Cost-Effective Feature Selection and Ordering for Personalized Energy Estimates

Kirstin Early
Machine Learning Department
Carnegie Mellon University
kearly@cs.cmu.edu

Abstract

Background Selecting homes with energy-efficient infrastructure is important for renters, because infrastructure influences energy consumption more than in-home behavior. Personalized energy estimates can guide prospective tenants toward energy-efficient homes, but this information is not readily available. Utility estimates are not typically offered to house-hunters, and existing technologies like carbon calculators require users to answer prohibitively many questions that may require considerable research to answer.

Aim We want to provide prospective tenants with personalized predictions for energy consumption that are certain (have narrow prediction intervals), accurate (are close to the true values), and cheap (require minimal burden on the user from answering questions). Giving energy estimates for a household requires eliciting information about that household; our goal is to strategically order questions so that the most informative features are answered first, to give confident predictions with minimal user burden.

Data The Residential Energy Consumption Survey (RECS) records energy consumption by fuel type and around 500 household features for homes across the U.S.; we can use this dataset to learn relationships between household features and energy consumption to predict energy usage for prospective tenants. We restricted our analysis to the 2470 homes in the same climate zone as Pittsburgh.

Methods For the task of providing personalized utility estimates to prospective tenants, we present a cost-based model for feature selection at training time, where all features are available and costs assigned to each feature reflect the difficulty of acquisition. At test time, we have immediate access to some features but others are difficult to acquire (costly). In this limited-information setting, we strategically order questions we ask each user, tailored to previous information provided, to give the most certain predictions while minimizing the cost to users.

Results During the critical first 10 questions that our approach selects, prediction accuracy improves proportionally to fixed order approaches, but prediction certainty is higher. We can then make useful predictions at only 30% of the cost of the full-feature model.

Conclusion Dynamically ordering questions at test time allows prospective tenants to receive certain and accurate estimates for energy consumption in potential future homes, without requiring them to collect prohibitively costly amounts of detailed information about each unit.

1 Introduction

Home infrastructure impacts energy usage far more than occupant behavior (Dietz, Gardner, Gilligan, Stern, & Vandenbergh, 2009), yet most efforts to reduce energy consumption focus on behavior because infrastructure upgrades are often costly. However, this ignores the impact of choosing among potential homes. In particular, 30% of the U.S. population rent, and renters move on average every two years (U.S. Census Bureau, 2013). They can potentially choose improved infrastructure more frequently than we can expect homeowners to upgrade. A tangible measure of infrastructure quality is utility cost, often a hidden factor

in tenant cost. In addition to guiding apartment-seekers toward energy-efficient housing, utility estimates can also give prospective tenants cost-of-living estimates to help them make informed decisions about where to live. However, utility costs for apartment units are not readily available. Prospective tenants can ask landlords (or current occupants) for typical utility charges, but these estimates may not be appropriate, as occupant behavior can change energy usage by as much as 100% (Seryak & Kissock, 2003). Additionally, many prospective tenants (especially first-time renters) might not think to ask these questions.

Since energy consumption depends on home infrastructure (*e.g.*, square footage) and occupant behavior (*e.g.*, preferred temperature), we can learn the relationship between these features and energy consumption through established datasets, like the Residential Energy Consumption Survey (RECS) (U.S. Energy Information Administration, 2009). Some information can be extracted automatically from online rental advertisements, while other information must be provided by prospective tenants at various costs (for example, the question of how many windows a home has requires more effort to answer than how many people will live there). To develop a predictive model of energy consumption at training time, we begin with the extractable (*i.e.*, "free") features in a regression model to predict energy usage and use forward selection to add a subset of the costly features.

After learning this predictive model on the training dataset, the main problem lies in how to make a prediction on a new test instance. Initially, only a subset of the features (the free features) in the model are available, and asking users for each unknown value incurs a cost, depending on how hard it is to provide that feature. Our dynamic question-ordering algorithm (DQO) chooses the best question to ask next by considering which feature, if its value were known, would most reduce uncertainty, measured by the width of the prediction interval, with a penalty term on that feature's cost.

We validated our question-ordering approach on a test subset of RECS by calculating prediction error, uncertainty, and cost as questions are asked. DQO achieves similar accuracy as several fixed-order baselines after asking only 10 questions (30% of the cost of all features), but the fixed orderings do not approach DQO's certainty until 20 questions.

2 Related work

Many previous studies consider modeling residential energy usage, either to reduce consumption (*e.g.*, (Van Raaij & Verhallen, 1983)) or to inform consumers about their energy usage (*e.g.*, carbon calculators (Pandey, Agrawal, & Pandey, 2011)). However, these methods require information that prospective tenants cannot easily provide. We first review past work in modeling residential energy consumption, with emphasis on the information that is assumed to be available. Next, we cover previous work in building predictive models that reduce the cost of future predictions and in gathering information of various costs to make confident, accurate predictions on a new instance at test time.

2.1 Residential energy modeling

Prior work modeling residential energy consumption falls into two categories: top-down and bottom-up analyses (Swan & Ugursal, 2009). Top-down approaches model aggregate residential energy consumption on broad characteristics, such as macroeconomic indicators, historical climate data, and estimates of appliance ownership. For example, Haas and Schipper (1998) tested several econometric models that used aggregate residential energy demand, fuel prices, income, and heating degree days to forecast energy consumption in the residential sector. They found that an asymmetrical price elasticity model best fits the data, indicating that the reason energy demand does not surge after price decreases is because people invest in irreversible efficiency improvements—upgrading the infrastructure of their homes. Labandeira, Labeaga, and Rodríguez (2005) also used aggregate data in econometric models; they found that energy products are inelastic in Spain. In contrast, bottom-up methods model consumption at a more granular level, using features of individual households. Household features may include the macroeconomic indicators from the top-down approach, as well as occupant-specific features. For example, Douthitt (1989) examined fuel consumption for space heating in Canada, using household-specific values for occupant demographics, the housing structure, and fuel cost. Kaza (2010) used RECS to estimate energy usage from low-level housing characteristics, with a quantile regression approach to separate the effects of variables on homes with different patterns of energy consumption. However, past work has not explored the feasibility of estimating energy consumption with the

limited information available in rental advertisements, nor how to incorporate additional, costly information from occupants into the estimation.

2.2 Cost-effective feature selection and ordering

Here, "cost" refers to the costs of obtaining values for individual features, as opposed to the costs of types of errors, as in cost-sensitive learning (Elkan, 2001), or the costs of acquiring labels for instances, as in active learning (Cohn, Ghahramani, & Jordan, 1996).

In the supervised learning setting, a training set of samples (each with D features) with labels is available, and the goal is to learn a function that maps features to labels. Feature selection, choosing $d < D$ relevant features, improves prediction performance by reducing both overfitting and feature-acquisition costs for test instances (Guyon & Elisseeff, 2003). Many feature selection algorithms do not incorporate feature-specific costs, but such cost measures can represent the true cost of feature sets.

After learning a predictor on a subset of d cost-effective features, making a prediction on a test instance requires gathering feature values, which can be costly, especially if it requires cooperation from users who might stop before completion. In this case, strategically ordering questions asked (based on previous answers) can get the most useful information first, while providing predictions on partial information. This way, people receive meaningful predictions without spending much time or effort answering questions.

The test-time feature ordering problem resembles active learning, which assumes *labels* are expensive. Active learning algorithms strategically select which unlabeled points to query to maximize the model's performance (using both labeled and unlabeled data) while minimizing the cost of data collection (Cohn et al., 1996). Test-time feature ordering has a similar goal of making accurate predictions while keeping data collection costs low; however, rather than choosing an *example to be labeled*, test-time feature ordering chooses a *single feature to be entered* (by asking the user a question).

2.2.1 Training-time feature selection

Several algorithms incorporate unique feature costs into supervised learning by penalizing the benefit of adding a feature by the cost of that feature. For example, Davis, Ha, Rossbach, Ramadan, and Witchel (2006) develop a cost-sensitive decision tree algorithm by modifying the ID3 splitting criterion, maximizing information gain (Quinlan, 1986), to penalize feature costs. They apply this method to forensic classification, where events cannot be reproduced and so all features need to be acquired before classification; cost refers to computational overhead for features. Saeedi, Schimert, and Ghasemzadeh (2014) take a related approach to build a low-cost classifier for sensor localization: a greedy algorithm selects the best feature, in terms of contribution to prediction performance and power consumption, and adds it to the model until accuracy meets a threshold; cost in their setting reflects power consumption.

2.2.2 Test-time feature ordering

At test time, the goal is to make a prediction on a new example. Some features are initially available, but others must be acquired at cost. Ordering the features to ask, based on known values, can improve prediction while keeping feature acquisition costs low. He, Daumé III, and Eisner (2012) consider a similar setting, where all features are available for training, and at test time they want an instance-specific subset of features for prediction, trading off feature cost with prediction accuracy. They formulate dynamic feature selection as a Markov decision process. The policy selects a feature to add; the reward function reflects the classifier margin with the next feature, penalized by the cost of including that feature. However, this method does not make sequential predictions, and instead only chooses whether to keep getting features or to stop and make a final prediction.

Dynamically ordering questions in a survey, based on previously-answered questions, can be considered an adaptive survey design (ASD). ASD attempts to improve survey quality (higher response rate, lower error) by giving respondents custom survey designs, rather than the same one (Schouten, Calinescu, Luiten, et al., 2013). Usually ASD tries to minimize nonresponse, and designs involve factors like number of follow-ups, which can be costly. The general technique is to maximize survey quality, while keeping costs below a budget.

Another related area is personalization, where a system suggests items from user preferences. However, the cold-start problem makes it hard to give recommendations at first, when the system knows nothing about the user, including which questions to ask. Sun et al. (2013) present a multiple-question decision tree for movie recommendation, where each tree node asks users for opinions on several movies, rather than just one. This model lets users sooner provide information about movies they have seen, but it is designed to minimize the number of questions and not the amount of effort required to answer questions of varying difficulty.

Most work in test-time feature ordering does not consider the situation of providing predictions with partial information *as* questions are answered, nor does it address the issue of giving measures of prediction uncertainty to users. Our work fills this gap to provide personalized energy estimates with limited, costly information to prospective tenants.

3 Problem and approach

We address two problems: (1) at training time, how to build a model that efficiently uses features of various costs, and (2) at test time, how to order costly questions we ask a user so that we can make confident predictions on limited information about a new instance. At training time, we use a cost-aware feature selection approach that adds higher-cost features to a regression model to predict household energy consumption. For making a prediction on a new test point, we develop a novel cost-effective dynamic question-ordering algorithm that sequentially chooses features to ask according to how much they are expected to reduce the uncertainty of the next prediction, penalized by the cost of each feature.

4 Method

4.1 Training time: cost-aware feature selection

A greedy approximation to feature selection, forward selection starts with an empty feature set and, at each iteration, adds the feature that minimizes error (Harrell, 2001; Tropp, 2004). For this analysis, we started with the free extractable features (rather than no features, as in classic forward selection) and minimized leave-one-out cross-validation error with linear regression to add successive higher-cost features.

4.2 Test time: cost-effective dynamic question ordering

After learning regression models for energy usage on the selected features from our training set, we want to make a prediction on a new test point, where initially only some features of the model are available. Our approach considers a trajectory of prediction intervals as a user provides information. A prediction interval consists of a lower and upper bound such that the true value lies in this interval with at least some probability (Weisberg, 2014). Prediction interval width corresponds to prediction uncertainty: a wider interval means less confidence. We select as the optimal next question the one whose inclusion most reduces the expected value of the prediction interval width; that is, it most reduces the expected uncertainty of the next prediction.

In this problem, there are features that are unknown (not yet supplied by the user). We use k nearest neighbors (kNN) (Cover & Hart, 1967) to supply values for unanswered features in vector $x \in \mathbb{R}^d$. For each unknown feature f, we find the k data points in the training set $X \in \mathbb{R}^{n \times d}$ (n samples, each d-dimensional) that are closest to x, along the dimensions $\mathcal{K}$ that are currently known. Then we estimate x_f as z_f, the mean or mode, as appropriate, of feature f in the k nearest neighbors (depending whether the feature is continuous or discrete) (see Algorithm 1).

Because these z values estimate unknown features of x, we use the measurement error model (MEM) (Fuller, 2009) to capture error associated with estimated features. Unlike traditional regression models, MEMs do not assume we observe each component x_f exactly; there is an error δ_f associated with the estimation:

$$z_f = x_f + \delta_f, \text{ where } \mathbb{E}[\delta_f | x_f] = 0.$$

Prediction $\hat{y}$ still depends on the *true, unobserved* value x:

$$\hat{y} = \hat{\beta}^T \bar{x} = \hat{\beta}(\bar{z} + \bar{\delta}),$$

Algorithm 1 Estimating values z for still-unknown features

Input: $X \in \mathbb{R}^{n \times d}, x \in \mathbb{R}^d, \mathcal{K} \subseteq \{1, ..., d\}, k \in \mathbb{Z}^+$
Output: $z \in \mathbb{R}^d$

function ESTIMATE_FEATURES($X, x, \mathcal{K}, k$)
  $z_{\mathcal{K}} \leftarrow x_{\mathcal{K}}$ ▷ Copy over the known features
  $\mathcal{I} \leftarrow$ get_knn($X_{:,\mathcal{K}}, z_{\mathcal{K}}, k$) ▷ Index $z_{\mathcal{K}}$'s kNNs
  for $f \in \{1, ..., d\} \setminus \mathcal{K}$ **do** ▷ For all unknown f
    $z_f \leftarrow \text{mean}(X_{\mathcal{I},f})$ ▷ Estimate z_f from kNNs' values for feature f
  end for
  return z
end function

where $\hat{\beta} \in \mathbb{R}^{d+1}$ is the parameter vector learned on the training set X (recall all feature values are known at training time). The notation $\bar{x}, \bar{z}, \bar{\delta}$ means vectors x, z have a 1 appended to them and δ a 0 to account for the constant term in the regression. Let $\bar{X}$ extend this notion to the training matrix: $\bar{X} = [\mathbb{1}^n X]$.

Recall that $\text{Var}(\hat{\beta}) = \sigma^2(\bar{X}^T\bar{X})^{-1}$, where σ^2 is the variance of the regression error. Then, we can calculate the variance between the true value y and the estimated value $\hat{y}$ as

$$\begin{aligned}
\text{Var}(y - \hat{y}) &= \text{Var}(y - \hat{\beta}^T(\bar{z} + \bar{\delta})) \\
&= \text{Var}(y) + \text{Var}(\hat{\beta}^T(\bar{z} + \bar{\delta})) \\
&= \sigma^2 + \sigma^2\bar{z}^T(\bar{X}^T\bar{X})^{-1}\bar{z} + \sigma^2\bar{\delta}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta} \\
&= \sigma^2\left(1 + \bar{z}^T(\bar{X}^T\bar{X})^{-1}\bar{z} + \bar{\delta}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta}\right).
\end{aligned}$$

Because the unbiased estimator of σ^2 is $\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-(d+1)}$,

$$\begin{aligned}
T &= \frac{y - \hat{y}}{\sqrt{\frac{\sum(y_i - \hat{y}_i)^2}{n-d-1}\left(1 + \bar{z}^T(\bar{X}^T\bar{X})^{-1}\bar{z} + \bar{\delta}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta}\right)}} \\
&\sim t_{n-d-1}.
\end{aligned}$$

Therefore, we can calculate a $100(1-\alpha)\%$ prediction interval for a new point z as

$$\hat{y} \pm t_{n-d-1;\alpha/2} \cdot \sqrt{\hat{\sigma}^2\left(1 + \bar{z}^T(\bar{X}^T\bar{X})^{-1}\bar{z} + \bar{\delta}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta}\right)}, \tag{1}$$

where the $\bar{\delta}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta}$ term accounts for error from estimated features and $t_{n-d-1;\alpha/2}$ is the value at which a Student's t distribution with $n-d-1$ degrees of freedom has cumulative distribution function value $\alpha/2$. We can estimate δ from training data by calculating the error of predicting each feature with kNN, from the other features. We also estimate $\hat{\sigma}^2$, the regression variance, from training data.

Then, we cycle through each candidate feature f and compute the expected prediction interval width $\mathbb{E}[w(f)]$ for asking that feature next, over each value r that feature f might take on from its range of potential values R:

$$\mathbb{E}[w(f)] = 2 \cdot t_{n-d-1;\alpha/2} \sum_{r \in R} p(z_f = r) \cdot \sqrt{\hat{\sigma}^2\left(1 + \bar{z}_{f:=r}^T(\bar{X}^T\bar{X})^{-1}\bar{z}_{f:=r} \ + \bar{\delta}_{f:=0}^T(\bar{X}^T\bar{X})^{-1}\bar{\delta}_{f:=0}\right)}, \tag{2}$$

where $p(z_f = r)$, the probability that the f-th feature's value is r, is calculated empirically from the training set, and the notation $\bar{u}_{f:=q}$ means the f-th component of u is replaced with the value q. Algorithm 2 writes this process in pseudocode. In this algorithm, feat_ranges and feat_proportions are both d-dimensional cell arrays where the f-th cells contain, respectively, the set of values R that feature f can take on and the proportions $p \in \mathbb{R}^{|R|}$ that each value $r \in R$ appears in the training set. The output, $E \in \mathbb{R}^d$, is a vector where the f-th component is the expected prediction interval width if the value of feature f were known.

Algorithm 2 Calculating the expected prediction interval width for each candidate feature to be asked

Input: $X \in \mathbb{R}^{n \times d}$, $\mathcal{K} \subseteq \{1, ..., d\}$, $z \in \mathbb{R}^d$, $\delta \in \mathbb{R}^d$,
$\hat{\sigma}^2 \in \mathbb{R}$, $\alpha \in [0, 1]$, feat_ranges, feat_proportions
Output: $E \in \mathbb{R}^d$

function EXPECTED_INTERVAL_WIDTH(X,$\mathcal{K}$, z, δ, $\hat{\sigma}^2$, α, feat_ranges, feat_proportions)
  $v \leftarrow \mathbb{0}^d$
  $\bar{X} \leftarrow [\mathbb{1}^N X]$
  for $f \in \{1, ..., d\} \setminus \mathcal{K}$ **do**
    $R \leftarrow$ feat_ranges$\{f\}$, $u \leftarrow \mathbb{0}^{|R|}$
    for $\ell \in \{1, ..., |R|\}$ **do**
      $\bar{z} \leftarrow [1; z]$, $\bar{z}_{f+1} \leftarrow R_\ell$
      $\bar{\delta} \leftarrow [0; \delta]$, $\bar{\delta}_{f+1} \leftarrow 0$
      $u_\ell \leftarrow \bar{z}^T (\bar{X}^T \bar{X})^{-1} \bar{z} + \bar{\delta}^T (\bar{X}^T \bar{X})^{-1} \bar{\delta}$
    end for
    $p \leftarrow$ feat_proportions$\{f\}$
    $v_f \leftarrow p^T u$
  end for
  return $E \leftarrow 2 \cdot t_{n-d-1;\alpha/2} \sqrt{\hat{\sigma}^2 (1 + v)}$
end function

Algorithm 3 Dynamically choosing a question ordering $\mathcal{A}$ and making a sequence of predictions $\hat{y}$ at the current feature values and estimates as feature values are provided

Input: $X \in \mathbb{R}^{n \times d}$, $x \in \mathbb{R}^d$, $\mathcal{K} \subseteq \{1, ..., d\}$, $k \in \mathbb{Z}^+$,
$\delta \in \mathbb{R}^d$, $\alpha \in [0, 1]$, feat_ranges, feat_proportions,
$\hat{\beta} \in \mathbb{R}^{d+1}$, $\hat{\sigma}^2 \in \mathbb{R}$, $\lambda \in \mathbb{R}$, $c \in \mathbb{R}^d$
Output: $\mathcal{A} \subseteq \{1, ..., d\}, \hat{y} \in \mathbb{R}^{|\mathcal{A}+1|}$

function DQO_ALL(X, x, $\mathcal{K}$, k, δ, α, feat_ranges, feat_proportions, $\hat{\beta}$, $\hat{\sigma}^2$, λ, c)
  $\mathcal{A} \leftarrow \{\}$, $\hat{y} \leftarrow \{\}$
  for $i \in \{1, ..., d - |\mathcal{K}|\}$ **do**
    $z \leftarrow$ ESTIMATE_FEATURES$(X, x, \mathcal{K}, k)$
    $\hat{y}_i \leftarrow \hat{\beta}^T z$ ▷ Predict on features and estimates
    $E \leftarrow$ EXPECTED_INTERVAL_WIDTH(X, $\mathcal{K}$, z, δ, $\hat{\sigma}^2$, α, feat_ranges, feat_proportions)
    $f^\star \leftarrow \arg\min_{f \notin \mathcal{K}} (E_f + \lambda \cdot c_f)$
    $\mathcal{A} \leftarrow \mathcal{A} \cup \{f^\star\}, \mathcal{K} \leftarrow \mathcal{K} \cup \{f^\star\}$
    $z_{f^\star} \leftarrow x_{f^\star}$ ▷ Ask and receive value for $f^\star$
    $\delta_{f^\star} \leftarrow 0$ ▷ No more uncertainty in $f^\star$
  end for
  $z \leftarrow$ ESTIMATE_FEATURES$(X, x, \mathcal{K}, k)$
  $\hat{y}_{d-|\mathcal{K}|+1} \leftarrow \hat{\beta}^T z$ ▷ Make final prediction
  return $\mathcal{A}, \hat{y}$
end function

Including the feature that attains the narrowest expected prediction interval width $\mathbb{E}[w(f)]$ will reduce the uncertainty of our prediction more than any other feature. This approach allows incorporation of feature cost into the question selection, by weighting the expected prediction interval width against the cost of acquiring the feature:

$$f^\star = \arg\min_f (\mathbb{E}[w(f)] + \lambda \cdot c_f),$$

where c_f is the cost of feature f and $\lambda \in \mathbb{R}$ trades off feature cost with reduced uncertainty. A high-cost feature might not be chosen, if another feature can provide enough improvement at lower cost. We ask for this information, update our vector of known data with the response (and estimate the unknown features

again, now including the new feature in the set for kNN prediction), and repeat the process until all feature values are filled in (or the user stops answering). Algorithm 3 formalizes this dynamic question-ordering (DQO) process.

More generally, this algorithm can be seen as a framework that makes predictions on partial information and selects which feature to query next by 1) estimating values for unknown features (here with kNN) and 2) asking for the feature that will most reduce the expected uncertainty of the next prediction (here measured by prediction interval width). With this approach, we strategically order questions, tailored to previous information, to give accurate predictions while minimizing the user burden of answering many or difficult questions that will not provide a substantial reduction in prediction uncertainty.

5 Data

The Residential Energy Consumption Survey (RECS) contains information about home infrastructure, occupants, and energy consumption. We can use this dataset to learn relationships between household features and energy consumption to predict energy usage for prospective tenants. The most recently released RECS was a nationally representative sample of 12,083 homes across the U.S. (U.S. Energy Information Administration, 2009). For each household, RECS records fuel consumption by fuel type (*e.g.*, electricity, natural gas) and around 500 features of the home (*e.g.*, age of refrigerator, number of occupants).

5.1 Defining feature costs

In our problem setting, features have different costs of obtaining, and we want to build models and make predictions that leverage features cost-effectively. Some information is easily found in the rental listing (*e.g.*, number of bedrooms) and is therefore "free," while other information requires asking users. For example, the number of windows does not appear in listings and would require a prospective tenant to visit each site; consequently, this question has high cost. Other useful features relate to occupant behavior (*e.g.*, preferred temperature). These questions likely remain constant for each user across homes and therefore require asking only once and are cheaper. We assign the cost of each feature as one of eight categories, depending on how difficult it is to acquire a value for that feature. Table 1 lists the cost categories and examples of RECS features in each category. Category 0 indicates that the feature can be extracted from a rental listing, categories 1 and 2 relate to occupant behavior, and categories 3 through 7 are unit-related features that cannot be extracted from a rental listing. Table 2 lists the information used for extractable (*i.e.*, category 0) features and how often it appears in Rent Jungle, a company that scrapes rental listings from the internet; these "free" features appear in the majority of listings on Rent Jungle.

Table 1: Feature cost reflects how difficult it is to acquire a value for a feature. Note that the cost of a feature might vary by home: for example, if a listing included pictures of the kitchen, a user could see the door arrangement of the refrigerator from the listing (cost 3). If there were no pictures, then it would be an easily-visible feature (cost 5).

Cost	**Method of answering**	**Example feature from RECS**
0	Extractable from listing	Number of bedrooms
1	User probably already knows	If someone stays home during the day
2	Might have to check current home	Size of TV
3	Could find in rental listing, or call for easy answer	Washing machine in home
4	Look up online	Year housing unit was built
5	Easily visible during visit	High ceilings
6	Requires more effort to find out during visit	Type of glass in windows
7	Requires visit + looking something up	Age of heating equipment

Table 2: The features we define as extractable (*i.e.*, "free") appear in most of the listings on Rent Jungle. Geographic features associated with the city, zip code, or state include climate zone and whether the area is urban or rural, among others.

Feature	**Presence in Rent Jungle database**
Number of bedrooms	85%
Number of full bathrooms	57%
Studio apartment	85%
City or zip code	99%
State	100%

6 Experimental validation

We validated our training-time feature selection and test-time feature ordering approaches on the RECS dataset for predicting household electricity and natural gas consumption. We restricted our analysis to homes in the same climate zone as our planned deployment location in Pittsburgh, a subset of 2470 households in climate zone 2. We used 90% of these homes for training and the remaining 10% for testing. The training set was further subdivided into feature selection and cross-validation subsets. We trained separate models for predicting electricity consumption (on all homes in our climate zone) and for predicting natural gas consumption (on the 75% of homes that use natural gas).

6.1 Training time: cost-aware feature selection

We can reasonably assume all zero-cost features are available (since these can be extracted from the rental listing). However, for higher-cost groups, we need to know how many features are needed in practice. We analyzed the predictive power of selecting features based on cost. To do a simple comparison of the added benefit of including higher-cost features into a regression model, we simplified the eight-category feature cost from Table 1 into three categories: free features, occupant-related features, and unit-related features. The 16 features in category 0 can be automatically extracted from online rental listings and are therefore free. Categories 1 and 2 (132 features total) relate to the occupants living in a home and their habits. These features are mid-cost because people can answer questions about their household relatively easily, and these features will likely remain constant regardless of where someone lives (*e.g.*, an occupant will probably have the same preferred temperature in any home). Categories 3 through 7 (312 features total) relate to the unit and require effort on the part of the user to answer. These features are high cost because a person does not know this information for an apartment already (unlike occupant-related features), and so they need to look something up or make a site visit to assess in-person. Furthermore, these features need to be entered for every apartment a user is considering.

Figure 1 summarizes the fraction of explained variance for these feature groups, on the 20% of the training set used for feature selection. Although extractable (cost 0) features explain only 21-34% of variance, the full feature set (costs 0-7) explains 99-100% of the variance.

Figure 2 shows how the degree of variance explained changes as higher-cost features are added in forward selection. The steep trajectory means the first few features added have a big impact on the explained variance, with about 10 costlier features needed to explain 60% of the variance, and 28 costlier features to explain 70% of the variance of electricity usage, on the feature selection subset. Natural gas performance is similar to electricity, as shown in Figure 2.

Based on this analysis, we chose 60 features (30 each for electricity and natural gas), plus the 16 low-cost features. We selected 30 for each prediction task both because our exploration demonstrates that that many can explain a substantial amount of the variance, and also to avoid overfitting in our final model from having too many features. Table 3 gives examples of features that are most predictive for electricity and natural gas prediction.

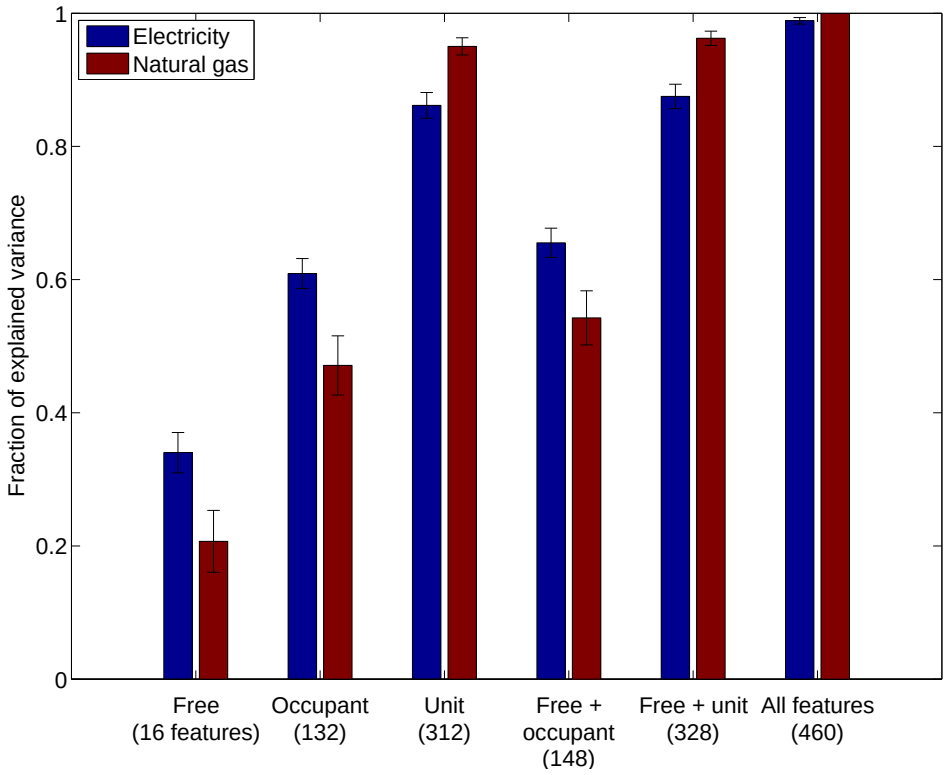


Figure 1: Different cost levels of features provide different prediction performances, measured by fraction of variance captured by a regression model (values closer to one indicate better performance). Occupant features are "mid-cost" (costs 1-2) and unit features are "high-cost" (costs 3-7).

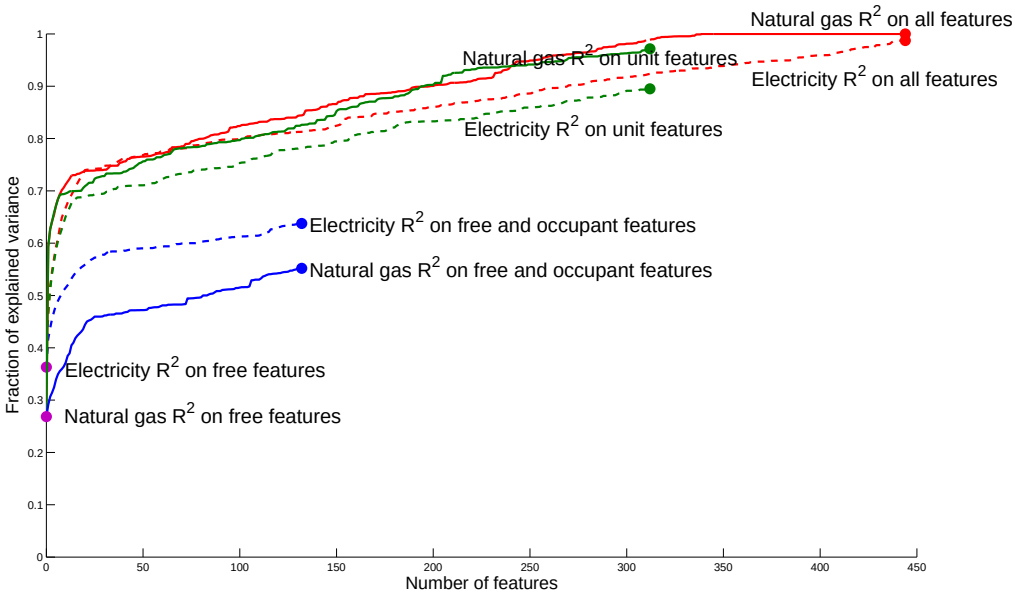


Figure 2: The fraction of explained variance for electricity (dashed) and natural gas (solid) usage as features are added in forward selection, starting with the extractable, free (cost 0) features. Including 30 higher-cost features for electricity and natural gas prediction can explain most of the variance of energy usage.

Table 3: Some features selected via forward selection for predicting electricity and natural gas consumption.

Electricity	**Natural gas**
Temperature at night in summer	Temperature at night in winter
Energy-star refrigerator	Number of rooms heated
Number of household members	Year housing unit was built
Number of computers used	Water heater size

We then used tenfold cross-validation on the remaining 80% of the training set to learn regression coefficients for the chosen features. Figure 3 summarizes performance of our final predictor on a held-out test set, as well as on the feature-selection training set and the parameter-selection cross-validation set.

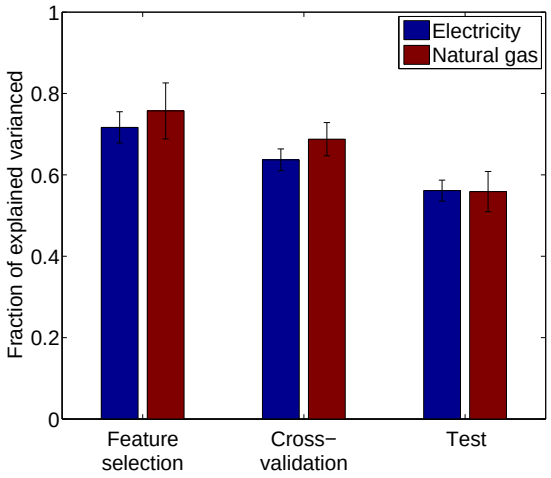


Figure 3: Our final predictor explains 56% of the variance on a held-out test set (values closer to one indicate better performance).

6.2 Test time: cost-effective dynamic question ordering

After learning regression models for electricity and natural gas prediction on the training set, we want to make predictions of energy usage for a new test point. We use our dynamic question-ordering framework (DQO) to make sequential predictions with partial, evolving information. To apply DQO, we first used the training set to choose parameters for imputing unknown features (k, from Algorithm 1, ESTIMATE_FEATURES) and to estimate the measurement error δ (from Algorithm 2, EXPECTED_INTERVAL_WIDTH). Then, we simulated question-asking on RECS to evaluate the performance of DQO for test-time feature ordering.

6.2.1 Parameter selection and estimation

We used the training set to choose $k = 100$ for imputing the values of features that have not yet been asked, based on the prediction performance of kNN for the higher-cost features. Then, we estimated the measurement error δ_f for each feature as the error from kNN on the training set.

6.2.2 Simulating the question-asking and -answering process with RECS

We simulated the process of asking and answering questions on the testing subset of RECS by hiding the values for features that were not yet known. After we used DQO to choose a feature to acquire (line 7 in

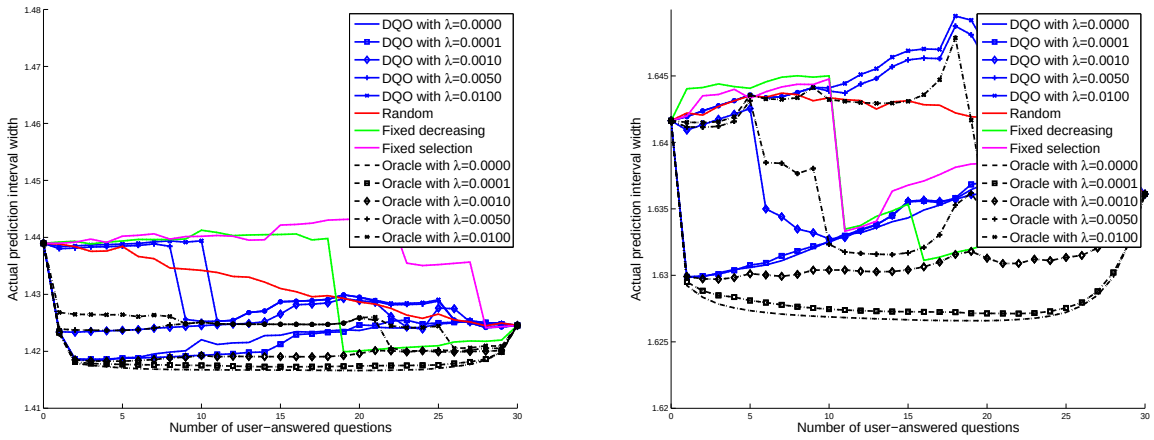
Algorithm 3, DQO_ALL), we "asked" this question and unveiled its value (line 9 in Algorithm 3) once it was "answered."

6.2.3 Evaluating DQO performance with prediction certainty, error, and cost

We evaluated the performance of our cost-effective *DQO* algorithm in making sequential predictions with partial, evolving information on a held-out test set. For comparison, we implemented several baselines. The *Oracle* chooses the next best feature according to the minimum true prediction interval width (calculated on the test sample using true feature values, rather than the *expected* width as in Algorithm 3). We tested two versions of DQO and oracle: ordering additional features *without cost* and *with cost* (implemented as $\lambda = 0$ and $\lambda > 0$). In addition, we implemented a *Random* algorithm which chooses a random question ordering for each sample; a *Fixed Decreasing* algorithm, which asks questions in decreasing order of feature measurement error δ_f (identical ordering for all samples); and a *Fixed Selection* algorithm, which asks questions in the order of forward selection in the training phase (also identical for all samples).

We calculated several metrics related to the trajectory of prediction performance and cost, for orderings given by the seven algorithms: *DQO* and *Oracle with* and *without cost, Random, Fixed Decreasing*, and *Fixed Selection*. We summarized prediction performance with the width of the current prediction interval (prediction *certainty*) and the absolute value of the difference between the current prediction and the truth (prediction *error*); we also measured the cumulative cost of all features asked at each step (prediction *cost*). We report on performance for two separately trained regressions (for electricity and for natural gas). Table 4 summarizes the metric trajectories for electricity and natural gas as areas under the curve—smaller values are better because they mean the algorithm spent less time in high uncertainty, error, and cost.

Certainty metrics For certainty, we calculated widths of 90% prediction intervals as features were answered. Since narrower prediction interval widths correspond to more certain predictions, we expect DQO interval widths to be less than those of the baselines, particularly in the early stages. Figure 4 plots the *actual* prediction interval widths as questions are asked (calculated with Equation 1, using the true known feature values and imputed values for unknown features), averaged across the test dataset, for the question sets from the seven orderings. The DQO sets result in the narrowest (or near-narrowest) prediction intervals (*i.e.*, most certain predictions), compared to the baselines, with improvements most notable in the first 10 questions answered—the situation that arises when users do not answer all the questions.

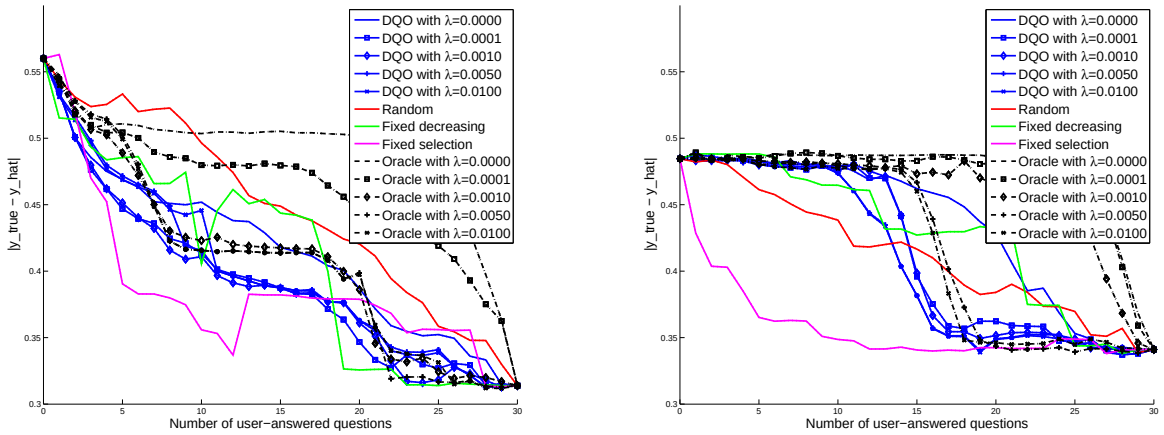


(a) Electricity.

(b) Natural gas.

Figure 4: Prediction interval widths as questions are asked: DQO results in more certain predictions (*i.e.*, lower prediction interval widths) than the baseline orderings.

Error metrics For error, we calculated the absolute value of differences between predictions and truth as questions are answered, plotted in Figure 5. For all orderings, predictions approach the true value as questions are answered. Once about 10 questions have been asked, *DQO with cost* reaches similar performance as *DQO without cost* and the fixed-order baselines (*Fixed decreasing* and *Fixed selection*).

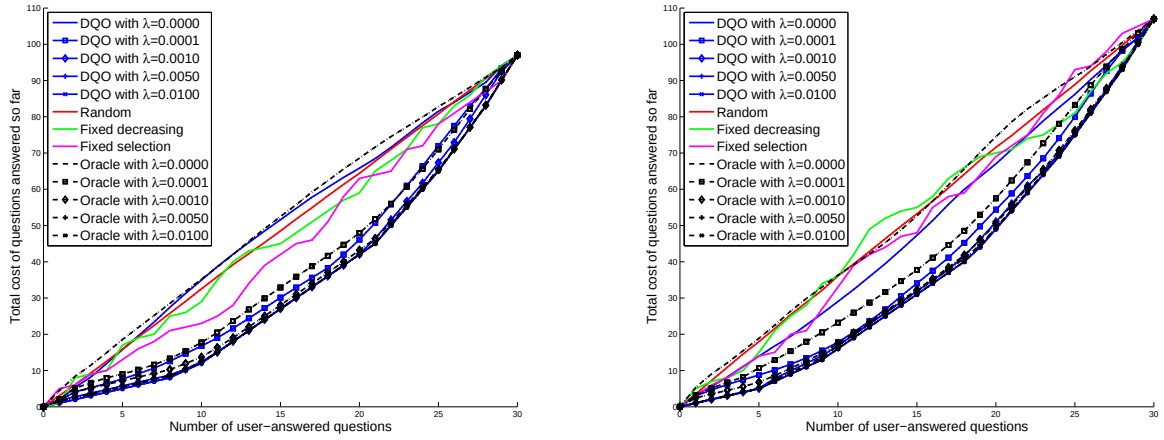


(a) Electricity.

(b) Natural gas.

Figure 5: Mean absolute error as questions are asked: DQO results in similarly-correct predictions as baselines.

Cost metrics Progressive total feature costs as features are asked and their true values are used in the models are plotted in Figure 6. Cumulative feature costs are lower for orderings that penalize feature cost (*DQO, Oracle with cost*), with cost decreasing as the penalty on cost λ increases. The other orderings have similar cost trajectories to each other.



(a) Electricity.

(b) Natural gas.

Figure 6: Total feature costs as questions are asked: the tradeoff parameter λ influences when expensive features are included.

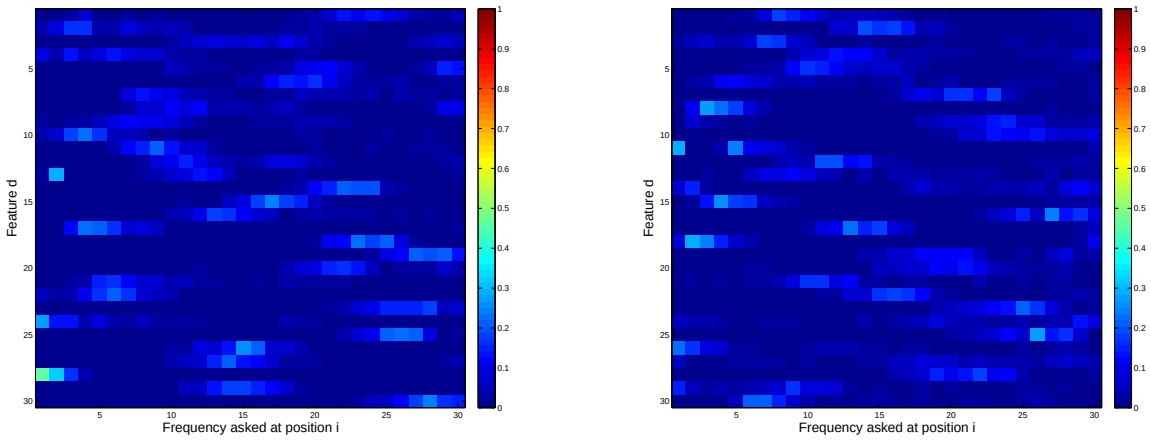
Overall, these metrics show that our test-time DQO approach quickly achieves accurate, confident predictions: by asking around 10 questions, DQO (with and without cost) reaches similar accuracy as the

fixed-order baselines, but the sequential predictions by the fixed orderings are less confident than DQO until about 20 questions have been asked.

Table 4: Areas under the curve for the certainty, error, and cost metrics from various methods, for electricity and natural gas prediction: smaller values mean the algorithm spent less time in high uncertainty, error, and cost.

Method	**Electricity**			**Natural gas**		
	Int. width	$\lvert y - \hat{y} \rvert$	*Cost*	*Int. width*	$\lvert y - \hat{y} \rvert$	*Cost*
DQO with $\lambda = 0.0000$	42.68	12.48	1485.80	49.04	13.07	1483.46
DQO with $\lambda = 0.0001$	42.67	11.76	1102.23	49.04	12.44	1240.19
DQO with $\lambda = 0.0010$	42.77	11.73	1013.78	49.10	12.40	1162.26
DQO with $\lambda = 0.0050$	42.91	11.99	988.34	49.29	12.21	1137.07
DQO with $\lambda = 0.0100$	42.94	12.02	987.56	49.31	12.20	1135.63
Random	42.53	14.72	1532.40	48.83	14.24	1639.48
Fixed decreasing	42.55	13.86	1134.31	48.85	14.19	1329.16
Fixed selection	42.59	12.28	1022.47	48.93	13.79	1176.79
Oracle with $\lambda = 0.0000$	42.71	12.23	994.78	49.06	12.58	1148.58
Oracle with $\lambda = 0.0001$	42.75	12.27	992.52	49.20	12.55	1137.76
Oracle with $\lambda = 0.0010$	42.83	12.40	988.96	49.21	12.55	1137.07
Oracle with $\lambda = 0.0050$	42.85	12.39	988.32	49.24	12.56	1136.22
Oracle with $\lambda = 0.0100$	42.89	12.42	987.65	49.27	12.55	1135.51

Figure 7 shows how frequently the oracle asked each feature in each position across test instances. Most features are chosen fairly uniformly at each position in the question ordering. This indicates that there is no single best order to ask questions across all households, which is why the dynamic question-ordering process is so valuable.



(a) Electricity.

(b) Natural gas.

Figure 7: The oracle chose to add features fairly uniformly across test samples, shown here as the frequency each feature was asked in each potential position of question orders.

7 Limitations

Currently, our DQO algorithm assumes that 1) users are able to answer the next question we ask and 2) their answers are accurate. However, situations could arise where these assumptions do not hold. For example, in the utility prediction task, a prospective tenant may be interested in getting personalized energy estimates for a home before visiting—they could still answer occupant-related features. DQO can be easily extended to this case by offering users a "don't know" option for answering questions and removing unknown features from consideration in later iterations. Breaking the second assumption, that user answers are accurate, would allow people to give estimates for features (*e.g.*, refrigerator size by looking at pictures in the rental listing). Incorporating this element into DQO would require a way to estimate error associated with user-provided feature estimates.

Furthermore, we have not yet tested the sequential question-asking and prediction-providing process of DQO with human users. We hypothesize that giving users estimates from partial information will motivate them to continue answering questions to receive more accurate personalized predictions.

8 Future Work

Expansions to this work include, in increasing order of generalization, refining feature costs to reflect the influences of crowdsourcing and context when collecting apartment-related data, generalizing the DQO framework to other methods for classification rather than regression, and taking a DQO-like approach to adaptive survey design.

8.1 Incorporating the effects of crowdsourcing and context into the cost model for personalized energy predictions

Here, we assumed that unit-related features (cost 3-7) were more costly than occupant-related features (cost 1-2), because they might require a user to visit an apartment to assess (*e.g.*, to count the number of windows). However, once enough people are looking at the same apartment, crowdsourcing could lower the effective cost for these unit-related features, since one apartment hunter's result could be shared among all potential occupants. Unit-level information crowdsourced in this way would likely remain relevant for a fairly long time, since feature values will change only when the landlord upgrades the property.

Additionally, unit-related features might also be less costly if we know the context in which a prospective tenant is answering questions. For example, if the user is currently viewing an apartment, it becomes much less expensive to ask them to gather information related to that unit since they are already on site.

8.2 Generalizing the DQO framework to other techniques

DQO as presented here relies on using kNN to estimate values for unknown features and calculating expected prediction uncertainty with prediction interval widths to choose the next question to ask. However, we can use other techniques for estimation of yet-unasked questions and calculations of expected uncertainty. For example, to do classification instead of regression, we could use distance from the decision boundary as a measure of certainty for SVM classification (farther away from boundary means a more certain prediction) or the probability the test point is in a particular class for decision trees, logistic regression, or naive Bayes classification.

8.3 Broadening the scope of DQO to adaptive survey design

Dynamically ordering questions can also be beneficial to survey design. Falling survey response rates lead to results that do not represent the full population (Porter, 2004), and response rates are often worse for online surveys than for mail surveys (Shih & Fan, 2008). However, unlike traditional paper surveys, online surveys can support adaptive questions, where later questions that are asked depend on responses given to previous questions. Past work in adaptive questions for online surveys has taken a rule-based, question-specific approach that means a certain response to a certain question leads to a new set of questions, uniformly across all respondents (Pitkow & Recker, 1995; Bouamrane, Rector, & Hurrell, 2008). A richer interpretation

of adaptive questions would use a dynamic question order, personalized to the individual respondent, based on their previous answers. Such an approach could increase engagement, and therefore response rate, as well as the quality of imputations for missing data.

9 Conclusion

Providing personalized energy estimates to prospective tenants with limited, costly information is a challenge. Our solution uses an established dataset to build cost-effective predictive models and, at test time, dynamically orders questions for each user. At training time, we use a cost-based forward selection algorithm to select relevant features from RECS and combine low-cost features that are extractable from rental advertisements with relevant higher-cost features related to occupant behavior and home infrastructure. At test time, when we want to make a personalized estimate for a new renter-home pair, we present a cost-effective way to choose questions to ask a user about their habits and a rental unit, based on which feature's inclusion would most improve the certainty of our prediction, given the information we already know. Our experiments show that, for predicting electricity and natural gas consumption, we achieve prediction performance that is equally accurate, but more certain, than two fixed-order baselines by asking users only 25% of features (30% of the cost of the full-feature model). This setting, where we know all feature values at training time but must acquire individual features at cost during test time, shows up in other applications, such as conducting tests to make a medical diagnosis, giving personalized recommendations, and administering adaptive surveys. Applying DQO to these areas can give people valuable predictions without unduly burdening them for additional information.

References

Bouamrane, M.-M., Rector, A., & Hurrell, M. (2008). Gathering precise patient medical history with an ontology-driven adaptive questionnaire. In *21st IEEE International Symposium on Computer-Based Medical Systems* (pp. 539–541).

Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active learning with statistical models. *Journal of Artificial Intelligence Research*.

Cover, T. M., & Hart, P. E. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27.

Davis, J. V., Ha, J., Rossbach, C. J., Ramadan, H. E., & Witchel, E. (2006). Cost-sensitive decision tree learning for forensic classification. In *Machine Learning: ECML 2006* (pp. 622–629). Springer.

Dietz, T., Gardner, G. T., Gilligan, J., Stern, P. C., & Vandenbergh, M. P. (2009). Household actions can provide a behavioral wedge to rapidly reduce US carbon emissions. In *Proceedings of the National Academy of Sciences* (Vol. 106, pp. 18452–18456). National Academy of Sciences.

Douthitt, R. A. (1989). An economic analysis of the demand for residential space heating fuel in Canada. *Energy*, *14*(4), 187–197.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (pp. 973–978).

Fuller, W. A. (2009). *Measurement error models*. John Wiley & Sons.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *The Journal of Machine Learning Research*, *3*, 1157–1182.

Haas, R., & Schipper, L. (1998). Residential energy demand in OECD-countries and the role of irreversible efficiency improvements. *Energy Economics*, *20*(4), 421–442.

Harrell, F. E. (2001). *Regression modeling strategies*. Springer Science & Business Media.

He, H., Daumé III, H., & Eisner, J. (2012). Cost-sensitive dynamic feature selection. In *ICML Inferning Workshop*.

Kaza, N. (2010). Understanding the spectrum of residential energy consumption: a quantile regression approach. *Energy Policy*, *38*(11), 6574–6585.

Labandeira, X., Labeaga, J. M., & Rodríguez, M. (2005). A residential energy demand system for Spain. *MIT Center for Energy and Environmental Policy Research Working Paper*(2005-001).

Pandey, D., Agrawal, M., & Pandey, J. S. (2011). Carbon footprint: current methods of estimation. *Environmental Monitoring and Assessment*, *178*(1-4), 135–160.

Pitkow, J. E., & Recker, M. M. (1995). Using the web as a survey tool: Results from the second WWW user survey. *Computer Networks and ISDN Systems*, *27*(6), 809–822.

Porter, S. R. (2004). Raising response rates: What works? *New Directions for Institutional Research*, *2004*(121), 5–21.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106.

Saeedi, R., Schimert, B., & Ghasemzadeh, H. (2014). Cost-sensitive feature selection for on-body sensor localization. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication* (pp. 833–842).

Schouten, B., Calinescu, M., Luiten, A., et al. (2013). Optimizing quality of response through adaptive survey designs. *Survey Methodology*, *39*(1), 29–58.

Seryak, J., & Kissock, K. (2003). Occupancy and behavioral effects on residential energy use. In *Proceedings of the Solar Conference* (pp. 717–722).

Shih, T.-H., & Fan, X. (2008). Comparing response rates from web and mail surveys: A meta-analysis. *Field Methods*, *20*(3), 249–271.

Sun, M., Li, F., Lee, J., Zhou, K., Lebanon, G., & Zha, H. (2013). Learning multiple-question decision trees for cold-start recommendation. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining* (pp. 445–454). New York, NY, USA: ACM.

Swan, L. G., & Ugursal, V. I. (2009). Modeling of end-use energy consumption in the residential sector: A review of modeling techniques. *Renewable and Sustainable Energy Reviews*, *13*(8), 1819–1835.

Tropp, J. A. (2004). Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information Theory*, *50*(10), 2231–2242.

U.S. Census Bureau . (2013). *American housing survey for the United States.* Washington, D.C.: U.S. Government Printing Office.

U.S. Energy Information Administration . (2009). *Residential energy consumption survey 2009.* `www.eia.gov/consumption/residential/data/2009/`.

Van Raaij, W. F., & Verhallen, T. M. (1983). A behavioral model of residential energy use. *Journal of Economic Psychology*, *3*(1), 39–63.

Weisberg, S. (2014). *Applied linear regression* (4th ed.). John Wiley & Sons.