'Agostino, Ralph B, Newton-Cheh, Christopher, Yamamoto, Jennifer F, Magnani, Jared W, Tadros, Thomas M, et al. Development of a risk score for atrial fibrillation (framingham heart study): a community-based cohort study. *The Lancet*, 373(9665):739–745, 2009.

Snoek, Jasper, Larochelle, Hugo, and Adams, Ryan P. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems (NIPS)*, pp. 2951–2959, 2012.

Sudlow, Cathie, Gallacher, John, Allen, Naomi, Beral, Valerie, Burton, Paul, Danesh, John, Downey, Paul, Elliott, Paul, Green, Jane, Landray, Martin, et al. Uk biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS medicine*, 12(3):e1001779, 2015.

Swersky, Kevin, Duvenaud, David, Snoek, Jasper, Hutter, Frank, and Osborne, Michael A. Raiders of the lost architecture: Kernels for bayesian optimization in conditional parameter spaces. *arXiv preprint arXiv:1409.4011*, 2014.

Wang, Zi, Li, Chengtao, Jegelka, Stefanie, and Kohli, Pushmeet. Batched high-dimensional bayesian optimization via structural kernel learning. *International Conference on Machine Learning (ICML)*, 2017.

Wang, Ziyu, Zoghi, Masrour, Hutter, Frank, Matheson, David, De Freitas, Nando, et al. Bayesian optimization in high dimensions via random embeddings. In *IJCAI*, pp. 1778–1784, 2013.

Weiss, Eric S, Allen, Jeremiah G, Arnaoutakis, George J, George, Timothy J, Russell, Stuart D, Shah, Ashish S, and Conte, John V. Creation of a quantitative recipient risk index for mortality prediction after cardiac transplantation (impact). *The Annals of thoracic surgery*, 92 (3):914–922, 2011.

Wong, Chih M, Hawkins, Nathaniel M, Petrie, Mark C, Jhund, Pardeep S, Gardner, Roy S, Ariti, Cono A, Poppe, Katrina K, Earle, Nikki, Whalley, Gillian A, Squire, Iain B, et al. Heart failure in younger patients: the meta-analysis global group in chronic heart failure (maggic). *European heart journal*, 35(39):2714–2721, 2014.

Yang, Yun, Tokdar, Surya T, et al. Minimax-optimal nonparametric regression in high dimensions. *The Annals of Statistics*, 43(2):652–674, 2015.

Yoo, Wonsuk and Coughlin, Steven S. Surveillance, epidemiology, and end results (seer) data for monitoring cancer trends. *Journal of the Georgia Public Health Association*, 2018.

Yoon, Jinsung, Alaa, Ahmed M, Cadeiras, Martin, and van der Schaar, Mihaela. Personalized donor-recipient matching for organ transplantation. In *AAAI*, pp. 1647–1654, 2017.

6. Supplementary Material

6.1. Clinical Endpoints Considered in Experiments

- **MAGGIC**: heart failure deaths within 3 years from the baseline.
- **UK Biobank**: onset of myocardial infraction or ischemic heart disease within 10 years of the baseline.
- **UNOS-I**: 3-year mortality.
- **UNOS-II**: 3-year mortality.
- **SEER**: 10-year cardiac-related mortality.

6.2. Acquisition Function Optimization using Backward Induction

The BO dynamics rely on optimizing the acquisition function as follows:

$$P_\theta^{t+1} = \arg\max_{P_\theta \in \mathcal{P}_\Theta} A(P_\theta; \mathcal{H}_t)..$$

Because of the kernel decomposition, the acquisition function can be decomposed into M components as follows:

$$P_\theta^{t+1} = \arg\max_{P_\theta \in \mathcal{P}_\Theta} \sum_m A_m(P_\theta; \mathcal{H}_t)..$$

Note that $\mathcal{P}_\Theta$ is a strict subset of $\mathbf{\Lambda}$, i.e. $\mathcal{P}_\Theta \subset \mathbf{\Lambda}$. That is, we cannot directly maximize $\sum_m A_m(P_\theta; \mathcal{H}_t)$ numerically without ensuring that the solution P_θ^{t+1} is a valid pipeline. To ensure that P_θ^{t+1} is a valid pipeline, we optimize $\sum_m A_m(P_\theta; \mathcal{H}_t)$ sequentially by selecting the algorithms at the different stages of the pipeline backwards as follows:

$$\begin{aligned}
A_{a*}^{t+1} &= \arg\max_{A_a \in \mathcal{A}_a} \sum_m A_m(A_a; \mathcal{H}_t) \\
A_{p*}^{t+1} &= \arg\max_{A_p \in \mathcal{A}_p} \sum_m A_m(A_p; \mathcal{H}_t \mid A_{a*}^{t+1}) \\
A_{f*}^{t+1} &= \arg\max_{A_f \in \mathcal{A}_f} \sum_m A_m(A_f; \mathcal{H}_t \mid A_{p*}^{t+1}, A_{a*}^{t+1}) \\
A_{d*}^{t+1} &= \arg\max_{A_d \in \mathcal{A}_d} \sum_m A_m(A_d; \mathcal{H}_t \mid A_{p*}^{t+1}, A_{a*}^{t+1}, A_{f*}^{t+1}).
\end{aligned}$$

Note that every optimization step above operates on the dimensions associated with only 1 stage of a pipeline, and hence is exponentially faster than the original full objective.

6.3. Meta-features

The system uses a set of 38 statistical meta-features and 15 clinical meta-features, all listed in Table 3.

Statistical meta-features	Statistical meta-features	Statistical meta-features	Clinical meta-features	Clinical meta-features
Entropy	Class imbalance	No. of data points	Lab tests indicator	Vital Signs indicator
No. of features	No. features with missing values	No. of missing values	Longitudinal data indicator	Survival data indicator
No. of binary features	No. of numeric features	No. of categorical features	Hematological tests	ICD-10 codes
Percentage of features with missing values	Percentage of missing values	Missing rate per class	ICD-9 codes	Number of repeated measures
Skewness-max	Skewness-mean	Skewness-min	Number of ICD-10 codes	Entropy of ICD-10 codes
Skewness-std	kurtosis-max	kurtosis-mean	Clinical notes	Treatment prescriptions
kurtosis-min	kurtosis-std	PCA number of components	Imaging data indicators	Diagnosis indicators

Table 3. List of all meta-features in AUTOPROGNOSIS.