From Predictions to Decisions: Using Lookahead Regularization

Nir Rosenfeld
School of Enginreering and Applied Science
Harvard University
nirr@g.harvard.edu

Sophie Hilgard
School of Enginreering and Applied Science
Harvard University
ash798@g.harvard.edu

Sai Srivatsa Ravindranath
School of Enginreering and Applied Science
Harvard University
saisr@g.harvard.edu

David C. Parkes
School of Enginreering and Applied Science
Harvard University
parkes@eecs.harvard.edu

Abstract

Machine learning is a powerful tool for predicting human-related outcomes, from credit scores to heart attack risks. But when deployed, learned models also affect how users act in order to improve outcomes, whether predicted or real. The standard approach to learning is agnostic to induced user actions and provides no guarantees as to the effect of actions. We provide a framework for learning predictors that are both accurate and promote good actions. For this, we introduce *look-ahead regularization* which, by anticipating user actions, encourages predictive models to also induce actions that improve outcomes. This regularization carefully tailors the uncertainty estimates governing confidence in this improvement to the distribution of model-induced actions. We report the results of experiments on real and synthetic data that show the effectiveness of this approach.

1 Introduction

Machine learning is increasingly being used in domains that have considerable impact on people, ranging from healthcare [7] to banking [43] to manufacturing [50]. Moreover, in many of these domains, the desire for transparency has led to published machine-learned models that play a dual role in prediction and influencing behavior change. Consider a doctor who uses a risk tool to predict whether a patient is in danger of having a heart attack, while at the same time wanting to recommend lifestyle changes to improve outcomes.[1] Consider a bank, looking to predict whether customers are likely to repay loans while customers are, at the same time, seeking to improve their credit scores. Consider a wine producer, looking to predict the demand for a new vintage while at the same time deciding how to make changes to their production process to improve future vintages.

It is well understood that correlation and causation need not go hand-in-hand [31, 37]. What is novel about this work is that we seek models that serve the dual purpose of achieving predictive accuracy as well as providing high confidence that decisions made with respect to the model improve outcomes. That is, we care about the utility that comes from having a predictive tool, while recognizing that these tools may also drive decisions. To illustrate the potential pitfalls of a purely predictive approach, consider a doctor who would like to advise a patient on how to reduce risk of heart attack. If the doctor assesses risk using a linear predictive model (as is often the case, see [49]), then a negative

[1] MDCalc is one example of a site that provides risk assessment calculators for use by medical professionals https://www.mdcalc.com/

Preprint. Under review.

coefficient for alcohol consumption may lead the doctor to suggest a daily glass of red wine. Is this decision justified? Perhaps not, although this recommendation has often been made based on correlative evidence and despite a clear lack of experimental support [17, 38].

At the same time, predictive models are valuable in and of themselves, for example in assessing whether a patient is in immediate risk. Similarly, banks want to understand credit risk while promoting good decisions by consumers in regard to true creditworthiness, and wine producers want to predict the marketability of a new vintage while improving their processes for next year. As designers of a learning framework, what degrees of freedom can we utilize to promote good decisions? Our main insight is that *controlling the tradeoff between accuracy and decision quality, where it exists, can be cast as a problem of model selection*. For instance, there may be multiple models with similar predictive performance but with different coefficients that therefore induce very different decisions [5]. To achieve this tradeoff we introduce *lookahead regularization*, which balances accuracy and the improvement associated with induced decisions. Lookahead regularization anticipates how users will act and penalizes a model unless there is high confidence that these decisions will improve outcomes.

Formally, these decisions, which depend on the predictive model, induce a target distribution p' on covariates that may differ from an initial distribution p. For an individual with covariates x, they are mapped to new covariates x'. For a prespecified confidence level τ, we want to guarantee improvement for at least a τ-fraction of the population, comparing outcomes under p' in relation to observed outcomes in p (under an invariance assumption on $p(y|x)$). The technical challenge is that p' may differ considerably from p, resulting in uncertainty in estimating the effect of decisions. To solve this, lookahead regularization makes use of an uncertainty model that provides confidence intervals around decision outcomes. A discriminative uncertainty model is trained under a learning framework that makes use of importance weighting [13, 42, 46] to handle covariate shift and is designed to provide accurate intervals for p'.

Our algorithm has stages that alternate between optimizing the different components of our framework: the *predictive model* (under the lookahead regularization term), the *uncertainty model* (used within the regularization term), and the *propensity model* (used for covariate shift adjustment). If the uncertainty model is differentiable and the predictive model is twice-differentiable, then gradients can pass through the entire pipeline and gradient-based optimization can be applied. We run three experiments. One uses synthetic data and illustrates how our approach can be useful, as well as helping to understand what is needed for it to succeed. The second application is to wine quality prediction and shows that even simple tasks have interesting tradeoffs between accuracy and improved decisions that can be utilized. The third experiment focuses on predicting diabetes progression and includes a demonstration of the framework in a setting with individualized actions.

1.1 Related work

Strategic Classification. In the field of *strategic classification*, the learner and agents engage in a Stackelberg game, where the learner attempts to publish a maximally accurate classifier taking into account that agents will shift their features to obtain better outcomes under the classifier [16]. While early efforts viewed all modifications as "gaming"— an adversarial effect to be mitigated [8, 6] —a recent trend has focused on creating incentives for modifications that lead to better outcomes *under the ground truth function* rather than simply better classifications [21, 2, 15, 47]. In the absence of a known mapping from effort to ground truth, Miller et al. [28] show that incentive design relates to causal modeling, and several responsive works explore how the actions induced by classifiers can facilitate discovery of these causal relationships [32, 3, 41]. The second order effect of strategic classification on algorithmic fairness has also motivated several works [26, 19, 29]. Generally, these works consider the equilibrium effects of classifiers, where the choice of model affects covariate distributions and in turn predictive accuracy. In contrast, we consider what can be done given a snapshot at a point in time, or when the input distribution remains unaffected by user actions.

Causality, Covariate Shift, and Distributionally Robust Learning. There are many efforts in ML to quantify the uncertainty associated with predictions and identify domain regions where models err [24, 18, 12, 14, 48, 25]. However, most methods fail to achieve desirable properties when deployed out of distribution (OOD) [45]. When the shifted distribution is unknown at train time, distributionally robust learning can provide worst-case guarantees for specific types of shifts but require unrealistic computational expense or restrictive assumptions on model classes [44]. Although we do not know ahead of training our shifted distribution of interest, our framework is concerned only

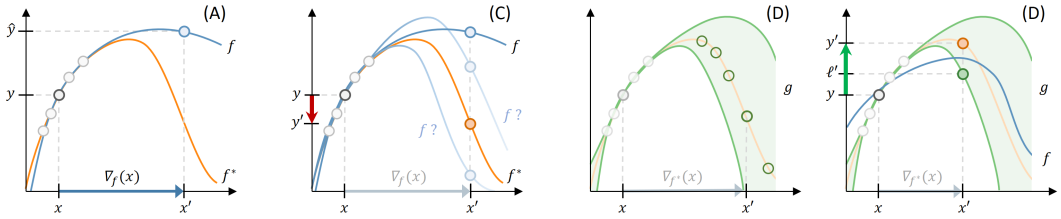

Figure 1: An illustration of our approach. The data density $p(x)$ is concentrated to the left of the peak, and $p(y|x)$ is deterministic, $y = f^*(x)$. (A) Users (x) seeking to improve their outcomes (y) often look to predictive models for guidance on how to act, e.g. by following gradient information ($x \mapsto x'$). (B) But actions may move x' into regions of high uncertainty where f is unconstrained by the training data, and models of equally good fit on p can behave very differently on p'. (C) To reason about the uncertainty in decision outcomes, our approach learns an interval model $g(x') = [\ell', u']$, decoupled from f and targeted specifically at p', guaranteeing that $y' \in [\ell', u']$ with confidence τ. (D) By incorporating into the objective a model of user behavior, lookahead regularization (Eq. (4)) allows for balancing between accuracy and improvement. By penalizing f whenever $y > \ell'$, the model learns predictive models that encourage decisions that are safe, i.e., $y' \geq y$ w.p. at least τ.

with the single, specific OOD distribution that is induced by the learned predictive model. Hence, we need only guarantee robustness to this particular distribution, for which we make use of tools from learning under covariate shift [4]. Relevant to our task, Mueller et al. [30] seek to identify treatments which are beneficial with high probability under the covariate shift assumption. Because model variance generally increases when covariate shift acts on non-causal variables [33], our framework of trading off uncertainty minimization with predictive power relates to efforts in the causal literature to find models which have optimal predictive accuracy while being robust to classes of interventional perturbations [27, 36].

2 Method

Let $x \in \mathcal{X} = \mathbb{R}^d$ denote a feature vector and $y \in \mathbb{R}$ denote a label, where x describes the object of interest (e.g., a patient, a customer, a wine vintage), and y describes the quality of an outcome associated with x, where we assume that higher y is better. We assume an observational dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^m$, which consists of IID samples from a population with joint distribution $(x, y) \sim p(x, y)$ over covariates (features) x and outcomes y. We denote by $p(x)$ the marginal distribution on covariates.

Let $f : \mathcal{X} \to \mathbb{R}$ denote a model trained on $\mathcal{S}$. We assume that f is used in two different ways:

1. **Prediction:** To predict outcomes y for objects x, sampled from $p(x)$.
2. **Decision:** To take action, through changes to x, with the goal of improving outcomes.

We will assume that user actions map each x to a new $x' \in \mathcal{X}$. We refer to x' as a user's *decision* or *action* and denote decision outcomes by $y' \in \mathbb{R}$, We set $x' = d(x)$ and refer to $d : \mathcal{X} \to \mathcal{X}$ as the *decision function*. We will assume that users consult f to drive decisions—either because they care only about predicted outcomes (e.g., the case of bank loans), or because they consider the model to be a valid proxy of the effect of a decision on the outcome (e.g., the case of heart attack risk or wine production). As in other works incorporating strategic users into learning [32, 16], our framework requires an explicit model of how users use the model to make decisions. For concreteness, we model users as making a step in the direction of the gradient of f, but note that the framework can also support any other differential decision model.[2] Since not all attributes may be susceptible to change, we distinguish between *mutable* and *immutable* features using a *masking operator* $\Gamma : \mathcal{X} \to \{0, 1\}^d$.

Assumption 1 (User decision model)**.** *Given masking operator* Γ*, we define user decisions as:*

$$x' = x + \eta\Gamma(\nabla_f(x)), \tag{1}$$

where the step size $\eta \geq 0$ *is a design parameter.*

[2]Many works consider decisions that take gradient steps under cost constraints $c(x, x') < B$. Note that such constraints can be incorporated into learning using, for example, differentiable optimization layers [1].

Through Assumption 1, user decisions induce a particular decision function $d(x)$, and in turn, a *target distribution* over $\mathcal{X}$, which we denote $p'(x)$. This leads to a new joint distribution $(x', y') \sim p'(x, y)$, with decisions inducing new outcomes. To achieve causal validity in the way we reason about the effect of decisions on outcomes, we follow Peters et al. [33] and assume that y depends only on x and is invariant to the distribution on x.

Assumption 2 (Covariate shift [42])**.** *The conditional distribution on outcomes, $p(y|x')$, is invariant for any arbitrary, marginal distribution $p'(x)$ on covariates, including the data distribution $p(x)$.*

Assumption 2 says that whatever the transform d, conditional distribution $p(y|x)$ is fixed, and the new joint distribution is $p'(x', y) = p(y|x')p'(x')$, for any p' (note that p' also depends on f). This covariate-shift assumption ensures the causal validity of our approach (and entails the property of *no-unobserved confounders*). Although a strong assumption, this kind of invariance has been leveraged in other works that relate to questions of causality [35, 30], as well as for domain adaptation [39, 34].

2.1 Learning objective

Our goals in designing a learning framework are twofold. First, we would like learning to result in a model whose predictions $\hat{y} = f(x)$ closely match the corresponding labels y for $x \sim p(x)$. Second, we would like the model to induce decisions x' for counterfactual distribution p' whose outcome y' improves upon the initial y. To balance between these two goals, we construct a learning objective in which a predictive loss function is augmented with a regularization term that promotes good decisions. The difficulty is that decision outcomes y' depend on decisions x' through the learned model f. Hence, realizations of y' are unavailable at train time, as they cannot be observed until after the model is deployed. For this reason, simple constraints of the form $y' \geq y$ are ill-defined, and to regularize we must reason about outcome distributions $y' \sim p(y|x')$, for $x' \sim p'$. A naive approach might consider the average improvement, with $\mu' = \mathbb{E}_{y' \sim p(y|x')}[y']$, for a given $x' \sim p'$, and penalize the model whenever $\mu' < y$, for example linearly using $\mu' - y$. Concretely, μ' must be estimated, and since f minimizes MSE, then $\hat{y}' = f(x')$ is a plausible estimate of μ', giving:

$$\min_{f \in F} \mathbb{E}_{p(x,y)}[(\hat{y} - y)^2] + \lambda \mathbb{E}_{p(x,y)}[\hat{y}' - y], \qquad \hat{y}' = f(x'), \tag{2}$$

where $\lambda \geq 0$ determines the relative importance of improvement over accuracy. There are two issues with this approach. First, learning can result in an f that severely overfits in estimating μ, meaning that at train time the penalty term in the (empirical) objective will appear to be low whereas at test time its (expected) value will be high. This can happen, for example, when x' is moved to a low-density region of $p(x)$ where f is unconstrained by the data and, if flexible enough, can artificially (and wrongly) signal improvement. To address this we use two decoupled models—one for predicting y on distribution p, and another for handling y' on distribution p'.

Second, in many applications it may be unsafe to guarantee that improvement hold only on average per individual (e.g., heart attack risk, credit scores). To address this, we encourage f to improve outcomes with a certain degree of confidence τ, for $\tau > 0$, i.e., such that $\mathbb{P}[y' \geq y] \geq \tau$ for a given (x, y) and induced x' and thus $p(y'|x')$. Importantly, while one source of uncertainty in y' is $p(y|x')$, other sources of uncertainty exist, including those coming from insufficient data as well as model uncertainty. Our formulation is useful when additional sources of uncertainty are significant, such as when the model f leads to actions that place x' in low-density regions of p.

In our method, we replace the average-case penalty in Eq. (2) with a *confidence-based penalty*:

$$\min_{f \in F} \mathbb{E}_{p(x,y)}[(\hat{y} - y)^2] + \lambda \mathbb{E}_{p(x,y)}[\mathbb{1}\{\mathbb{P}[y' \geq y] < \tau\}], \qquad y' \sim p(y|x'), \tag{3}$$

where $\mathbb{1}\{A\} = 1$ if A is true, and 0 otherwise. In practice, $\mathbb{P}[y' \geq y]$ is unknown, and must be estimated. For this, we make use of an *uncertainty model*, $g_\tau : \mathcal{X} \to \mathbb{R}^2$, $g_\tau \in G$, which we learn, and maps points $x' \in \mathcal{X}$ to intervals $[\ell', u']$ that cover y' with probability τ. We also replace the penalty term in Eq. (3) with the slightly more conservative $\mathbb{1}\{\ell' < y\}$, and to make learning feasible we use the hinge loss $\max\{0, y - \ell'\}$ as a convex surrogate.[3] For a given uncertainty model, g_τ, the

[3]The penalty is conservative in that it considers only one-sided uncertainty, i.e., $y' < \ell$ and u is not used explicitly. Although open intervals suffice here, most methods for interval prediction consider closed intervals, and in this way our objective can support them. For symmetric intervals, τ simply becomes $\tau/2$.

empirical learning objective for model f on sample set $\mathcal{S}$ is:

$$\min_{f \in F} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 + \lambda R(g_\tau; \mathcal{S}), \quad \text{where } R(g_\tau; \mathcal{S}) = \sum_{i=1}^{m} \max\{0, y_i - \ell'_i\}, \tag{4}$$

where $R(g_\tau; \mathcal{S})$ is the *lookahead regularization* term.

By anticipating how users decide, this penalizes models whose induced decisions do not improve outcomes at a sufficient rate (see Figure 1).The novelty in the regularization term is that it accounts for uncertainty in assessing improvement, and does so for points x' that are out of distribution. If f pushes x' towards regions of high uncertainty, then the interval $[\ell', u']$ is likely to be large, and f must make more "effort" to guarantee improvement, something that may come at some cost to in-distribution prediction accuracy. As a byproduct, while the objective encodes the rate of decision improvement, we will also see the magnitude of improvement increase in our experiments.

Note that the regularization term R depends both on f and g—to determine x', and to determine ℓ' given x', respectively. This justifies the need for the decoupling of f and g: without this, uncertainty estimates based on $f(x')$ are prone to overfit by artificially manipulating intervals to be higher than y, resulting in low penalization at train time without actual improvement (see Figure 2 (right)).

2.2 Estimating uncertainty

The usefulness of lookahead regularization relies on the ability of the uncertainty model g to correctly capture the various kinds of uncertainties about the outcome value for the perturbed points. This can be difficult because uncertainty estimates are needed for out-of-distribution points x'.

Fortunately, for a given f the counterfactual distribution p' is known (by Assumption 1), and we can use the covariate transform associated with the decision to construct sample set $\mathcal{S}' = \{x'_i\}_{i=1}^m$. Even without labels for $\mathcal{S}'$, estimating g is now a problem of *learning under covariate shift*, where the test distribution p' can differ from the training distribution p. In particular, we are interested in learning uncertainty intervals that provide good coverage. There are many approaches to learning under covariate shift. Here we describe the simple and popular method importance weighting, or inverse propensity weighting [42]. For a loss function $L(g) = L(y, g(x))$, we would like to minimize $\mathbb{E}_{p'(x,y)}[L]$. Let $w(x) = p'(x)/p(x)$, then by the covariate shift assumption:

$$\mathbb{E}_{p'(x,y)}[L(g)] = \int L(g)\, \mathrm{d}p'(x)\, \mathrm{d}p(y|x) = \int \frac{p'(x)}{p(x)} L(g)\, \mathrm{d}p(x)\, \mathrm{d}p(y|x) = \mathbb{E}_{p(x,y)}[w(x)L(g)].$$

Hence, training g with points sampled from distribution p but weighted by w will result in an uncertainty model that is optimized for the counterfactual distribution p'. In practice, w is itself unknown, but many methods exist for learning an approximate model $\hat{w}(x) \approx w(x)$ using sample sets $\mathcal{S}$ and $\mathcal{S}'$ (e.g. [20]). To remain within our discriminative approach, here we follow [4] and train a logistic regression model $h : \mathcal{X} \to [0, 1]$, $h \in H$, to differentiate between points $\tilde{x} \in \mathcal{S}$ (labeled $\tilde{y} = 0$) and $\tilde{x} \in \mathcal{S}'$ (labeled $\tilde{y} = 1$) and set weights to $\hat{w}(x) = e^{h(x)}$. As we are interested in training g to gain a coverage guarantee, we define $L(y, g(x)) = \mathbb{1}\{y \notin [\ell, u]\}$.

2.3 Algorithm

All the elements in our framework— the predictive model f, the uncertainty model g, and the propensity model h —are interdependent. Specifically, optimizing f in Eq. (4) requires intervals from g; learning g requires weights from h; and h is trained on $\mathcal{S}'$ which is in turn determined by f. Our algorithm therefore alternates between optimizing each of these components while keeping the others fixed. At round t, $f^{(t)}$ is optimized with intervals $[\ell'_i, u'_i] = g^{(t-1)}(x'_i)$, $g^{(t)}$ is trained using weights $w_i = h^{(t)}(x_i)$, and $h^{(t)}$ is trained using points x'_i as determined by $f^{(t)}$. The procedure is initialized by training $f^{(0)}$ without the lookahead term R. For training g and h, weights $w_i = \hat{w}(x_i)$ and points $\{x'_i\}_{i=1}^m$, respectively, can be precomputed and plugged into the objective. Training f with Eq. (4), however, requires access to the *function* g, since during optimization, the lower bounds ℓ' must be evaluated for points x' that vary as updates to f are made. Hence, to optimize f with gradient methods, we use an uncertainty model g that is differentiable, so that gradients can pass

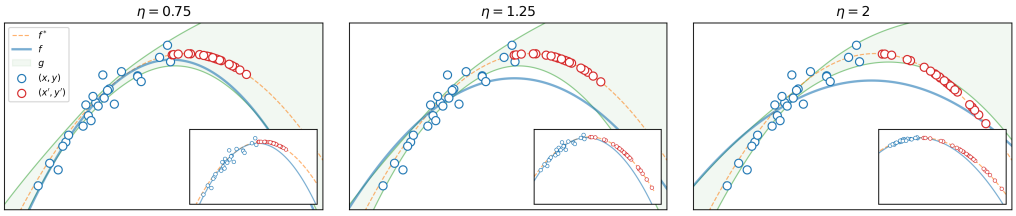

Figure 2: Results for the synthetic experiment for lookahead (main plot) and baseline (inlay) models.

through them (while keeping their parameters fixed). Furthermore, since gradients must also pass through x' (which includes ∇_f), we require that f be twice-differentiable.

In the experiments we consider two methods for learning g:

1. Bootstrapping [10], where a collection of models $\{g^{(i)}\}_{i=1}^{k}$ is trained for prediction each on a subsampled dataset and combined to produce a single interval model g, and

2. Quantile regression [22], where models $g^{(\ell)}, g^{(u)}$ are discriminatively trained to estimate the τ and $1-\tau$ quantiles, respectively, of the counterfactual target distribution $p'(y|x')$.

3 Experiments

In this section, we evaluate our approach in three experiments of increasing complexity and scale, where the first is synthetic and the latter two use real data. Because the goal of regularization is to balance accuracy with decision quality, we will be interested in understanding the attainable frontier of accuracy vs. improvement. For our method, this will mostly be controlled by varying lookahead regularization parameter, $\lambda \geq 0$. In all experiments we measure predictive accuracy with root mean squared error (RMSE), and decision quality in two ways: *mean improvement rate* $\mathbb{E}[\mathbb{1}\{y'_i > y_i\}]$ and *mean improvement magnitude* $\mathbb{E}[y'_i - y_i]$.

To evaluate the approach, we need a means for evaluating counterfactual outcomes y' for decisions x'. Therefore, and similarly to Shavit and Moses [40], we make use of an inferred 'ground-truth' function f^* to test decision improvement, assuming $y' = f^*(x')$. Model f^* is trained on the entirety of the data. By optimizing f^* for RMSE, we think of this as estimating the conditional mean of $p(y|x)$, with the data labels as (arbitrarily) noisy observations. To make for an interesting experiment, we learn f^* from a function class F^* that is more expressive than F or G. The sample set $\mathcal{S}$ will contain a small and possibly biased subsample of the data, which we call the 'active set', and that plays the role of a representative sample from p. This setup allows us not only to evaluate improvement, but also to experiment with the effects of different sample sets.

3.1 Experiment 1: Quadratic curves

For a simple setting, we explore the effects of regularized and unregularized learning on decision quality in a stylized setting using unidimensional quadratic curves. Let $f^*(x) = -x^2$, and assume $y = f(x) + \varepsilon$ where ε is independently, normally distributed. By varying the decision model step-size η, we explore three conditions: one where a naïve approach works well, one where it fails but regularization helps, and one where regularization also fails.

In Figure 2 (left), η is small, and the x' points stay within the high certainty region of p. Here, the baseline works well, giving both a good fit and effective decisions, and the regularization term in the lookahead objective remains inactive. In Figure 2 (center), η is larger. Here, the baseline model pushes x' points to a region where y' values are low. Meanwhile, the lookahead model, by incorporating into the objective the decision model and estimating uncertainty surrounding y', is able to adjust the model to induce good decisions with some reduction in accuracy. In Figure 2 (right), η is large. Here, the x' points are pushed far into areas of high uncertainty. The success of lookahead relies on the successful construction of intervals at p' through the successful estimation of w, and may fail if p and p' differ considerably, as is the case here.

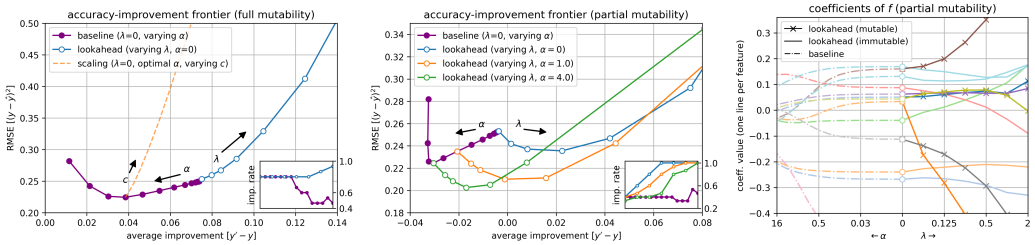

Figure 3: Results for the wine experiment. Tradeoff in accuracy and improvement under full mutability (left) and partial mutability (center), for which model coefficients are also shown (right).

3.2 Experiment 2: Wine quality

The second experiment focuses on wine quality using the wine dataset from the UCI data repository [9]. The wine in the data set has 13 features, most of which correlate linearly with quality y, but two of which (alcohol and malic acid) have a non-linear U-shaped or inverse-U shaped relationship with y. For the ground truth model, we set $f^*(x) = \sum_i \theta_i x_i + \sum_i \theta'_i x_i^2$ (RMSE $= 0.2$, $y \in [0, 3]$) so that it captures these nonlinearities, . To better demonstrate the capabilities of our framework, we sample points into the active set non-uniformly by thresholding on the non-linear features. The active set includes $\sim$30% of the data, and is further split 75-25 into a train set used for learning and tuning and a held-out test set used for final evaluation.

For the predictive model, our focus here is on linear models. The baseline includes a linear f_{base} trained with ℓ_2 regularization (i.e., Ridge Regression) with regularization coefficient $\alpha \geq 0$. Our lookahead model includes a linear f_{look} trained with lookahead regularization (Eq. (4)) with regularization coefficient $\lambda \geq 0$. In some cases we will add to the objective an additional ℓ_2 term, so that for a fixed α, setting $\lambda = 0$ recovers the baseline model. Lookahead was trained for 10 rounds and the baseline with a matching number of overall epochs. The uncertainty model g uses residuals-based bootstrapping with 20 linear sub-models. The propensity model h is also linear. We consider two settings: one where all features (i.e., wine attributes) are mutable and using decision step-size $\eta = 0.5$, and another where only a subset of the features are mutable and using step-size $\eta = 2$.

Full mutability. Figure 3 (left) presents the frontier of accuracy vs. improvement on the test set when all features are mutable. The baseline and lookahead models coincide when $\alpha = \lambda = 0$. For the baseline, as α increases, predictive performance (RMSE) displays a typical learning curve with accuracy improving until reaching an optimum at some intermediate value of α. Improvement, however, monotonically decreases with α, and is highest with no regularization ($\alpha = 0$). This is because in this setting, gradients of f_{base} induce reasonably good decisions: f_{base} is able to approximately recover the dominant linear coefficients of f^*, and shrinkage due to higher ℓ_2 penalization reduces the magnitude of the (typically positive, on average) change. With lookahead, increasing λ leads to better decisions, but at the cost of higher (albeit sublinear) RMSE. The initial improvement rate at $\lambda = 0$ is high, but lookahead and ℓ_2 penalties have opposing effects on the model. Here, improvement is achieved by (and likely requires) increasing the size of the coefficients of linear model, f_{look}. We see that f_{look} learns to do this in an efficient way, as compared to a naïve scaling of the predictively-optimal f_{base}.

Partial mutability. Figure 3 (center) presents the frontier of accuracy vs. improvement when only a subset of the features are mutable (note that this effects the scale of possible improvement). The baseline presents a similar behavior to the fully-mutable setting, but with the optimal predictive model inducing a negative improvement. Here we consider lookahead with various degrees of additional ℓ_2 regularization. When $\alpha = \lambda = 0$, the models again coincide. However, for larger λ, significant improvement can be gained with very little or no loss in RMSE, while moderate λ values improve both decisions and accuracy. This holds for various values of α, and setting α to the optimal value of f_{base} results in lookahead dominating the trade-off curve for all observed λ. Improvement is reflected in magnitude and rate, which rises quickly from the baseline's $\sim 40\%$ to an optimal 100%, showing how lookahead learns models that lead to safe decisions.

Figure 3 (right) shows how the coefficients of f_{base} and f_{look} change as α and λ increase, respectively (for lookahead $\alpha = 0$). As can be seen, lookahead works by making substantial changes to mutable

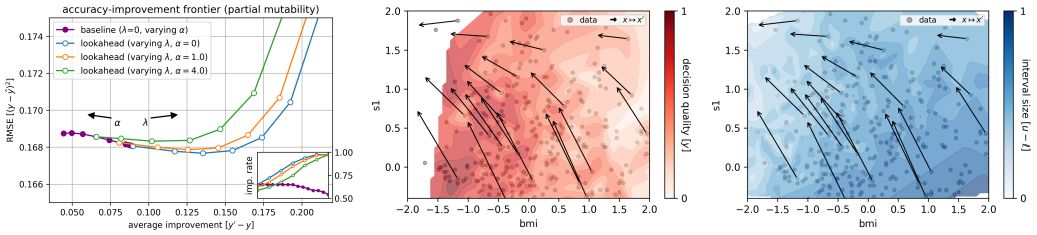

Figure 4: Results for the diabetes experiment. Tradeoff in accuracy and improvement under linear f with partial mutability (left), visualization of shift $p \to p'$ with non-linear f to regions of higher decision quality (center), and regions of lower uncertainty (right).

coefficients, sometimes reversing their sign, with milder changes to immutable coefficients. Lookahead achieves improvement by capitalizing on its freedom to learn a useful direction of improvement within the mutable subspace, while compensating for the possible loss in accuracy through mild changes in the immutable subspace.

3.3 Experiment 3: Diabetes

The final experiment focuses on the prediction of diabetes progression using the diabetes dataset[4] [11]. The dataset has 10 features describing various patient attributes. We consider two features as mutable: BMI and T-cell count (marked as 's1'). While both display a similar (although reversed) linear relationship with y, feature s1 is much noisier. The setup is as in wine but with two differences: to capture nonlinearities we set f^* to be a flexible generalized additive model (GAM) with splines of degree 10 (RMSE $= 0.15$), and train and test sets are sampled uniformly from the data. We normalize y to $[0, 1]$ and set $\eta = 5$.

Figure 4 (left) presents the accuracy-improvement frontier for linear f and bootstrapped linear g. Results show a similar trend to the wine experiment, with lookahead providing improved outcomes (both rate and magnitude) while preserving predictive accuracy. Here, lookahead improves results by learning to increase the coefficient of s1, while adjusting other coefficients to maintain reasonable uncertainty. The baseline fails to utilize s1 for improvement since from a predictive perspective there is little value in placing weight on s1.

When f is linear, decisions are uniform across the population in that $\nabla_{f_\theta}(x) = \theta$ is independent of x. To explore individualized actions, we also consider a setting where f is a more flexible quadratic model (i.e., linear in x and x^2) in which gradients depend on x and uncertainty is estimated using quantile regression. Figure 4 (center) shows the data as projected onto the subspace $(x_{\text{BMI}}, x_{\text{s1}})$, with color indicating outcome values $f^*(x)$, interpolated within this subspace. As can be seen, the mapping $x \mapsto x'$ due to f_{look} generally improves outcomes. The plot reveals that, had we had knowledge of $f^*(x)$, uniformly decreasing BMI would also have improved outcomes, and this is in fact the strategy envoked by the linear f_{base}. But decisions must be made based on the sample set, and so uncertainty must be taken into account. Figure 4 (right) shows a similar plot but with color indicating uncertainty estimates as measured by the interval sizes given by g. The plot shows that decisions are directed towards regions of lower uncertainty (i.e., approximately following the negative gradients of the uncertainty slope), showing how lookahead successfully utilizes these uncertainties to adjust the predictive model f_{look}.

4 Discussion

Given the extensive use of machine learning across an ever-growing range of applications, we think it is appropriate to assume, as we have here, that predictive models will remain in widespread use, and that at the same time, and despite well-understood concerns, users will continue to act upon them. In line with this, our goal with this work has been to develop a machine learning framework that accounts for decision making by users but remains fully within the discriminative framing of statistical machine learning. The lookahead regularization framework that we have proposed

[4] `https://www4.stat.ncsu.edu/~boos/var.select/diabetes.html`

augments existing machine learning methodologies with a component that promotes good human decisions. We have demonstrated the utility of this approach across three different experiments, one on synthetic data, one on predicting and deciding about wine, and one on predicting and deciding in regard to diabetes progression. We hope that this work will inspire continued research in the machine learning community that embraces predictive modeling while also being cognizant of the ways in which our models are used.

Broader Impact

In our work, the learning objective was designed to align with and support the possible use of a predictive model to drive decisions by users. It is our belief that a responsible and transparent deployment of models with "lookahead-like" regularization components should avoid the kinds of mistakes that can be made when predictive methods are conflated with causally valid methods.

At the same time, we have made a strong simplifying assumption, that of covariate shift, which requires that the relationship between covariates and outcome variables is invariant as decisions are made and the feature distribution changes. This strong assumption is made to ensure validity for the lookahead regularization, since we need to be able to perform inference about counterfactual observations. As discussed by Mueller et al. [30] and Peters et al. [33], there exist real-world tasks that reasonably satisfy this assumption, and yet at the same time, other tasks— notably those with unobserved confounders —where this assumption would be violated. Moreover, this assumption is not testable on the observational data. This, along with the need to make an assumption about the user decision model, means that an application of the method proposed here should be done with care and will require some domain knowledge to understand whether or not the assumptions are plausible.

Furthermore, the validity of the interval estimates requires that any assumptions for the interval model used are satisfied and that weights w provide a reasonable estimation of p'/p. In particular, fitting to p' which has little to no overlap with p (see Figure 2) may result in underestimating the possibility of bad outcomes.

If used carefully and successfully, then the system provides safety and protects against the misuse of a model. If used in a domain for which the assumptions fail to hold then the framework could make things worse, by trading accuracy for an incorrect view of user decisions and the effect of these decisions on outcomes.

We would also caution against any specific interpretation of the application of the model to the wine and diabetes data sets. We note that model misspecification of f^* could result in arbitrarily bad outcomes, and estimating f^* in any high-stakes setting requires substantial domain knowledge and should err on the side of caution. We use the data sets for purely illustrative purposes because we believe the results are representative of the kinds of results that are available when the method is correctly applied to a domain of interest.

References

[1] Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and J Zico Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems*, pages 9558–9570, 2019.

[2] Tal Alon, Magdalen Dobson, Ariel D Procaccia, Inbal Talgam-Cohen, and Jamie Tucker-Foltz. Multiagent evaluation mechanisms.

[3] Yahav Bechavod, Katrina Ligett, Zhiwei Steven Wu, and Juba Ziani. Causal feature discovery through strategic modification. *arXiv preprint arXiv:2002.07024*, 2020.

[4] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(Sep):2137–2155, 2009.

[5] Leo Breiman et al. Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Statistical science*, 16(3):199–231, 2001.

[6] Michael Brückner and Tobias Scheffer. Stackelberg games for adversarial prediction problems. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 547–555, 2011.

[7] Alison Callahan and Nigam H Shah. Machine learning in healthcare. In *Key Advances in Clinical Informatics*, pages 279–291. Elsevier, 2017.

[8] Jinshuo Dong, Aaron Roth, Zachary Schutzman, Bo Waggoner, and Zhiwei Steven Wu. Strategic classification from revealed preferences. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, pages 55–70, 2018.

[9] Dheeru Dua and Casey Graff. Uci machine learning repository, 2017.

[10] Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

[11] Bradley Efron, Trevor Hastie, Iain Johnstone, Robert Tibshirani, et al. Least angle regression. *The Annals of statistics*, 32(2):407–499, 2004.

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[13] Arthur Gretton, Alex Smola, Jiayuan Huang, Marcel Schmittfull, Karsten Borgwardt, and Bernhard Schölkopf. Covariate shift by kernel mean matching. *Dataset shift in machine learning*, 3(4):5, 2009.

[14] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1321–1330. JMLR. org, 2017.

[15] Nika Haghtalab, Nicole Immorlica, Brendan Lucier, and Jack Wang. Maximizing welfare with incentive-aware evaluation mechanisms. Technical report, working paper, 2020.

[16] Moritz Hardt, Nimrod Megiddo, Christos Papadimitriou, and Mary Wootters. Strategic classification. In *Proceedings of the 2016 ACM conference on innovations in theoretical computer science*, pages 111–122, 2016.

[17] Sohaib Haseeb, Bryce Alexander, and Adrian Baranchuk. Wine and cardiovascular health: A comprehensive review. *Circulation*, 136(15):1434–1448, 2017.

[18] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.

[19] Lily Hu, Nicole Immorlica, and Jennifer Wortman Vaughan. The disparate effects of strategic manipulation. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 259–268, 2019.

[20] Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. A least-squares approach to direct importance estimation. *Journal of Machine Learning Research*, 10(Jul):1391–1445, 2009.

[21] Jon Kleinberg and Manish Raghavan. How do classifiers induce agents to invest effort strategically? In *Proceedings of the 2019 ACM Conference on Economics and Computation*, pages 825–844, 2019.

[22] Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15(4):143–156, 2001.

[23] Augustine Kong, Jun S Liu, and Wing Hung Wong. Sequential imputations and bayesian missing data problems. *Journal of the American statistical association*, 89(425):278–288, 1994.

[24] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413, 2017.

[25] Jeremiah Liu, John Paisley, Marianthi-Anna Kioumourtzoglou, and Brent Coull. Accurate uncertainty estimation and decomposition in ensemble learning. In *Advances in Neural Information Processing Systems*, pages 8950–8961, 2019.

[26] Lydia T Liu, Ashia Wilson, Nika Haghtalab, Adam Tauman Kalai, Christian Borgs, and Jennifer Chayes. The disparate equilibria of algorithmic decision making when individuals invest rationally. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 381–391, 2020.

[27] Nicolai Meinshausen. Causality from a distributional robustness point of view. In *2018 IEEE Data Science Workshop (DSW)*, pages 6–10. IEEE, 2018.

[28] John Miller, Smitha Milli, and Moritz Hardt. Strategic adaptation to classifiers: A causal perspective. *arXiv preprint arXiv:1910.10362*, 2019.

[29] Smitha Milli, John Miller, Anca D Dragan, and Moritz Hardt. The social cost of strategic classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*, pages 230–239, 2019.

[30] Jonas Mueller, David N Reshef, George Du, and Tommi Jaakkola. Learning optimal interventions. *arXiv preprint arXiv:1606.05027*, 2016.

[31] Judea Pearl et al. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.

[32] Juan C Perdomo, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. Performative prediction. *arXiv preprint arXiv:2002.06673*, 2020.

[33] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):947–1012, 2016.

[34] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

[35] Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. Invariant models for causal transfer learning. *The Journal of Machine Learning Research*, 19(1):1309–1342, 2018.

[36] Dominik Rothenhäusler, Nicolai Meinshausen, Peter Bühlmann, and Jonas Peters. Anchor regression: heterogeneous data meets causality. *arXiv preprint arXiv:1801.06229*, 2018.

[37] Donald B Rubin. Causal inference using potential outcomes: Design, modeling, decisions. *Journal of the American Statistical Association*, 100(469):322–331, 2005.

[38] Amirhossein Sahebkar, Corina Serban, Sorin Ursoniu, Nathan D Wong, Paul Muntner, Ian M Graham, Dimitri P Mikhailidis, Manfredi Rizzo, Jacek Rysz, Laurence S Sperling, et al. Lack of efficacy of resveratrol on c-reactive protein and selected cardiovascular risk factors—results from a systematic review and meta-analysis of randomized controlled trials. *International journal of cardiology*, 189:47–55, 2015.

[39] Gabriele Schweikert, Gunnar Rätsch, Christian Widmer, and Bernhard Schölkopf. An empirical analysis of domain adaptation algorithms for genomic sequence analysis. In *Advances in neural information processing systems*, pages 1433–1440, 2009.

[40] Yonadav Shavit and William S Moses. Extracting incentives from black-box decisions. *arXiv preprint arXiv:1910.05664*, 2019.

[41] Yonadav Shavit, Benjamin Edelman, and Brian Axelrod. Learning from strategic agents: Accuracy, improvement, and causality. *arXiv preprint arXiv:2002.10066*, 2020.

[42] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000.

[43] Naeem Siddiqi. *Credit risk scorecards: developing and implementing intelligent credit scoring*, volume 3. John Wiley & Sons, 2012.

[44] Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

[45] Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D Sculley, Joshua Dillon, Jie Ren, and Zachary Nado. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13969–13980, 2019.

[46] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul V Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems*, pages 1433–1440, 2008.

[47] Behzad Tabibian, Stratis Tsirtsis, Moein Khajehnejad, Adish Singla, Bernhard Schölkopf, and Manuel Gomez-Rodriguez. Optimal decision making under strategic behavior. *arXiv preprint arXiv:1905.09239*, 2019.

[48] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In *Advances in Neural Information Processing Systems*, pages 6414–6425, 2019.

[49] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.

[50] Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. Machine learning in manufacturing: advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1):23–45, 2016.

Appendix A Pseudocode

Our algorithm alternates between optimizing the three components of the framework: a predictive model, a propensity model, and an uncertainty model. Here we give pseudocode for the following per-component objectives:

1. A predictive model $\hat{y} = f(x)$, optimizing the squared loss:

$$L_{\text{pred}}(f; \mathcal{S}) = \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

2. A propensity weight model $w = e^{h(x)}$, optimizing the log-loss:

$$L_{\text{prop}}(h; \mathcal{S}, \mathcal{S}') = \sum_{i=1}^{m} \log(1 + e^{h(x_i)}) + \log(1 + e^{-h(x_i')})$$

3. An uncertainty interval model $[\ell, u] = g_\tau(x)$, optimizing the τ-quantile loss:

$$L^{(\tau)}_{\text{uncert}}(g; \mathcal{S}, w) = \sum_{i=1}^{m} w(x_i) \max\{(\tau - 1)(y_i - \ell_i), \tau(y_i - \ell_i)\}$$

but note that others can be plugged in. The pseudocode is given below.

Algorithm 1 Lookahead$(\mathcal{S}, T, \lambda, \eta, \tau)$

$f^{(0)} \leftarrow \text{argmin}_{f \in F} L_{\text{pred}}(f; \mathcal{S})$
for $t = 1, \ldots, T$ **do**
  $x_i' \leftarrow d_\eta(x_i; f^{(t-1)})$ for all $i = 1, \ldots, m$ ▷ e.g., $d_\eta(x; f) = x + \eta\Gamma(\nabla_f(x))$
  $\mathcal{S}' \leftarrow \{x_i'\}_{i=1}^m$
  $h^{(t)} \leftarrow \text{argmin}_{h \in H} L_{\text{prop}}(h; \mathcal{S}, \mathcal{S}')$
  $w \leftarrow e^{h^{(t)}}$
  $g^{(t)} \leftarrow \text{argmin}_{g \in G} L^{(\tau)}_{\text{uncert}}(g; \mathcal{S}, w)$
  $f^{(t)} \leftarrow \text{argmin}_{f \in F} L_{\text{pred}}(f; \mathcal{S}) + \lambda R(g^{(t)}; \mathcal{S})$
return $f^{(T)}$

Appendix B Uncertainty models

Here we describe the two uncertainty methods used in our paper and how they apply to our setting.

B.1 Bootstrapping

Bootstrapping produces uncertainty intervals by combining the outputs of a collection of k models $\{g^{(i)}\}_{i=1}^k$, each trained independently for *prediction* on a random subset of the data. There are many approaches to bootstrapping, and here we describe two:

- **Vanilla bootstrapping**: Each $g^{(i)}$ is trained using a predictive objective (e.g., squared loss) on a sample set $\mathcal{S}^{(i)} = \{(x_j^{(i)}, y_j^{(i)})\}_{j=1}^m$ where $(x_j^{(i)}, y_j^{(i)})$ are sampled with replacement from $\mathcal{S}$. The sub-models are then combined using:

$$g(x) = [\mu(x) - z\sigma(x), \mu(x) + z\sigma(x)]$$

where:

$$\mu(x) = \frac{1}{k} \sum_{i=1}^{k} g^{(i)}(x), \qquad \sigma(x) = \frac{1}{k} \sum_{i=1}^{k} (\mu(x) - g^{(i)}(x))^2$$

and z is the z-score corresponding to the confidence parameter τ under a normal distribution.

- **Bootstrapping residuals**: First, a predictive model $\bar{g}$ is fit to the data, and residuals $r = y - \bar{g}(x)$ are computed. Then, each $g^{(i)}$ is trained on the original sample data but with ground truth-labels y_i replaced with random pseudo-labels:

$$\mathcal{S}^{(i)} = \{(x_j, \bar{y}_j^{(i)})\}_{j=1}^m \qquad \bar{y}_j^{(i)} = y_i + r_j$$

where r_j are sampled with replacement from $\{r_j\}_{j=1}^m$.

In our framework, because g must apply to p', each $g^{(i)}$ is trained with propensity weights w. To account for cases where p and p' differ, the $g^{(i)}$ are trained not on sample sets of size m, but rather, of size $\tilde{m}(w)$, where $\tilde{m}(w)$ is the *effective sample size* [23] given by:

$$\tilde{m}(w) = \frac{\text{mean}(\{w_i\}_{i=1}^m)}{\text{var}(\{w_i\}_{i=1}^m)}, \qquad w_i = w(x_i) \;\; \forall\, i = 1, \ldots, m$$

B.2 Quantile regression

Quatile regression is a learning framework for training models to predict the τ-quantile of the conditional label distribution $p(y|x)$. Just as training with the squared loss is aimed at predicting the mean of $p(y|x)$, training with the absolute loss $|y - \hat{y}|$ is aimed at the median. Quantile regression generalizes the absolute loss by considering a 'tilted' variant with slopes $\tau - 1$ and τ:

$$Q_\tau(y, \hat{y}) = \max\{(1-\tau)(y - \hat{y}), \tau(y - \hat{y})\}$$

Appendix C Experimental details

C.1 Experiment 1: Quadratic curves

Here we set $f^*(x) = -0.8x^2 + 0.5x + 0.1$. F and G include quadratic functions, and H to include linear functions. For uncertainty estimation we used vanilla bootstrap, and for propensity scores we used logistic regression. For lookahead, we set $\lambda = 4$, $\tau = 0.95$, use $k = 10$ bootstrapped models, and train for $T = 5$ rounds. The data includes $m = 25$ samples x drawn from $N(-0.8, 0.5)$, and $y = f^*(x) + \epsilon$ where $\epsilon \sim N(0, 0.25)$. We use a $75 : 25$ train-test split. The three conditions vary only in η with values $\eta = 0.75, 1.25$, and 3.5.

Quantitative results are given in the table below:

		RMSE	Imp. rate	Imp. mag.
$\eta = 0.75$	baseline	0.349	0.857	1.109
	lookahead	0.351	0.857	1.108
$\eta = 1.25$	baseline	0.342	0.143	-0.261
	lookahead	0.424	0.714	1.065
$\eta = 3.5$	baseline	0.342	0	-35.13
	lookahead	0.675	0.571	0.604

C.2 Experiment 2: Wine quality

The wine dataset includes $m = 178$ examples and $d = 13$ features. We learn a quadratic $f^*(x) = \sum_i \theta_i x_i + \sum_i \theta'_i x_i^2$. F, G, and H include linear functions. For uncertainty estimation we used residuals bootstrap, and for propensity scores we used logistic regression. For lookahead, we set $\tau = 0.95$, use $k = 20$ bootstrapped models, and train for $T = 10$ rounds. For f, we use SGD with a learning rate of 0.1 and 1000 epochs for initialization and 100 additional epochs per round. For g, each sub-model was trained with SGD using a learning rate of 0.1 and for 500 epochs. We set $\eta = 0.5$ and $\eta = 2$ for the fully and partially mutable settings, respectively.

C.3 Experiment 3: Diabetes

The diabetes dataset includes $m = 442$ examples and $d = 10$ features. We set $f^*(x)$ to be a generalized additive model (GAM) with splines of degree 10 trained on the entire dataset and tuned

using cross-validation. In the first setting, F, G, and H include linear functions. In the second setting, F, G are quadratic functions (i.e., linear in x_i and in x_i^2) and H remains linear. For uncertainty estimation we used quantile regression, and for propensity scores we used logistic regression. For lookahead, we set $\tau = 0.8$ and train for $T = 10$ rounds. For f, we use SGD with a learning rate of 0.05 and 1000 epochs for initialization and 100 additional epochs per round. For g, we use SGD with a learning rate of 0.05 and for 500 epochs. For both linear and non-linear settings we set $\eta = 5$, and normalize y to be in $[0, 1]$.