Modern approaches to the assessment of demand sensitivity based on uplift models and neural networks (Machine Learning 2021 Course)

Anastasiia Kurmukova [1] **Daria Ustinova** [1] **Evgeny Avdotin** [1] **Kundyz Onlabek** [1]

Abstract

In modern business, a need for targeted offers often appears. One of the most common tasks is sending messages to customers who would not have purchased without this communication. Such problems need a specific approach that is uplift modeling. In this project, we consider the most popular uplift modeling techniques, such as the meta-learners and tree-based approaches. We compare several different classic machine learning models as well as neural networks for base models in meta-learners. Finally, we investigate the performance of the mentioned algorithms in different datasets, both real and synthesized.

Github repo: Github
Video presentation: Google drive

1. Introduction

Nowadays, uplift modeling is a very perspective field of machine learning that predicts the incremental effect of a treatment on an individual's behavior. Despite that retail and marketing are the most popular applications of uplift modeling, the technique is also used in fundraising, clinical trials, medical treatment, human resources, economics, sociology, and even political campaigns that look for people who can be convinced by the right intervention, or "treatment", at the right time.

Let's consider for simplicity a particular example when a company wants to send a customer an SMS with a discount offer. Uplift here is a causal effect of client's re-action on the company's action. Customers can be divided into 4 groups based on their future actions:

1. Sure Things: loyal clients that would make a purchase even without a discount. Sending the offer to them will be a waste of money for the company.
2. Lost Causes: clients that would not buy a product in any case. They are also a bad option for sending SMS.
3. Do-Not-Disturbs: clients that would conversely refuse the service when the discount is offered. This is the worst situation for the company.
4. Persuadables: clients that would buy a product only with the discount. This group can be considered as the best for sending the message.

Obviously, the company would like to detect the fourth group of people and offer them a discount. To deal with it, uplift modeling often relies on randomized experiments. For example, customers can be randomly assigned into 2 groups: treatment and control. The first group will receive the treatment, and the second will not. Then, curves of customers' feedback with impact and without are calculated. The difference between them will be profit. Finally, the company needs to apply the proper action to each user to maximize the target, i.e. money.

1.1. Contributions of the report

The main contributions of this report are as follows:

1. Reviewing different approaches for uplift modeling.
2. Reproducing the most popular uplift modeling techniques: meta-learners using ML approaches and neural networks as basic models, and tree-based models.
3. Evaluating the methods on different datasets and comparing the results with the ones presented in the analyzed papers.

The paper is organized in the following way:

- Section 2 describes the main challenge behind the uplift modeling and presents the solution.

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Daria Ustinova <daria.ustinova@skoltech.ru>.

Final Projects of the Machine Learning 2020 Course, Skoltech, Moscow, Russian Federation, 2020.

- Section 3 contains a review of the classical approaches for uplift modeling as well as the state-of-the-art techniques.
- Sections 4-6 describes the most popular uplift modeling methods.
- In Section 7, there is a description of the datasets used for the experiments with all the required download links.
- In Section 8, the results are reported in terms of performance, reasonable interpretation of the results are provided, and the improvements are suggested.

2. Preliminaries

The biggest challenge in designing uplift modeling algorithms is known in statistical literature as the Fundamental Problem of Causal Inference: "For every individual, only one of the outcomes is observed, after the individual has been subject to an action (treated) or when the individual has not been subject to the action (as a control case), never both" (Sołtys, 2015). In other words, the company cannot simultaneously send and do not send the offer to the same person, so it cannot investigate if the action was beneficial for a given person. That means an uplift for every individual cannot be directly counted (so there is no target). Hence, the uplift cannot be predicted by traditional machine learning approaches as classification, where the true class of an individual is known, at least in the training set.

Denote Y_i^1 as a potential reaction of a client if the message was sent to him (treatment), and Y_i^0 as a reaction when the message was sent (control). Then, the causal effect can be described as the delta of potential reaction of the i-th person and equals:

$$\tau_i = Y_i^1 - Y_i^0.$$

Knowing the feature vector of the i-th person X_i, the expected causal effect of the communication called CATE (Conditional Average Treatment Effect) can be defined in the following way:

$$CATE = \mathsf{E}\left[Y_i^1 \mid X_i\right] - \mathsf{E}\left[Y_i^0 \mid X_i\right].$$

However, τ_i and CATE cannot be estimated directly as the person cannot be in treatment and control groups simultaneously.

The following solution for uplift modeling is proposed:

$$uplift = \widehat{CATE} = \mathsf{E}\left[Y_i \mid X_i = x, W_i = 1\right] - \\ -\mathsf{E}\left[Y_i \mid X_i = x, W_i = 0\right],$$

where

$$Y_i = W_i Y_i^1 + (1 - W_i) Y_i^0$$

is customer's observed response.

3. Related Work

There are several basic methods that are widely used for uplift modeling. In the paper (Gutierrez, 2016), 3 main approaches of uplift modeling were compared: Two-Model, Class Transformation and Modeling Uplift Directly.

In the Two-Model approach, 2 independent models are proposed: the first on the treatment group and the second on the control group. Uplift for every customer is calculated as the difference of predicted probabilities of purchase. The obvious advantage of the approach is its simplicity. However, the models predict the outcome separately and, hence, can miss the "weaker" uplift signal. That is why the technique is mostly outperformed by other methods.

The Class Transformation approach is used in the case of the binary outcome variable. This method is based on creating a new target, which would equal 1 if the outcome is 1 in the treatment group or 0 in the case of the control group. On the one hand, while remaining simple, the approach is better than Two-Model. On the other hand, the approach's condition is too restrictive.

Modeling Uplift Directly is the modification of existing machine learning approaches to estimate a treatment effect directly. Many different suggestions were proposed, including logistic regression, kNN, SVM, but one of the most popular is based on causal trees. Causal trees are very well-known in the tasks of uplift modeling. Firstly, there is a method for decision tree construction based on information-theoretical split criteria described in (Rzepakowski & Jaroszewicz, 2012). In this article, the authors proposed gain normalization for different splitting criteria. They also considered and analyzed the situation when there are several treatment groups.

Also, methods for uplift modeling based on ensembles were investigated. In the paper (Soltys & Jaroszewicz, 2018), the authors proposed a general uplift boosting algorithm with some modifications for three algorithms: uplift AdaBoost, balanced uplift boosting, and balanced forgetting boosting. They provided the results on various datasets, and the best results were obtained with uplift Adaboost. The suggested algorithm is universal and flexible for using different models as estimators in the ensemble. It was also shown that boosting significantly improved their simple base model (in the article it was an uplift tree). The main drawback of the work is that boosting models are compared only with the simple base model. There are no comparisons with other methods such as meta-learners that show great results.

In addition, in the paper (Sołtys, 2015), the authors analyzed the use of bagging and random forest for this task. It was revealed that both approaches performed excellently. Moreover, it was shown that ensembles bring a significant improvement comparing to single trees. In some cases, the

Area Under the Uplift Curve of an ensemble was triple that of the base learner.

One of the recently suggested meta-learner approaches is R-learner proposed in (Nie, 2020). The main novelty of this work is a proposition to minimize R-loss that allows finding an uplift value. This method provides a flexible estimation of heterogeneous treatment effect and can be used with various machine learning models. Moreover, the approach was shown to have a quasi-oracle property with penalized regression, indicating the consistency of training three different models. The authors do not describe exactly their method for constructing a two-step learner, and we think that it can be implemented in a few ways following the main idea. The investigation of the most effective combination of base models is a subject of further research.

As the multiple treatment groups are the common problem in real-life tasks, the authors from Uber Technologies in (Zhao & Harinen, 2020) extended some meta-learners to support the multiple treatments case. Also, they proposed a net value optimization framework to maximize the outcome with treatment cost considered. They provided the evaluation of these models against other models for the multiple treatment case. The most interesting challenge is a further extension of the uplift models to regression problems, where the incremental value will be optimized in a continuous metric.

Neural networks-based models can also be applied to uplift modeling. Authors in (Mouloud et al., 2020) developed a model, which allows to jointly optimize difference in conditional means and the transformed outcome losses. It is shown that such an approach improves modern methods on both synthetic and real data.

4. Meta-learners

The meta-algorithms (or meta-learners) can be described as a framework to estimate the Conditional Average Treatment Effect (CATE) using any machine learning estimators (base learners).

4.1. S-learner

S-learner, proposed in (Künzel et al., 2019), calculates the treatment effect using a single machine learning model in the following way:

1. Estimate the average outcomes $\mu(x)$ with covariates X and an indicator variable for treatment effect W using a machine learning method:

$$\mu(x) = \mathsf{E}\left[Y \mid X = x, W = w\right].$$

2. Define the CATE estimate as:

$$\hat{\tau}(x) = \hat{\mu}(x, W = 1) - \hat{\mu}(x, W = 0).$$

4.2. T-learner

T-learner is a Two-Model approach that consist of the two stages that can be described as follows (Künzel et al., 2019):

1. Estimate the average outcomes $\mu_0(x)$ and $\mu_1(x)$ using machine learning methods:
   - $\mu_0(x) = \mathsf{E}\left[Y(0) \mid X = x\right]$
   - $\mu_1(x) = \mathsf{E}\left[Y(1) \mid X = x\right]$.
2. Define the CATE estimate as:

$$\hat{\tau}(x) = \hat{\mu}_1(x) - \hat{\mu}_0(x).$$

4.3. X-learner

X-learner is an extension of T-learner that consists of three following stages (Künzel et al., 2019):

1. Estimate the average outcomes $\mu_0(x)$ and $\mu_1(x)$ using machine learning methods:
   - $\mu_0(x) = \mathsf{E}\left[Y(0) \mid X = x\right]$
   - $\mu_1(x) = \mathsf{E}\left[Y(1) \mid X = x\right]$.
2. Impute user level treatment effects D_i^1 and D_j^0 for user i in treatment group using $\mu_0(x)$, and user j in control group using $\mu_1(x)$:
   - $D_i^1 = Y_i^1 - \hat{\mu}_0\left(X_i^1\right)$
   - $D_j^0 = \hat{\mu}_1\left(X_j^0\right) - Y_j^0$,

   then estimate $\tau_1(x) = \mathsf{E}\left[D^1 \mid X = x\right]$ and $\tau_0(x) = \mathsf{E}\left[D^0 \mid X = x\right]$ using machine learning methods.
3. Define the CATE estimate by weighted average of $\tau_1(x)$ and $\tau_0(x)$:

$$\tau(x) = g(x)\tau_0(x) + (1 - g(x))\,\tau_1(x),$$

   where $g(x) \in [0, 1]$. Propensity scores can be used as $g(x)$.

4.4. R-learner

R-learner is a two-step algorithm to estimate the treatment effect by minimising the R-loss. The algorithm is the following (Nie, 2020):

1. Fit two based models using cross-validation and predict on holdout fold i the outcomes $\hat{m}^{(-i)}(X_i)$ the propensity scores $\hat{e}^{(-i)}(X_i)$

2. Minimise the R-loss:

$$\tau^*() = \arg\min_{\tau}\{\frac{1}{n}\sum_{i=1}^{n}[(Y_i - \hat{m}^{(-i)}(X_i)) - \\ -(W_i - \hat{e}^{(-i)}(X_i))\tau(X_i)]^2 + \Lambda(\tau())\},$$

where $\Lambda(\tau())$ is a regularizer on the complexity of the τ.

The R-loss is minimized by fitting another model. In our implementation, we use Lasso. The target of the model at the second step is constructed according to the R-loss. In our realization, there are two approaches for minimizing the R-loss. The approach that is used by authors (Nie, 2020) (¡¡original¿¿) in their implementation on R and the approach that is used in causalml library (¡¡causalml¿¿). Our realization of R-learner allows us to use various models as base models unlike causalml, so our learner is much more flexible.

5. Tree based approaches

Tree-based approaches were proposed in the paper (Rzepakowski & Jaroszewicz, 2012). The tree is constructed to maximize the difference between distribution divergence for the treatment and control groups before and after the split. The best split in each node is chosen based on the gain:

$$D_{gain} = D_{aftersplit}(P^T, P^C) - D_{beforesplit}(P^T, P^C),$$

where P^T, P^C are the probability distributions of the outcome for the treatment and control groups.

There are three approaches to calculate the divergence:

- the Kullback-Leibler (KL) divergence:

$$KL(P:Q) = \sum_i p_i \log \frac{p_i}{q_i};$$

- the Euclidean distance:

$$ED(P:Q) = \sum_i (p_i - q_i)^2;$$

- the Chi-squared divergence:

$$\xi^2(P:Q) = \sum_i \frac{(p_i - q_i)^2}{q_i}.$$

We also implemented the technique presented in the paper of (Rzepakowski & Jaroszewicz, 2012), where the normalization value for different gain evaluation functions was proposed. After the realization of the uplift tree, the random forest and the boosting approach for uplift modeling were implemented. The algorithm for Uplift AdaBoost is proposed in the paper of (Soltys & Jaroszewicz, 2018) and described in Fig. 1. The main difference from traditional AdaBoost is the distribution of the different weights for the treatment and control groups and their separated update. The coefficients for the models in the ensemble are the same for the treatment and control groups in AdaBoost.

Input: set of treatment training records, $\mathcal{D}^T = \{(x_1^T, y_1^T), \ldots, (x_{N^T}^T, y_{N^T}^T)\}$,
set of control training records, $\mathcal{D}^C = \{(x_1^C, y_1^C), \ldots, (x_{N^C}^C, y_{N^C}^C)\}$,
base uplift algorithm to be boosted,
integer M specifying the number of iterations

1. Initialize weights $w_{1,i}^T, w_{1,i}^C$
2. For $m \leftarrow 1, \ldots, M$
   (a) $w_{m,i}^T \leftarrow \frac{w_{m,i}^T}{\sum_j w_{m,j}^T + \sum_j w_{m,j}^C}$; $w_{m,i}^C \leftarrow \frac{w_{m,i}^C}{\sum_j w_{m,j}^T + \sum_j w_{m,j}^C}$
   (b) Build a base model h_m on $\mathcal{D}$ with $w_{m,i}^T, w_{m,i}^C$
   (c) Compute the treatment and control errors ϵ_m^T, ϵ_m^C
   (d) Compute $\beta_m^T(\epsilon_m^T, \epsilon_m^C)$, $\beta_m^C(\epsilon_m^T, \epsilon_m^C)$
   (e) If $\beta_m^T = \beta_m^C = 1$ or $\epsilon_m^T \notin (0, \frac{1}{2})$ or $\epsilon_m^C \notin (0, \frac{1}{2})$:
      i. choose random weights $w_{m,i}^T$, $w_{m,i}^C$
      ii. continue with next boosting iteration
   (f) $w_{m+1,i}^T \leftarrow w_{m,i}^T \cdot (\beta_m^T)^{1[h_m(x_i^T) = y_i^T]}$
   (g) $w_{m+1,i}^C \leftarrow w_{m,i}^C \cdot (\beta_m^C)^{1[h_m(x_i^C) = 1 - y_i^C]}$
   (h) $\beta_m \leftarrow \min\{\beta_m^T, \beta_m^C\}$
   (i) Add h_m with coefficient β_m to the ensemble

Output: The final hypothesis

$$h_f(x) = \begin{cases} 1 & if \ \sum_{m=1}^{M}\left(\log\frac{1}{\beta_m}\right) h_m(x) \geq \frac{1}{2}\sum_{m=1}^{M}\log\frac{1}{\beta_m}, \\ 0 & otherwise. \end{cases} \quad (2)$$

Figure 1. A general uplift boosting algorithm from (Soltys & Jaroszewicz, 2018)

6. Neural networks for uplift modeling

The neural network architecture is described in (Mouloud et al., 2020). In the paper, the transformed outcome

$$z_i = \begin{cases} 2 & \text{if } w_i = 1, y_i = 1, \\ -2 & \text{if } w_i = 0, y_i = 1, \\ 0 & \text{otherwise} \end{cases}$$

is used as target values. The model takes the user feature vectors concatenated with the treatment flag. Then, the feature vectors are fed forward through fully-connected layers with the *leaky ReLU* activation function. The output values are transformed in the following way:

$$\hat{y}_i = \begin{cases} \mu(x_i) & \text{if } w_i = 1 \\ -\mu(x_i) & \text{if } w_i = 0 \end{cases},$$

because causal effect by definition equals $Y_i^1 - Y_i^0$, but it is impossible to know both Y_i^1 and Y_i^0 for a user i.

7. Datasets description

7.1. MineThatData E-Mail Analytics And Data Mining Challenge dataset by Kevin Hillstrom

This is synthetic data that contains 64,000 customers who bought something within the last twelve months. The customers were included in an e-mail test, where 50% were randomly chosen to receive an e-mail campaign featuring Women's merchandise and 50% not to receive an e-mail campaign. In the original dataset, there was the third male category for Men's merchandise, but the column was excluded to maintain the data binary. The result was recorded during a period of two weeks following the e-mail campaign. Columns:

- *visit* (binary): target. 1/0 indicator, 1 for customer visited website in the following two weeks.
- *segment* (str): treatment. The e-mail campaign the customer receives: *Women E-Mail* or *No E-Mail*.
- *women* (binary): 1/0 indicator, 1 for customer that purchased Women merchandise in the past year.
- Other columns: *recency*, *history_segment*, *history*, *women*, *zip_code*, *newbie* and *channel*.

The dataset can be reached here.

7.2. Synthetic Data from Kuusisto Thesis

In the paper (Kuusisto, 2015), the author generated a synthetic population of 10000 customers and simulated marketing activity to create a dataset for which the ground truth customer groups are known. To produce the customer population, he first generated a random Bayesian network with 20 nodes and 30 edges. After that, he randomly chose one node with four possible values to be the customer group feature. Columns:

- *outcome* (binary): target. 1/0 indicator, 1 if *positive* and 0 if *negative*.
- *target_control* (binary): treatment. 1/0 indicator, 1 if *target* and 0 if *control*.
- *customer_type* (str): *persuadable*, *sleeping_dog*, *lost_cause*, *sure_thing*.
- Other columns: *customer_id* and 20 *nodes*.

The dataset is available here.

7.3. Dataset by X5 Retail Group

The dataset is made by X5 Retail Group which is the top-1 retail company in Russia. The company sends messages to the customers to encourage them to shop more. The first dataset that was used is `clients.csv`. It contains general information about clients (id, first issue date, first redeem date, age, and gender). There are also other descriptive datasets presented by the company such as `products.csv` and `purchases.csv`. In addition, there is a dataset `uplift_train.csv`, and its main columns are the following:

- *target* (binary): target. 1/0 indicator, 1 - indicates if there was a purchase after communication.
- *treatment_flg* (str): treatment. 1/0 indicator, 1 - shows if there was a communication.
- *client_id*.

The link for the dataset is here.

8. Experiments and Results

Github repo: Github

One of the main tasks of this project was to get familiar with uplift modeling and to create our models by hand, which should be comparable with other ones from the special library - causalml.

For our experiments, we split each dataset into 2 parts: X_{train} and X_{test}. For the test part, we have left $\frac{1}{3}$ of the sample size.

8.0.1. Metrics

We used uplift at top 30% predictions ($uplift\_at\_k$) as a base quality metric. This score was chosen according to $X5$ competition, as it has industrial application. It is important to obtain the largest uplift for some fixed number of customers. Also, we took into consideration $qini\_auc$ and $uplift\_auc$ scores.

$$uplift\_curve(t) = (\frac{Y_t^T}{N_t^T} - \frac{Y_t^C}{N_t^C})(N_t^T + N_t^C)$$

$$qini\_curve(t) = Y_t^T - \frac{Y_t^C N_t^T}{N_t^C},$$

where Y_t^T and Y_t^C are targets in the treatment and control groups, respectively. N_t^T and N_t^T are the sizes of groups and t is the number of the targeted objects.

8.1. Base learners results

8.1.1. Grid search

Firstly, for all datasets, we performed a kind of grid search for each learner over its hyperparameters. For meta learners, a hyperparameter is a classifier (or regressor), which in turn has its own hyperparameters. For models with 2 input models, we considered all possible combinations of them. As a score we used average $qini\_auc$ calculated over cross validation: in the training part, we split X_{train} into 3 parts, fit the model on $\frac{2}{3}$ of X_{train} and predicted on other $\frac{1}{3}$. Parameters for grid search can be found in the Appendix C.1.

8.1.2. Numerical Results

Table 1 shows a comparison of top 30% uplift scores for our and causalml meta-learners. As S and T learners are rather simple approaches, all scores are equal.

Our X learner performed significantly better only on Kusisto's dataset, while R meta-learner performed better or equal on all datasets.

On Figures 2-3 some results for qini and uplift curves are provided.

Table 1. Top 30% uplift for meta-learners

Dataset	Our models	CausalML
S learner		
X5	0.0655± 0.0016	0.0654± 0.0011
Kuusisto	0.005± 0.019	0.005± 0.019
MineThatData	0.076± 0.005	0.076± 0.005
T learner		
X5	0.067± 0.005	0.067± 0.005
Kuusisto	0.009± 0.006	0.009± 0.006
MineThatData	0.061± 0.008	0.061± 0.008
X learner		
X5	0.049± 0.003	0.053± 0.001
Kuusisto	0.029± 0.006	0.009± 0.021
MineThatData	0.058 ± 0.012	0.07± 0.008
R learner		
X5	0.04± 0.000	0.039± 0.001
Kuusisto	0.012± 0.000	0.012± 0.000
MineThatData	0.067 ± 0.000	0.04± 0.000

8.2. Tree approaches results

We implemented uplift Decision Tree, uplift Random Forest (Rzepakowski & Jaroszewicz, 2012) and uplift AdaBoost (Soltys & Jaroszewicz, 2018) with GridSearch for each dataset and obtained the best parameters for Decision Tree and Random Forest. The example of Qini curve for an uplift tree for the X5 dataset can be found in Fig. 4. Then, we

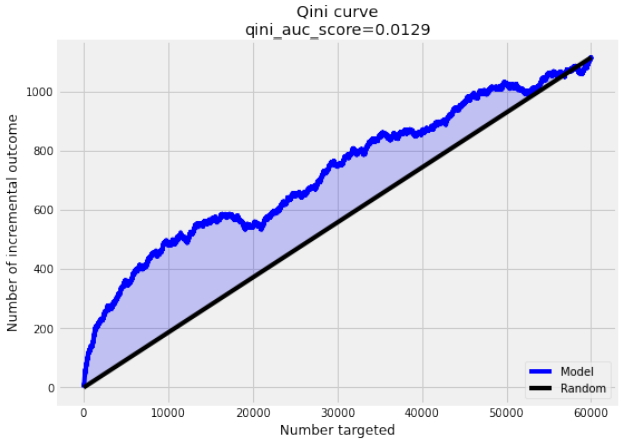

Figure 2. Qini curve for our T learner, tested on X5 dataset

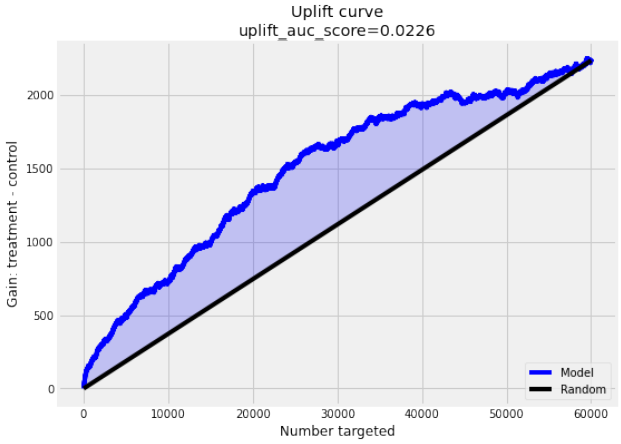

Figure 3. Uplift curve for our S learner, tested on X5 dataset

trained a few models with the best parameters on different parts of the train and predicted our best treatment group (0 or 1) for each observation on the test. After that, we calculated the mean 30% uplift and its variance. The base model for boosting was an uplift tree with depth 5 and the number of estimators in boosting was 50. The results can be seen in Table 2. Causalml library only has an uplift Decision Tree and an uplift Random Forest. It does not contain any boosting approaches. Also, the normalization value for an uplift tree in causalml is implemented in another way than in the original article (Rzepakowski & Jaroszewicz, 2012). We have fixed the drawbacks of this library in our implementation and expanded the functionality with boosting.

It can be seen from the results that boosting has better or the same performance compared to Decision Tree and Random Forest. However, AdaBoost and Random Forest take much

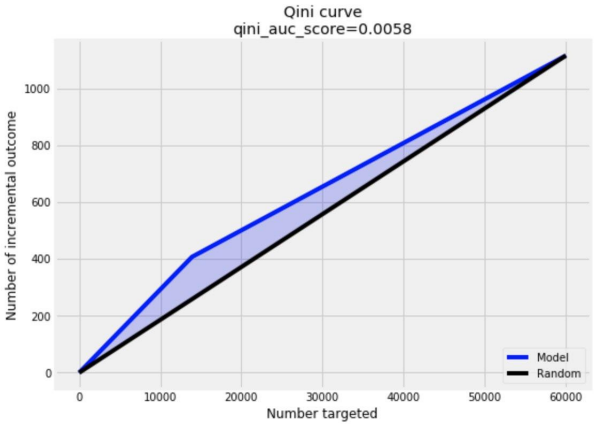

Figure 4. Qini curve for tree with depth 8, X5 dataset

Table 2. Top 30% uplift at tree-based approaches

DATA	DECISIONTREE	RANDOMFOREST	BOOSTING
X5	0.0422± 1E-4	0.0386± 0.	0.038
KUU	0.012± 0.	0.012± 0.	0.018
MTD	0.0514± 1E-4	0.0434± 1E-4	0.052

more time for fitting than the single Decision Tree. On some data, the improvement of the score by using ensembles can be small, while on the other the increase of time is significant.

8.3. Neural Networks results

In this section, we present the results of a neural network, described in 6. It was trained on the training part of the X5 dataset with Adam optimizer and constant learning rate for 40 epochs. Validation results are presented in Figures 5-6. As it can be seen, the results suffer from outliers, and AUC is negative. That is why such architecture cannot be considered efficient and requires further improvement.

9. Conclusion

To sum up, uplift modeling is found to be very applied method for solving various modern problems such as sending a discount to the right customers. In this work, the most well-known uplift modeling approaches were considered and reproduced. The approaches are the following: meta-learners (S-, T-, X- and R-learners), tree-based approaches (uplift tree, random Forest), boosting (AdaBoost), and neural networks. The methods were tested and compared on 3 different datasets (both real and synthetic). Meta-learners revealed relatively good results on datasets. We provided

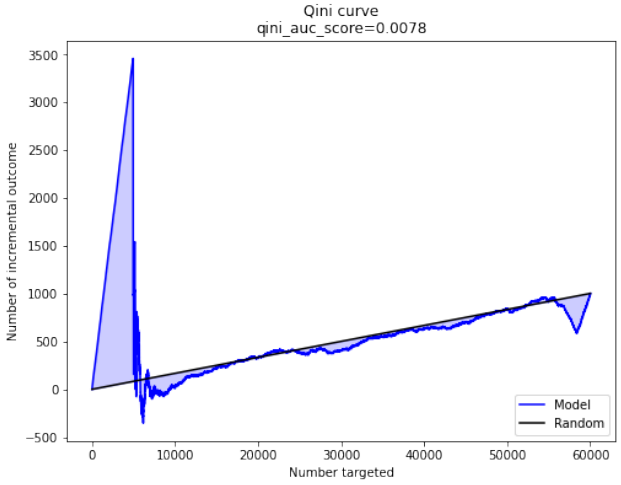

Figure 5. Validation Qini curve for neural network, X5 dataset

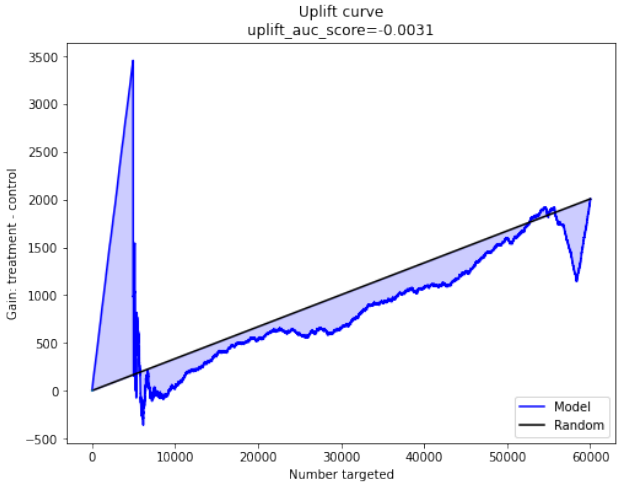

Figure 6. Validation uplift curve for neural network, X5 dataset

the comparison with causalml for proving the correctness of our implementation. Tree-based models and boosting have shown comparably good results on all datasets. Boosting almost in all cases performed better than a single decision tree and random forest. Causalml library did not contain Adaboost, so we cannot compare results, but with boosting we were able to achieve better scores. In addition, the neural networks approach turned out to be not very effective on our datasets.

References

Gutierrez, P., G. J. Causal inference and uplift modeling: A review of the literature. *JMLR: Workshop and Conference*

Proceedings 67, pp. 1–13, 2016.

Künzel, S. R., Sekhon, J. S., Bickel, P. J., and Yu, B. Metalearners for estimating heterogeneous treatment effects using machine learning. *Proceedings of the national academy of sciences*, 116(10):4156–4165, 2019.

Kuusisto, F. C. Machine learning for medical decision support and individualized treatment assignment. *http://pages.cs.wisc.edu/ dpage/kuusisto.thesis.pdf*, pp. 38–40, 2015.

Mouloud, B., Olivier, G., and Ghaith, K. Adapting neural networks for uplift models. *arXiv preprint arXiv:2011.00041*, pp. 1–10, 2020.

Nie, X., W. S. Quasi-oracle estimation of heterogeneous treatment effects. *https://arxiv.org/abs/1712.04912*, pp. 1–50, 2020.

Rzepakowski, P. and Jaroszewicz, S. Decision trees for uplift modeling with single and multiple treatments. *Knowl. Inf. Syst., 32(2):303–327*, 2012.

Soltys, M. and Jaroszewicz, S. Boosting algorithms for uplift modeling. *https://arxiv.org/pdf/1807.07909.pdf*, 2018.

Sołtys, M., J. S. . R. P. Ensemble methods for uplift modeling. In *Data Min Knowl Disc 29*, pp. 1531–1559, 2015.

Zhao, Z. and Harinen, T. Uplift modeling for multiple treatments with cost optimization. *https://arxiv.org/pdf/1908.05372.pdf*, 2020.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Anastasiia Kurmukova (35% of work)

- Implementation of R-learner
- Implementation of uplift Decision Tree, Random Forest
- Implementation of uplift AdaBoost
- Obtaining results for trees, random forest and boosting
- Literature review of 4 articles
- Partially prepared sections 3,4,5,8 of this report

Daria Ustinova (25% of work)

- Reviewing literature on the topic (3 papers)
- Coding S and T meta-learners
- Coding preprocessing for MineThatData and Kuusisto's datasets
- Coding grid searches algorithms for meta learners
- Coding the experiments for base meta learners on all datasets
- Partially prepared sections 1,3,8 of this report

Evgeny Avdotin (25% of work)

- X-learner implementation
- Metalearners cross-validation on MineThatData dataset
- Neural network implementation

Kundyz Onlabek (15% of work)

- Literature review of 2 articles
- In the report, prepared sections 1,2,3,4.1,4.2,7,9 and edited the remaining sections
- Data preproccessing: MineThatData and Kuusisto thesis

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment:

   - We used ready preprocessing of X5 dataset (approx.10% wrote on our own, reference)
   - We modified ready preprocessing of MineThatData dataset (approx.50% wrote on our own, reference)
   - While coding our uplift models we used general structure from causalml package , so it can be approx. 70% on our own

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

4. A complete description of the data collection process, including sample size, is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

9. The exact number of evaluation runs is included.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: None

10. A description of how experiments have been conducted is included.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    Students' comment: None

11. A clear definition of the specific measure or statistics used to report results is included in the report.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    Students' comment: None

12. Clearly defined error bars are included in the report.

    ☐ Yes.
    ☐ No.
    ☑ Not applicable.

Students' comment: None

13. A description of the computing infrastructure used is included in the report.

    ☐ Yes.
    ☐ No.
    ☑ Not applicable.

    Students' comment: None

C. Grid search parameters

C.1. Grid search for meta learners

For classifiers-based learners S, T, R RandomForestClassifier ({n_estimators: $[50, 100]$,max_depth: $[7, 11]$}), LogisticRegression ({penalty: $['l2']$}), XGBClassifier({n_estimators: $[50, 100]$, max_depth: $[3, 5]$}), KNeighborsClassifier({n_neighbors:[10, 20, 50]}) were tested at each grid search.

For X learner better results were obtained exactly at regression models, so grid search were performed over RandomForestRegressor(),SGDRegressor(),XGBRegressor(), KNeighborsRegressor() with the same parameters, respectively.