统计学方法
F filter、LR Filter、KL divergence

Feature Selection Methods for Uplift Modeling

Zhenyu Zhao
Uber Technologies, Inc.
San Francisco, CA, USA
zzy287@gmail.com

Yumin Zhang
Purdue University
West Lafayette, IN, USA
zhan2013@purdue.edu

Totte Harinen
Uber Technologies, Inc.
Amsterdam, Netherlands
totte@uber.com

Mike Yung
Uber Technologies, Inc.
Los Angeles, CA, USA
mike.yung@uber.com

特征选择的优点

***Abstract*—Uplift modeling is a predictive modeling technique that estimates the user-level incremental effect of a treatment using machine learning models. It is often used for targeting promotions and advertisements, as well as for the personalization of product offerings. In these applications, there are often hundreds of features available to build such models. Keeping all the features in a model can be costly and inefficient. Feature selection is an essential step in the modeling process for multiple reasons: improving the estimation accuracy by eliminating irrelevant features, accelerating model training and prediction speed, reducing the monitoring and maintenance workload for feature data pipeline, and providing better model interpretation and diagnostics capability. However, feature selection methods for uplift modeling have been rarely discussed in the literature. Although there are various feature selection methods for standard machine learning models, we will demonstrate that those methods are sub-optimal for solving the feature selection problem for uplift modeling. To address this problem, we introduce a set of feature selection methods designed specifically for uplift modeling, including both filter methods and embedded methods. To evaluate the effectiveness of the proposed feature selection methods, we use different uplift models and measure the accuracy of each model with a different number of selected features. We use both synthetic and real data to conduct these experiments. We also implemented the proposed filter methods in an open source Python package (CausalML).**

***Keywords*-uplift modeling, causal tree, experimentation, feature selection, feature importance**

I. Introduction

Uplift modeling [1]–[10], also known as heterogeneous treatment effect estimation or incremental modeling, is a technique designed to estimate the individual treatment effect (ITE) of an intervention. It can be used for optimizing user targeting and personalization in many areas, including promotion, advertisement, customer service, recommendation system and product design. Most typically, such optimization is achieved by first estimating the treatment effect of the intervention or product experience on each user and then delivering the treatment condition to the users with the largest estimated uplift.

Using informative and predictive features is key for the performance of an uplift model. In practice, there is often a rich set of features that can be used to build a model. However, using all of the available features in the model can lead to computational inefficiency, over-fitting, high maintenance workload, and model interpretation challenges. Consequently, feature selection becomes an essential step to leverage the benefits of a rich feature set and to reduce the associated cost. A feature selection method calculates an importance score for each feature and then ranks them based on the score. An uplift model can then be built based on the most important features. Focusing on the important features only has multiple benefits for uplift modeling applications: (1) faster computation speed for model training; (2) more accurate prediction by avoiding over-fitting; (3 lower maintenance cost for data pipelines; and (4) easier model interpretation and diagnostics.

Although feature selection is an important topic for uplift modeling, it has been rarely discussed in the literature. Feature selection methods for classic machine learning problems have been well studied [11]–[13]. However, as we will show, these methods are ineffective for solving feature selection problem for uplift modeling. Therefore, it is necessary to develop and discuss feature selection methods specifically for uplift modeling.

uplift特征选择很少

We contribute to this area from both methodological and empirical evaluation perspectives. Specifically:

- We propose multiple feature selection methods for uplift modeling.
- We evaluate feature selection methods with various uplift models, in both synthetic and real data settings, in order to provide empirical evidence of method performance.
- We demonstrate that important features for uplift modeling are different from important features for standard machine learning problems, and that feature selection methods for standard machine learning problems are sub-optimal in the uplift modeling context.
- We make the proposed filter methods available in CausalML Python package [14].

We focus on the uplift modeling classification problem where the outcome variable is categorical, which covers many commonly seen use cases such as advertisement click through, new user conversion and existing user retention. However, the idea can be generalized to uplift modeling regression problems.

The structure of the paper is as follows. In Section II, we review the key concepts of uplift modeling and describe why feature selection for uplift modeling is a unique challenge. In Section III, we introduce a list of feature selection methods for uplift modeling. In Section IV, we evaluate these methods with both synthetic and real-world data. Finally, in Section V, we summarize the findings and make recommendations for choosing and using the proper feature selection methods for uplift modeling applications.

II. Background

A. Uplift Models

uplift特征选择的挑战

Uplift modeling can be viewed as a way to estimate heterogeneous treatment effects at a user level using machine learning. It is helpful to frame the problem and introduce uplift modeling from a causal inference perspective. Following the commonly used Neyman-Rubin causal model [15]–[18], the treatment effect for user $i \in \{1, 2, ..., n\}$ can be expressed as:

$$Y_i(1) - Y_i(0) \tag{1}$$

where $Y_i(1)$ and $Y_i(0)$ denotes the outcome variable for individual i under treatment condition and control condition respectively.

The treatment effect can vary from user to user. The conditional average treatment effect (CATE) is defined as:

$$\tau(x_i) := \mathbb{E}[Y_i(1) - Y_i(0) \mid X = x_i] \tag{2}$$

where X is a feature vector and x_i is the feature value for user i.

The CATE quantifies how treatment effects vary among users depending on the observed user features, and it is the target quantity uplift modeling tries to estimate [3]. Based on the estimated CATE, different treatment conditions can be selected and applied to users to achieve preferred outcome. if a model estimates CATEs at an individual level, we also refer to this quantity as the individual treatment effect (ITE),

There are two main types of uplift modeling frameworks. The first category is known as "meta-learners" ([4], [19]), which is based on combining standard machine learning models to estimate the CATE. For example, the "*Two Model*" approach ([20]), also known as *T-learner*, is constructed by fitting a separate model for the control and treatment observations and then taking the difference between the predicted treatment outcome and the predicted control outcome to estimate the CATE. More complex meta-learners include *X-Learner* proposed by [4] and *R-Learner* proposed by [19]. The other category is based on modifying the component within the existing machine learning algorithms such as classification and regression trees. [2], [5], [7], [21]–[23] For example, [5] proposes modifying the splitting criterion of a classification tree algorithm such that the split is optimized for maximizing the heterogeneity of treatment effects in the resulting subgroups. In this paper, we evaluate feature selection methods using models from both categories.

B. Relation with Standard Feature Selection Methods

There are various feature selection methods available for standard classification and regression problems. The methods can be roughly divided into three categories: filter methods, wrapper methods, and embedded methods ([11]–[13]). However, these standard methods fail to perform for the feature selection task for uplift modeling. The reason is that, in the classification problem, the modeling goal is to predict the outcome probability of each class based on the features. Therefore, feature importance is usually measured in terms of its relationship with class probability.

In contrast to the standard classification problem, the goal for uplift modeling is to predict the CATE. Consequently, a good feature should be predictive of the treatment effect rather than a class probability. These two prediction targets do not necessarily coincide. Thus, an important feature for standard classification is not necessarily an important feature for uplift modeling, and *vice versa*. The same argument applies to regression problems.

uplift与ml目标不同

To address the feature selection problem for uplift modeling, we propose both filter methods, which are easy and fast to use as a pre-processing step for uplift modeling, as well as embedded methods, which are a by-product from training an uplift model. We compare the performance between these proposed methods and standard feature selection methods in Section IV.

III. Feature Selection Methods for Uplift Modeling

A. Filter Methods

In an uplift modeling task, a feature's importance depends on how well it predicts the treatment effect. A filter method calculates the importance score for each feature based on the marginal relationship between the treatment effect and the feature. It is a fast pre-processing step because only simple metrics are calculated for one feature at a time.

The first proposed filter method, called F filter, is based on a linear regression model for the outcome variable with the treatment indicator, the feature of interest, and their interaction terms as the predictors. The importance score is defined as the F-statistic for the coefficient of the interaction term: a large statistic value implies the feature is correlated with a strong heterogeneous treatment effect. The second filter method, called LR filter for "likelihood ratio", defines the importance score as the likelihood ratio test statistic for the interaction term coefficient in a logistic regression model.

The third filter method has three variants and is motivated by the split criteria for uplift trees proposed by [5]. For a given feature, this method first divides the samples into K bins based on the percentiles of the feature, where K is a hyperparameter for this method. The importance score is defined as the divergence measure of treatment effect over these K bins. Specifically, assuming there are C classes in the outcome variable, let $P_k = (p_{k1}, \ldots, p_{kC})$ and $Q_k = (q_{k1}, \ldots, q_{kC})$ denote the sample proportion of class in the kth ($k = 1, \ldots, K$) bin for the treatment group and control group respectively. The importance score is defined as:

$$\Delta = \sum_{k=1}^{K} \frac{N_k}{N} D\left(P_k : Q_k\right), \tag{3}$$

where N_k is the sample size in the kth bin, N is the total sample size, and the distribution divergence D is one of the three measures proposed by [5], namely Kullback-Leibler divergence (denoted as KL), the squared Euclidean distance

(denoted as ED), and the chi-squared divergence (denoted as Chi):

$$KL\left(P_k : Q_k\right) = \sum_{i=1}^{n} p_{ki} \log \frac{p_{ki}}{q_{ki}} \quad (4)$$

$$ED\left(P_k : Q_k\right) = \sum_{i=1}^{n} \left(p_{ki} - q_{ki}\right)^2 \quad (5)$$

$$\text{Chi}\left(P_k : Q_k\right) = \sum_{i=1}^{n} \frac{\left(p_{ki} - q_{ki}\right)^2}{q_{ki}} \quad (6)$$

The time complexity for filter methods are linear with the sample size n and number of features m: $O(m \cdot n)$.

B. Embedded Methods

The embedded methods obtain feature importance as a by-product from training a uplift model and can be derived for both meta-learners and uplift trees. For meta-learners, feature importances can be obtained from the base-learners, which are the composite models making up a meta-learner. For example, for the *Two Model* approach, a feature's importance score can be defined as the sum of its embedded importance scores produced by the two base-learners. For uplift trees, the importance score for a feature can be defined as the cumulative contribution to the loss function during the tree node splits in the trees. This is similar to the well-known embedded feature importance for standard classification trees, except the score is obtained from a uplift tree with special splitting criterion. At each split, we calculate the gain in the distribution divergence:

$$\Delta = \sum_{k=\text{left, right}} D\left(P_k : Q_k\right) - D(P : Q), \quad (7)$$

where D is defined as in Eq.(4) to (6), and P, Q denote the outcome distributions of the treatment group and control group. The feature importance score is calculated by summing over all the Δ from the tree node splits where the feature is used.

The time complexity for embedded methods depend on the learners used, for random forest algorithms, it is at order of $O(t_{tree} \cdot m_s \cdot n \cdot \log(n))$, where t_{tree} is the number of trees and m_s is the maximum features considered in each split.

IV. Empirical Evaluation

In this section our goal is to answer following questions: (1) Which feature selection method works better than others? (2) Is the performance consistent in different scenarios? (3) How does the feature selection step affect the accuracy of uplift modeling? (4) How does the number of bins, as a hyperparameter for the bin-based uplift filter methods, affect their performance?

We use both synthetic and real-world data to evaluate the performance of the feature selection methods. The advantage of synthetic data is that the true individual treatment effect and true important features are known, while the advantage of real world data is in helping us to understand how feature selection methods work in practice.

One approach for evaluating the performance of a feature selection method is to feed the top features selected by this method to an uplift model, and then report the accuracy of the uplift model output. We would expect a good feature selection method to identify the truly important features and increase the predictive performance of an uplift model.

A. Experiment 1: Evaluation with Synthetic Data

We consider a binary conversion problem in the study with synthetic data [24]. The generated data has three types of features: (1) uplift features influencing the treatment effect on the conversion probability; (2) classification features affecting the conversion probability but independent of the treatment effect; and (3) irrelevant features that are independent of both conversion probability and the treatment effect. To model the relationship between uplift features and the treatment effect and classification features and outcome probability, we implement six types of association patterns in the data generation process: linear, quadratic, cubic, ReLU (Rectified Linear Unit [25]), trigonometric function sine, and cosine. Example feature patterns are plotted in Figure 1.

The data generating process is composed by the following steps:

- Supposing there are n users and m features, with m_1 classification features, m_2 uplift features, and m_3 irrelevant features ($m_1 + m_2 + m_3 = m$).
- Generating feature value for the ith user and the jth feature from a standard normal distribution: $x_{ij} \sim N(0, 1)$, where $i = 1, 2, ..., n$ and $j = 1, 2, ..., m$.
- Transforming the features to represent different association patterns by applying one [1] of the transformation functions on the feature: $f(x_{ij})$ where $f(x) \in \{x, x^2, x^3, \max(0, x), \sin(x), \cos(x)\}$. The transformed feature values are then standardized by subtracting the mean and dividing by the standard error, and are denoted by z_{ij}.
- Generating the conversion probability based on a logistic model:

$$Pr(Y_i = 1 | X = x_i, Z = z_i, W = w_i) = \frac{1}{1 + exp(-a_1 - \sum_{j=1}^{m_1} z_{ij}\beta_j - w_i \cdot (a_2 + \sum_{j=m_1+1}^{m_1+m_2} z_{ij}\beta_j) + e_{ij})}$$

where X denotes the feature vector, Z denotes the transformed feature vector, W is the treatment indicator variable with 1 for treatment and 0 for control, x_i, z_i, w_i are the realized sample values, a_1 is a constant controlling the baseline conversion probability for control group, a_2 is a constant controlling the average treatment effect, β_j is a coefficient ($\beta_j = U/m_1$ with $U \sim N(0, 1)$ for $j = 1, 2, ..., m_1$, and $\beta_j = 0.5$ for $j = m_1 + 1, m_1 + 2, ..., m_1 + m_2$), and $e_{ij} \sim N(0, 0.3)$ is an error term from a normal distribution with mean

[1] In this simulation study, the transformation function is selected by the natural order for the first six uplift features and the first six classification features. If there are more than six features in a type, then a random transformation function is selected from the set for each additional feature.

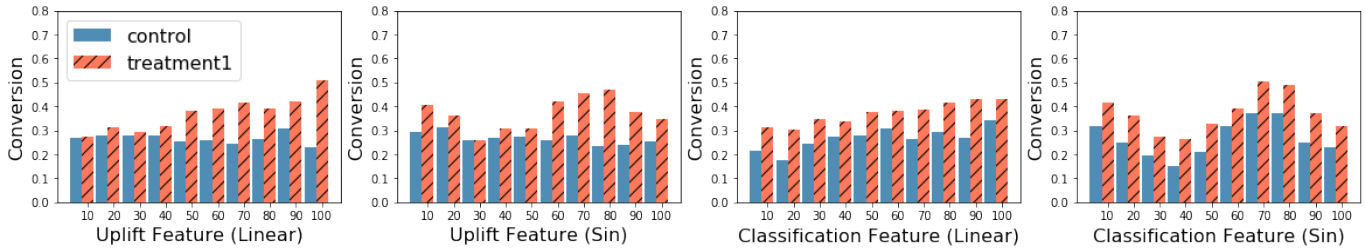

Fig. 1. Feature Association Pattern with Outcome by Experiment Group in Experiment 1. The first two plots demonstrate a heterogeneous treatment effect associated with uplift features in a linear and sine pattern respectively. The last two plots illustrate classification features are correlated with outcome, but not treatment effect.

0 and standard deviation 0.3. Note that classification features affect the conversion probability regardless of the treatment group, while the uplift features only affect the conversion probability for the treatment group, which cause the treatment effect. For each user, we generate a counterfactual conversion probability under both control and treatment: $Pr(Y_i = 1|X = x_i, Z = z_i, W = 0)$ and $Pr(Y_i = 1|X = x_i, Z = z_i, W = 1)$.

- Randomly assigning the control and treatment labels w_i to users with equal probability.
- According to the observed experiment group w_i, generating the observed conversion Y_i by a Bernoulli distribution with probability $Pr(Y_i = 1|X = x_i, Z = z_i, W = w_i)$.

Note that for each user, the true CATE is: $Pr(Y_i = 1|X = x_i, Z = z_i, W = 1) - Pr(Y_i = 1|X = x_i, Z = z_i, W = 0)$. For feature selection and model training, only the feature values x_i, experiment group w_i, and the corresponding outcome Y_i are observed as a training data set.

In this study, there are $m = 36$ features in total, including $m_1 = 10$ classification features, $m_2 = 6$ uplift features, and $m_3 = 20$ irrelevant features. The values for the constants a_1 and a_2 are set such that the average control conversion probability is around 0.2 and the average treatment effect is around 0.1.

We evaluate eight feature selection methods, including five filter methods (F filter, LR filter, KL filter, Chi filter, and ED filter), two embedded methods (*Two Model* embedded and KL embedded), and one standard embedded method for classification as a benchmark (feature importance based on random forest classifier denoted as "outcome embedded"). The embedded methods associated with uplift random forests (KL embedded, Chi embedded, ED embedded) are very similar to each other. Therefore, we use the KL embedded method to represent the performance of this class of methods. For the three uplift filter methods (KL filter, Chi filter, and ED filter), we set the number of bins at 10 as default. We use four uplift models to evaluate the performance of the feature selection methods: *Two Model*, X-learner, R-learner, and KL uplift random forest. As the uplift random forests have a similar performance ([5], [10]), we use the KL model to represent this model family.

For all the meta-learners, we use a random forest classifier as the base learner. In the simulation, all the random forest classifiers in the meta-learners and uplift random forest share the same hyper-parameter values: the number of trees is 10, the maximum tree depth is 10, the minimum sample size in leaf to perform split is 100, and the maximum number of features for split is 3. If the number of features fed into the model is smaller than 3, then we set the maximum number of features for split equal to the number of features.

Each simulation trial consists of four steps. First, we use the data generator to simulate the data with a new random seed and by randomly splitting the data into training and testing (with $50\% : 50\%$ ratio). Second, we apply each feature selection method on the training data and rank the features from the most important to the least important. Third, for each feature selection method, we collect the top m^* (for $m^* \in \{1, 2, 3, ..., 10\}$) features selected and build uplift models based on these features using training data. Fourth, we use testing data to evaluate the accuracy of the uplift models based on the top features selected by each feature selection method. For each trial, we generate $10,000$ samples. The simulation study consists of $t = 100$ trials.

As the main functional goal of uplift modeling is to estimate the CATE or ITE, we expect a good feature selection method to improve an uplift model's accuracy in estimating these effects. Figure 2 summarizes the RMSE (Root Mean Square Error) of ITE estimates by different model and feature selection combinations. The four plots are divided by uplift models. Within each plot, the x-axis shows the number of top features used from the ranked feature list produced by each feature selection method, and the y-axis shows the RMSE of ITE. We use the mean RMSE of the $t = 100$ trials ($\overline{RMSE}$) to make the dot plot and calculate the confidence intervals as $\overline{RMSE} \pm 1.96 \cdot \sigma(RMSE)/\sqrt{t}$, where $\sigma(RMSE)$ is the standard error of the RMSE across trials. We provide a benchmark line as the mean RMSE of the uplift model with all features included.

The results show that the three uplift filter methods (KL filter, Chi filter, ED filter) have consistent top performance in all scenarios, followed by F filter, LR filter, KL embedded methods. The Outcome embedded method has the poorest performance in nearly all scenarios. This observation supports the theory that the standard feature selection method (Outcome embedded) fails for feature selection tasks for uplift modeling. The potential reason for KL filter method outperforming KL

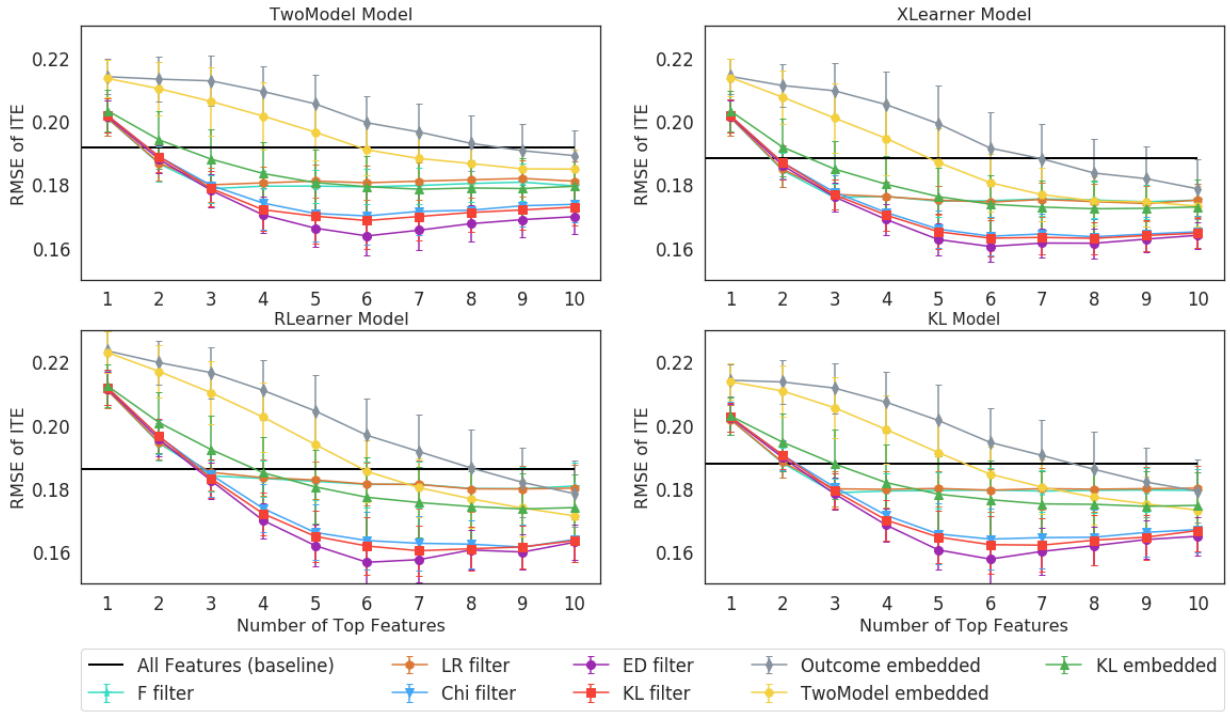

Fig. 2. RMSE of ITE (Individual Treatment Effect) based on Synthetic Data (RMSE lower the better). Each plot stands for one type of uplift model.

embedded method is that the binning in the filter method provides richer information compared with binary node split in the uplift trees.

Except the cases with fewer than 3 features, there is a clear advantage of performing feature selection compared to including all of the features. Peak model performance is achieved at the top 6 features by the three uplift filter methods. This is expected since there are 6 uplift features in the data generation process by design. This also shows that the uplift filter methods are able to choose the 6 true uplift features as the most important ones. As a comparison, the accuracy of other methods keeps improving beyond 6 features, which means they missed some true uplift features in the top 6 positions.

The F filter and LR filter methods have similar performance with the three top performing filter methods for top $1, 2, 3$ features. However, their performance declines after top 3 features. The reason is that F filter and LR filter are good at picking features with a linear uplift pattern but miss features with a nonlinear uplift pattern.

The relative performance of feature selection methods is consistent across different uplift models. Although the purpose of this study is not to compare uplift model performance, the X-learner, R-learner, and KL model perform better than the TwoModel approach (consistent with [5], [10]).

To better understand what explains the increase in uplift model accuracy, we report the proportion of uplift features selected in top 6 positions in Table I. The proportion is averaged across the 100 trials. Note that in each trial, there are 6 uplift features, one in each pattern category. For example, on average, ED filter is able to capture 93.3% of uplift features in the top 6 positions and 99% times the linear uplift feature can be captured in the top 6 positions.

The table shows the three filter methods perform the best for capturing the uplift features, with the ED approach as the strongest method. The order of the feature selection methods in the table is consistent with the order based on uplift modeling performance. This shows the connection between selecting the true uplift features and having a good uplift modeling performance. Consistently with the previous results, we also see poorer performance by standard feature selection methods like "outcome embedded".

The detailed breakdown by uplift feature type explains why some methods are not performing well. F filter and LR filter fail to capture quadratic features, Sin features, and Cos features. These methods, by design, have limitations for selecting nonlinear uplift features. KL embedded method also does not perform well for recognizing Sin features and Cos features.

The three top performing uplift filter methods have one common hyper-parameter: the number of bins. In the study above, we use 10 bins for these methods. It is interesting to study the sensitivity of the feature selection method performance with respect to this hyper-parameter. Therefore, we perform an additional simulation study for KL filter, Chi filter and ED filter. The simulation setting is similar to the one above, except for the number of bins taking different values in $\{2, 5, 10, 20, 50\}$. Figure 3 summarizes the results. The plots are divided by number of top features selected and uplift model type. Within each plot, the x-axis shows the number of bins used by each filter method and the y-axis shows the RMSE of

TABLE I

PROPORTION OF UPLIFT FEATURES SELECTED IN TOP 6 POSITIONS BY METHOD (FEATURE RECALL). THE TABLE IS RANKED BY THE 'ALL UPLIFT' COLUMN, THAT INDICATES PROPORTION OF ALL UPLIFT FEATURES (6 IN TOTAL) BEING CAPTURED IN THE TOP 6 FEATURES RANKED BY EACH METHOD. A BREAKDOWN OF FEATURE RECALL SCORE BY DIFFERENT UPLIFT FEATURE PATTERN IS PRESENTED.

Method	All Uplift	Linear	Quadratic	Cubic	ReLU	Sin	Cos
ED filter	93.3%	99%	97%	78%	97%	94%	95%
KL filter	85%	92%	90%	61%	92%	85%	90%
Chi filter	81.7%	91%	86%	53%	90%	84%	86%
KL embedded	59.8%	77%	90%	65%	62%	25%	40%
F filter	54.8%	100%	11%	100%	100%	8%	10%
LR filter	53.8%	100%	7%	100%	98%	7%	11%
TwoModel embedded	42.7%	74%	9%	35%	76%	24%	38%
Outcome embedded	27.5%	61%	35%	23%	37%	5%	4%

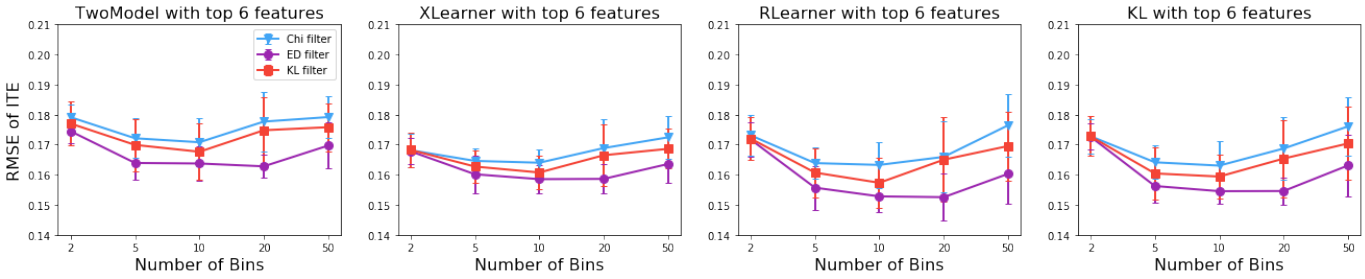

Fig. 3. Performance of Bin-based Uplift Filter Methods with Different Number of Bins. Each plot stands for a different uplift model. The Y-axis shows the performance measured by RMSE of ITE, and the X-axis shows the number of bins used in the filter method.

ITE with a confidence interval. Across these scenarios, the common pattern is that 2 bins is an inefficient choice for fully capturing feature importance, while using 5 or 10 bins is generally a good choice. However, adding more bins does not necessarily improve performance.

B. Experiment 2: Evaluation with Real Data

In this example, we evaluate the proposed methods by using real-world data from an experiment conducted in a mobile phone application. The business context is that a product team would like to increase user conversion for a paid product feature on the application by offering a discount to users. Conversion is defined as whether the user chooses to click and use this feature or not. The default control experience is showing the original price without a discount and the treatment experience is showing the discounted price. The intervention is tested in a randomized experiment and a Chi-squared test shows that the average treatment effect on conversion is statistically significant (p value < 0.01). We train an uplift model on this data and historical user features to predict who would be the customers with the highest expected lift if they were given a promotion. The data set contains 85 features and $300,000$ samples with an equal split between treatment group and control group. We randomly split the observations into training and testing data at $1:2$ ratio.

To test the performance and generalizability of the feature selection methods on uplift models beyond random forest learners, different sets of base learners are tested within the meta-learner approaches. The uplift model variants considered are: (1) TwoModel-LR, XLearner-LR, RLearner-LR using { Logistic Regression Classifier & Linear Regression Regressor } as base learners; (2) TwoModel-LGBM, XLearner-LGBM, RLearner-LGBM using { Gradient Boosting Classifier & Gradient Boosting Regressor } from LightGBM implementation [26] as base learners, with hyperparameter values ($n\_estimators = 100, max\_depth = 8, min\_child\_samples = 100$); (3) TwoModel-RF, XLearner-RF, RLearner-RF using { Random Forest Classifier & Random Forest Regressor } as base learners, with hyperparameter values ($n\_estimators = 100, max\_depth = 8, min\_samples\_leaf = 100$); (4) KL-RF as the uplift random forest using KL divergence criterion with hyperparameter values ($n\_estimators = 10, max\_depth = 8, min\_samples\_leaf = 100$).

The results are summarized in Figure 4, reporting the AUUC (area under the uplift curve) scores [3], [5], [6], [9] from the uplift models using the top 20 features selected by each feature selection method. The relative performance of different feature selection methods can be compared within each column given the same uplift model. Generally speaking, the three bin-based uplift filter methods (Chi filter, ED filter, and KL filter) keep performing well. The KL embedded method also has competitive performance. On the contrary, F filter, LR filter, and outcome embedded methods show poorer performance compared with the methods above. In addition, most uplift models perform more accurately with a feature selection method than they do without a feature selection method. The Logistic / Linear based meta-learner perform worse than more complex models such as LGBM and Random Forests. Despite the differences in uplift models, the relative order of feature selection method performance is quite consistent across different uplift models.

Computation time for feature selection is reported in Table

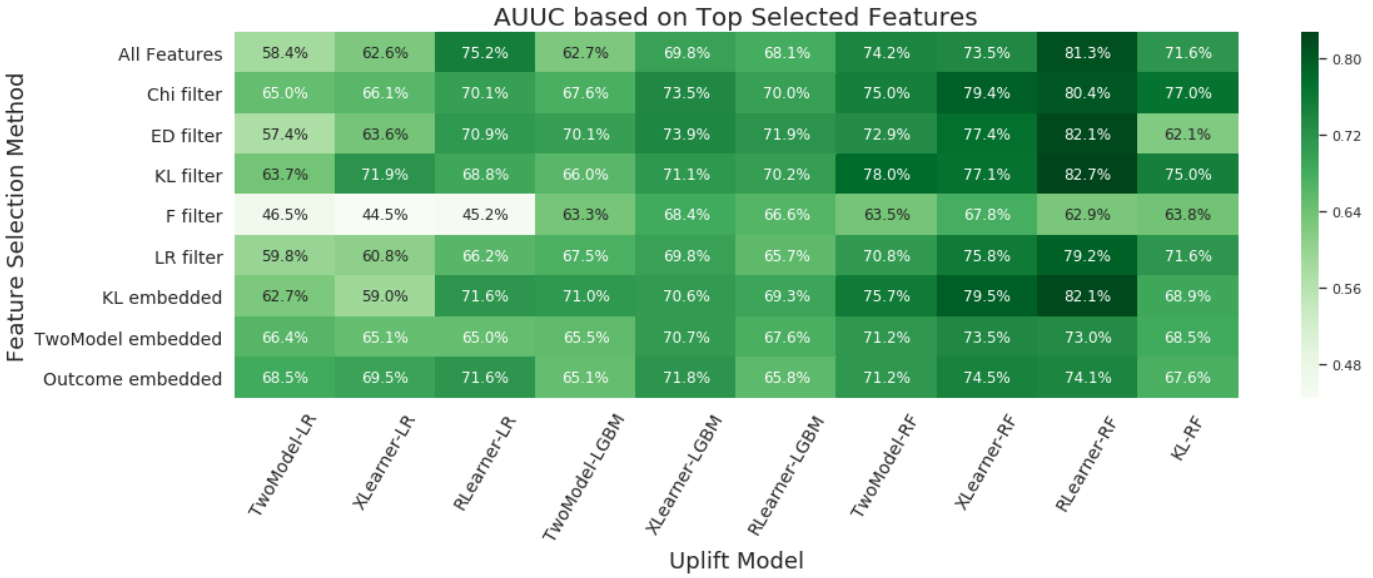

Fig. 4. AUUC of Uplift Models Using the Top 20 Selected Features by Different Feature Selection Methods in the Real Data Experiment. The Y-axis shows the feature selection method and the X-axis shows the uplift model (in a {uplift model - base learner} format) used for producing the AUUC score.

TABLE II
COMPUTATION TIME FOR RANKING 85 FEATURES WITH 100,000 TRAINING SAMPLES IN EXPERIMENT 2. (MEASURED ON A SYSTEM WITH LINUX X86_64, 4 CORES, AND 64 GB MEMORY.)

Category	Filter					Embedded		
Method	Chi	ED	KL	F	LR	KL	TwoModel	Outcome
Time (second)	144	161	161	56	502	6,643	43	58

II. All filter methods have moderate time, while TwoModel embedded method and Outcome embedded method benefit from the Cython implementation of the underlying model in scikit-learn [27]. As a comparison, the KL embedded method has the highest time cost due to pure Python implementation of the tree algorithm.

V. CONCLUSION

We have discussed seven feature selection methods designed for uplift modeling, including filter methods and embedded methods. Our experiments demonstrate that the proposed methods are able to select important features based on their association with heterogeneous treatment effects and improve the ability of uplift models to predict individual treatment effects. In the empirical evaluation on synthetic and real-world data, the three bin-based filter methods, namely Chi filter, ED filter, and KL filter, stand out with a consistently good performance. The embedded method with uplift random forest also shows competitive results. Our experiments also indicate that standard feature selection methods for classification and regression cannot effectively solve the feature selection problem for uplift modeling.

One assumption of the proposed feature selection methods is that the data is collected from randomized experiments, where the treatment assignment mechanism breaks any systematic relationship between the features and whether a unit is in the treatment or control group. If the data is observational and the collected features differ between the treatment and control groups, then the methods proposed here may not improve the accuracy of ITE estimation. The reason is that accurate ITE estimation in observational studies requires us to condition on confounding variables, which are not guaranteed to survive the variable selection process. Extending the approaches proposed here into the observational setting is a promising area of future research.

REFERENCES

[1] J. Grimmer, S. Messing, and S. J. Westwood, "Estimating heterogeneous treatment effects and the effects of heterogeneous treatments with ensemble methods," *Polit. Anal.*, vol. 25, no. 4, pp. 413–434, Oct. 2017.

[2] L. Guelman, M. Guillén, and A. M. Pérez-Marín, "Uplift random forests," *Cybern. Syst.*, vol. 46, no. 3-4, pp. 230–248, May 2015.

[3] P. Gutierrez and J.-Y. Gerardy, "Causal inference and uplift modeling a review of the literature," *JMLR: Workshop and Conference Proceedings 67*, 2016.

[4] S. R. Künzel, J. S. Sekhon, P. J. Bickel, and B. Yu, "Meta-learners for estimating heterogeneous treatment effects using machine learning," Jun. 2017.

[5] P. Rzepakowski and S. Jaroszewicz, "Decision trees for uplift modeling with single and multiple treatments," *Knowl. Inf. Syst.*, vol. 32, no. 2, pp. 303–327, Aug. 2012.

[6] M. Sołtys, S. Jaroszewicz, and P. Rzepakowski, "Ensemble methods for uplift modeling," *Data Min. Knowl. Discov.*, vol. 29, no. 6, pp. 1531–1559, Nov. 2015.

[7] S. Wager and S. Athey, "Estimation and inference of heterogeneous treatment effects using random forests," Oct. 2015.

[8] L. Zaniewicz and S. Jaroszewicz, "Support vector machines for uplift modeling," in *2013 IEEE 13th International Conference on Data Mining Workshops*, Dec. 2013, pp. 131–138.

[9] Y. Zhao, X. Fang, and D. Simchi-Levi, "Uplift modeling with multiple treatments and general response types," May 2017.

[10] Z. Zhao and T. Harinen, "Uplift modeling for multiple treatments with cost optimization," *arXiv preprint arXiv:1908.05372*, 2019.

[11] V. Bolón-Canedo, N. Sánchez-Maroño, and A. Alonso-Betanzos, "A review of feature selection methods on synthetic data," *Knowledge and information systems*, vol. 34, no. 3, pp. 483–519, 2013.

[12] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[13] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," *Data classification: algorithms and applications*, p. 37, 2014.

[14] H. Chen, T. Harinen, J.-Y. Lee, M. Yung, and Z. Zhao, "Causalml: Python package for causal machine learning," *arXiv preprint arXiv:2002.11631*, 2020.

[15] D. B. Rubin, "Estimating causal effects of treatments in randomized and nonrandomized studies," *J. Educ. Psychol.*, vol. 66, no. 5, pp. 688–701, 1974.

[16] J. Neyman, "Sur les applications de la théorie des probabilités aux experiences agricoles: Essai des principes," *Roczniki Nauk Rolniczych*, vol. 10, pp. 1–51, 1923.

[17] D. B. Rubin, "Causal inference using potential outcomes," *J. Am. Stat. Assoc.*, vol. 100, no. 469, pp. 322–331, Mar. 2005.

[18] P. W. Holland, "Statistics and causal inference," *J. Am. Stat. Assoc.*, vol. 81, no. 396, pp. 945–960, 1986.

[19] X. Nie and S. Wager, "Quasi-Oracle estimation of heterogeneous treatment effects," Dec. 2017.

[20] B. Hansotia and B. Rukstales, "Incremental value modeling," *Research Council Journal*, 2001.

[21] S. Athey, J. Tibshirani, and S. Wager, "Generalized random forests," Oct. 2016.

[22] L. Guelman, M. Guillén, and A. M. Pérez-Marín, "Random forests for uplift modeling: An insurance customer retention case," in *Modeling and Simulation in Engineering, Economics and Management*. Springer Berlin Heidelberg, 2012, pp. 123–133.

[23] S. Athey and G. Imbens, "Recursive partitioning for heterogeneous causal effects," Apr. 2015.

[24] Anonymous, "Synthetic data set for uplift modeling," Feb. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3653141

[25] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 8609–8613.

[26] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in neural information processing systems*, 2017, pp. 3146–3154.

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.