2018 International Conference on Intelligent Systems (IS)

A Comparison of Data-Driven Approaches for Mobile Marketing User Conversion Prediction

Luís Miguel Matos
ALGORITMI Centre
University of Minho
Guimarães, Portugal
id6929@alunos.uminho.pt

Paulo Cortez
ALGORITMI Centre
Dep. Information Systems
University of Minho
Guimarães, Portugal
pcortez@dsi.uminho.pt

Rui Mendes
ALGORITMI Centre
Dep. Informatics
University of Minho
Braga, Portugal
rcm@di.uminho.pt

Antoine Moreau
OLAmobile
Spinpark
Guimarães, Portugal
antoine.moreau@olamobile.com

***Abstract*—In this paper, we perform an exploratory study of user Conversion Rate (CVR) prediction using recent big data from a global mobile marketing company. We design a stream processing engine to collect sampled mobile marketing data. Then, we execute a large set of CVR prediction tests, under a two-stage experimental procedure that considers a rolling window evaluation. First, several preprocessing and machine learning combinations are analyzed using preliminary data. Next, the selected combinations are tested on a larger set of unseen datasets. Interesting classification performances were achieved, with some learning models (e.g., XGboost, Logistic Regression) requiring a reduced computational effort, thus showing a potential value for user CVR prediction in this domain.**

Keywords – Classification*; *Conversion Rate (CVR)*; *Big Data*; *Data Mining*; *Mobile Performance Marketing

I. Introduction

Mobile performance marketing is growing due to the widespread usage of mobile devices (e.g., smartphones, tablets). Several mobile advertising commercial platforms have been created, in what is known as Demand-Side Platforms (DSP) [1]. The DSP acts as a broker, matching user profiles to ads, thus linking user traffic, coming from publishers (e.g., news site, game app) to advertisers. And, if there is a conversion (e.g., product sale), the DSP facilitates the cash flows, returning a portion of the advertiser's revenue to the publishers. All ad clicks and sales generate data events, leading to big data with its 4Vs characteristics [2]: volume, velocity, variety and value. A critical aspect of the DSP big data system is the prediction of the user Conversion Rate (CVR), which involves estimating if there will be a sale when a user clicks and views an advertisement [3].

In this paper, we perform a large number of computational experiments, aiming to compare several data preprocessing and machine learning approaches for predicting user CVR responses after clicking a mobile ad link. As a case study, we work with recent real-world data from OLAmobile, which is a global mobile advertising company that created and maintains its own DSP. First, we design a stream processing engine to collect sampled data from the DSP data center. Then, we execute a vast experimental comparison, under a two-stage experimental design that includes distinct datasets, data preprocessing (five categorical and five balancing training set transformations) and machine learning (three offline and three online algorithms) combinations, and a realistic rolling window evaluation.

This paper is organized as follows. First, the related work is introduced (Section II). Next, the data and methods are described (Section III). Then, the experimental results are presented and analyzed (Section IV). Finally, the main conclusions are discussed (Section V).

II. Related Work

Several works approached user CVR prediction, mostly using linear models, such as Logistic Regression (LR) [4]. Recently, more flexible machine learning methods have been proposed, such as Gradient Boosting Decision Trees (GBDT) [4], [5], Random Forest (RF) [3] and Deep Learning [6]. Yet, these studies tend to only consider the prediction performance and not the computational effort. For instance, the Deep Learning models proposed in [6] are more complex than the LR method, although the classification only improved slightly. Since DSP generate big data, with millions of worldwide clicks per hour, constant model updates and real-time predictions are required. In this paper we evaluate both the predictive performance and computational effort.

Data preprocessing is another relevant issue. Due to privacy and DSP issues, only a limited set of mobile CVR related attributes is available, which increases the complexity of the prediction task (e.g., it is not possible to identify a user). Attributes are mostly categorical, often presenting a large cardinality, with hundreds of levels. CVR works have adopted either raw numeric encodings (e.g., [4]) or one-hot-encoding (e.g., [3], [6]), thus distinct categorical transformation methods are rarely compared. One-hot is a popular transformation but it presents limitations, since it heavily increases the computational effort for high cardinality attributes. Moreover, CVR are highly imbalanced, with sales often corresponding to less than 1% of the generated data events. Thus, balancing the training data (e.g., undersampling or SMOTE [7]) might

978-1-5386-7097-2/18/$31.00 ©2018 IEEE

improve results, although this aspect has been neglected in most CVR prediction studies.

In contrast with previous CVR works, in this paper we test a larger set of combinations of preprocessing and learning methods, including five categorical transformations (e.g., raw, categorical or one-hot), five training set setups (e.g., none, SMOTE) and six learning algorithms (e.g., LR, RF). Also, we adopt a more realistic and robust rolling window validation [8], [9], which simulates several holdout (train and test) iterations through time, rather than the simpler holdout validation used in [3]–[6].

III. Materials and Methods

A. Stream Engine and Collected Data

Under the analyzed market, publishers put a dynamic link in their web pages or apps. Once it is clicked, the DSP selects a marketing campaign, redirecting the user to a specific ad and advertiser. Two data events are generated: redirects, when users click the dynamic link; and sales, when there is a conversion. All events are stored at the DSP data center, which receives millions of redirects and thousands of sales per hour. The DSP provided us a secure web service (https) that allows to request a total of NR redirects or NS sales from the data center.

In this work, we had access to an Intel Xeon 1.70GHz server with 56 cores and 2TB of disk, which has limited capabilities when compared with the data center and thus we worked with sampled data. We designed a stream processing engine (Figure 1) using the **R** tool [10]. The engine sets K cores for requesting redirects and sales. After receiving the stream (in JSON format), each core sleeps for SR or SS seconds and then repeats the request (asking for more data). The received streams are sent to another layer of cores, which filter the data according to some of its attribute values. The filtered streams are then stored (first in, first out - FIFO order) in three files: redirects; sales; and an event log, used for monitoring the data collection. These files were stored using MongoDB, a fast NoSQL JSON database system [11].

Table I describes the stream collection parameters and resulting datasets. The **Traffic** column distinguishes two main event types:

- TEST – initial DSP testing mode, used to measure campaign performance; and
- BEST – with best product campaigns that have obtained a minimal TEST performance and that corresponds to most traffic.

The last two columns denote the number of redirects that produced sale (Y_{yes}) and no sale events (Y_{no}). For two TEST datasets (30 minutes and 1 week), the amount of events collected is around half when compared with the other datasets. This is due to the fact that TEST traffic, which is scarcer than BEST traffic, changes through time and the datasets were collected at different periods.

Although we increased the number of redirect request cores (K column of Table I) for the shorter duration datasets, due to the web service limitations it was not possible to retrieve all redirects. Thus, our ratio of collected sales $Y_{\text{yes}}/(Y_{\text{no}}+Y_{\text{yes}})$ is often higher than the real DSP ratio, ranging from 2.1% to 34.4% (BEST) and 0.2% to 20.4% (TEST). This issue is handled by setting two data variants:

- **collected** – with all stored events; and
- **realistic** – with a sample of the collected data such that the overall sales ratio is 1% for BEST and 0.5% for TEST.

All redirect and sale events were merged into tabular files for the classification modeling. Table II lists the respective input categorical attributes and output target (last row), as provided by the DSP. The attributes are defined in terms of their context (user, advertiser, publisher or target) and description (including the number of levels and example values).

B. Data Preprocessing

We compare five transformations to handle the nominal inputs: raw (R), categorical or one-hot coding (C), Inverse Document Frequency IDF (I), categorical pruned (CP) and IDF pruned (IP). The categorical coding was computed using only training data. When necessary, training transformation variables (e.g., IDF numeric value for a level) were stored, such that test data could be encoded using the same transform. Moreover, a special "new" category was set to match any new levels present in test data and that could not be known at training time.

The transformations work as follows:

- R – uses the original numeric value of the data ("new" is encoded as 0).
- C – assumes a categorical attribute for methods that can handle directly such attributes (e.g., tree based or RF) or one-hot encoding for numeric based methods (e.g., LR). In this second case, the **R** tool transforms each attribute into $L-1$ binary inputs, where L is the number of levels (including the special "new" value).
- CPF - proposed variant used to reduce the input memory requirements. It first ranks the L levels according to their frequency in the data. Only the most frequent F levels are used and all other levels (including "new") are merged into the special "other" category.
- I – coding adopts the transform [12]: $I(x) = ln(n/f_x)$, where n is the number of instances and f_x is the frequency of attribute x. The transformed $I(x)$ values range from near 0 (most frequent level) to a I_{max} (less frequent).
- IPF – proposed procedure that works by selecting the most frequent F levels, grouping other levels except "new" into an "other" category, and then applying the $I(x)$ function to the $F+1$ levels. In both I and IP procedures, the "new" level is transformed into the least frequent numeric value (I_{max}).

We explore one normal and four balancing methods, which were applied only to training data. Thus, the test sets are kept with their original unbalanced target distributions. The data transformation methods include [7]: none (N), undersampling (U), oversampling (O), both (B) and SMOTE (S). The N

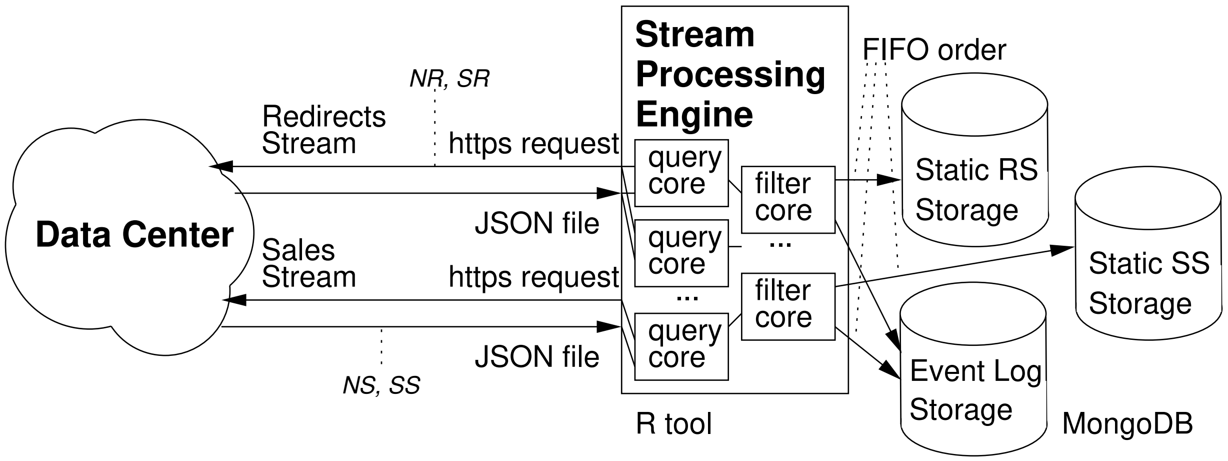

Fig. 1. Scheme of the developed stream processing engine.

TABLE I
SUMMARY OF THE STREAM ENGINE SETUP AND COLLECTED DATASETS

Traffic	Duration	Start	NR	NS	SR	SS	K	Y_{no}	Y_{yes}
BEST	30 min.	14/09/2017 16:12	7,500	2,000	0	0	5	235,069	9,401
	1 hour	28/11/2017 19:29	5,000	1,000	0	0	5	229,707	4,847
	1 day	16/11/2017 12:55	500	500	60	60	2	227,687	119,388
	1 week	07/12/2017 15:23	70	50	150	150	2	180,629	80,775
TEST	30 min.	28/09/2017 09:29	10,000	2,000	0	0	5	85,312	167
	1 hour	22/01/2018 18:10	7,500	1,000	0	0	5	217,102	765
	1 day	22/01/2018 18:10	5,000	500	60	60	5	216,369	1,242
	1 week	17/12/2017 22:25	200	50	150	150	2	98,172	25,131

TABLE II
DESCRIPTION OF THE DATA ATTRIBUTES

Context	Attribute	Description
user	country	user country: 192 to 216 levels (e.g., Russia, Spain, Brazil)
	region	region of the country: 22 to 24 levels (e.g., Asia, Europe)
	browser	browser name: 13 to 14 levels (e.g., Chrome, Safari)
	operator	mobile carrier or WiFi: 377 to 437 levels (e.g., Vodafone)
advertiser	vertical	ad type: 3 to 5 levels (e.g., video, mainstream, dating)
	campaign	ad product identification: 797 to 1564 numeric levels
	special	smart link or special offer: 451 to 1271 levels
publisher	account	publisher type: 7 to 11 levels (e.g, app developer, webmaster)
	manager	publisher account manager: 18 to 26 levels (numeric)
target	Y	if there is a conversion: 2 levels (no, yes)

method uses the original data. The U, O and B methods work by sampling the data according to:

- U – uses a sample of Y_{yes} negative examples;
- O – uses a sample (with repetition) of Y_{no} positive examples;
- B – the minority classes are oversampled and the majority classes are undersampled (according to the default **R ROSE** package) [13]; and
- S – creates new synthetic positive examples within the neighborhood input space of the positive labels and undersamples the negative examples, as implemented in the **R DMwR** package [14].

C. *Classification Methods*

The comparison includes three offline and three online classifiers. To reduce the bias towards a given algorithm and perform a fair comparison, we executed all algorithms with their default parameters.

The offline algorithms include: Logistic Regression (LR), Random Forests (RF) and XGboost (XB). LR is a popular linear model for CVR prediction. Both RF and XB are based on decision tree ensembles. RF was proposed in 2001 [15] and it combines the responses of a large number of decision trees. In [3] it provided the best CVR prediction results, although it required much more computation than LR. More recently, the scalable XB gradient boosting algorithm was proposed in 2016 [16], winning several classification challenges and requiring less computational effort than RF.

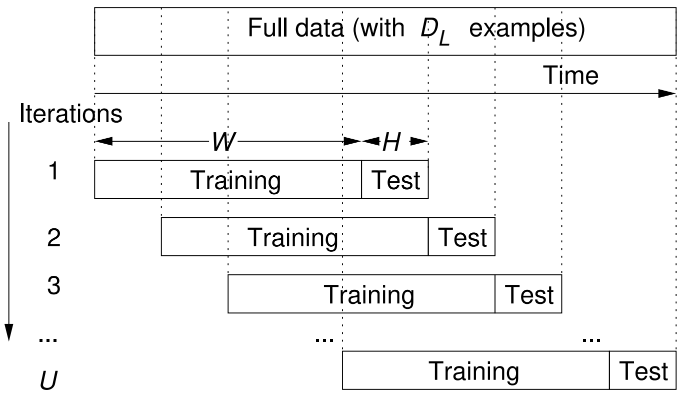

Fig. 2. Rolling window evaluation.

Regarding the online learning algorithms, these include OzaBoost (OB), DecisionStump (DS) and Random Hoeffding Trees (RH). OB is an online boosting ensemble version of the *AdaBoost.M.1* algorithm, DS is based on one-level decision trees and RH uses incremental decision trees [17]. All algorithms were implemented in the **R** tool [10], using the packages **rminer** [18], for the offline learning, and **RMOA** [17], for the online learning.

D. Evaluation

We adopted the robust rolling window validation [8], [9], which simulates a real classifier usage through time, with several training and test updates (Figure 2). In the first iteration, the learning model is fit to a training window with the W oldest examples, and then predicts H ahead predictions. Next, the training set is updated by discarding the oldest H records and adding H more recent ones. A new model is fit, producing H new predictions, and so on. In total, this produces:

$$U = \frac{D_L - (W + H)}{H} + 1 \qquad (1)$$

classifier updates (training and test iterations), where D_L is the data length (number of examples). In this work, after consulting OLAmobile experts, we opted to use the realistic values of $W = 50,000$ and $H = 3,000$.

The predictive performance is measured using test data and the area under the curve (AUC) of receiver operating characteristic (ROC) curve [19]. Often, the quality of the AUC values is interpreted as: 50% performance of a random classifier; 60% - reasonable; 70% good; 80% very good; 90% excellent; and 100% perfect. We also record the computational effort (in seconds) for each rolling window iteration.

The experimental design includes two stages. First, we conduct a large number of preliminary experiments using the oldest collected datasets (duration of 30 minutes), with all preprocessing and classifier combinations. Then, the selected first stage combinations are tested over a larger number of unseen datasets. The goal is to measure the performance of the selected combinations on unseen data. To aggregate all execution results (e.g., AUC values of the U rolling window

TABLE III
RESULTS FOR DISTINCT CPF AND IPF LEVELS (MEDIAN VALUES, BEST VALUES IN **BOLD**)

Traffic	**Variant**	**AUC (in %)**			**Effort (in s)**		
	(results)						
BEST		CP10	CP20	CP30	CP10	CP20	CP30
	collected	**84.52**	75.57	72.03	**53.03**	68.26	68.58
	realistic	**82.98**	72.01	72.8	**63.13**	80.64	78.96
	(R)	(60)	(59)	(59)	(60)	(59)	(59)
		IP10	IP20	IP30	IP10	IP20	IP30
	collected	84.49	83.95	**84.61**	**62.93**	63.57	63.82
	realistic	**85.15**	84.04	84.34	82.93	**83.03**	83.34
	(R)	(60)	(60)	(60)	(60)	(60)	(60)
TEST		CP10	CP20	CP30	CP10	CP20	CP30
	collected	**64.30**	63.00	57.85	**42.6**	51.51	58.54
	realistic	**69.47**	66.22	63.78	**52.05**	74.27	74.15
	(R)	(60)	(59)	(59)	(60)	(59)	(59)
		IP10	IP20	IP30	IP10	IP20	IP30
	collected	61.07	59.5	**62.80**	50.06	**49.77**	49.88
	realistic	67.48	66.01	**68.37**	61.71	**61.67**	62.67
	(R)	(60)	(60)	(60)	(60)	(60)	(60)

test iterations), we compute the median value, instead of average values, since this statistic is less sensitive to outliers. All median values are estimated by the Wilcoxon non parametric statistic [20]. The same Wilcoxon test is used to check if paired median differences are statistically significant.

IV. RESULTS

A. First Phase

This phase uses only the 30 minutes data and starts by the setting of the number of pruned levels (F). Then, the preprocessing and classifier performances are compared. For each factor of analysis (e.g., CP10), there are E executed experiments (e.g., different classifiers). Some experiments produce computational errors (e.g., lack of memory), resulting in R rolling window results ($R \leq E$). For each rolling window execution, we compute the Wilcoxon median result over all U iterations. Then, we compute the Wilcoxon median over all R executions.

For the pruned encodings (CP and IP), we tested $F \in \{10, 20, 30\}$. Table III presents the median results (AUC and computational effort) when fixing a particular F value, resulting in E = 5 (training setups) × 6 (classifiers) = 30 experiments per level and data variant. In Table III, the number of execution results (R) is shown in brackets and aggregated for both data variants. In a few cases, the F = 20 and $F = 30$ levels led to an execution error, resulting in $R = 59$ (and not 60). Since $F = 10$ did not lead to execution errors and provided the best AUC and computational effort overall results, we opted to fix this value. Table IV presents the overall first phase results when fixing an encoding method. All executions were successful, except for the categorical (C) encoding, confirming that the C transformation is problematic for high cardinality attributes. The last row presents the overall **median** values, computed over the four data setups. The C encoding also produced the worst AUC values. Considering both the predictive AUC performance (e.g., best overall **median** value of 76.4) and computation effort, we opted to select CP10 as the encoding method for the second stage experiments.

The results for a fixed balanced training are shown in Table V. In this table, $R = 54$ since 6 C encodings produced computational errors. The table includes the **median** value for each traffic type. For the BEST traffic events, the no balancing step (N) achieved the best AUC results. Balancing the training data seems more useful for the TEST traffic, which makes sense, since it presents a lower ratio of sales. Considering both the AUC and computational effort, for the second phase we selected the N training mode for BEST and S for TEST.

The last analyzed factor is the learning algorithm (Table VI). The number of executed experiments was $E = 5$ (encodings) $\times$ 5 (training setups) $\times$ 2 (variants) = 50 experiments. Three learning algorithms produced computational errors for the C encoding, resulting in a smaller $R = 40$. RF achieved the best overall result, followed by XB, OB and LR (for BEST) and followed by LR, OB and XB (for TEST). OB provided the best online learning AUC results. Yet, under the adopted rolling window scheme, the computational effort is still higher than LR and XB, being only comparable to RF. For the second phase, we selected three methods: RF (best AUC results), LR (second best TEST results), and XB (fastest method, second best BEST results).

B. Second Phase

We tested the first phase selected combinations (CP10 encoding; N training for BEST and S balancing for TEST; LR, XB and RF) in the unseen datasets (1 hour, 1 day and 1 week). This resulted in $E = 2$ (traffic type) $\times$ 2 (data variants) $\times$ 3 (durations) = 12 rolling window executions per classifier. There were no computational errors ($R = 12$).

The results are shown in Table VII in terms of median values for all U rolling window iterations for each classifier and data. In terms of AUC, RF is the best model for the collected setups (BEST and TEST), XG is the best option for BEST and realistic, and LR produces best results for TEST and realistic. The quality of the obtained AUC values can be valued: as good (around 70%) or very good (around 80%) for BEST collected; good for BEST realistic and TEST collected (around 70%); and reasonable (around 60%) for TEST realistic. This last case is particularly relevant, since the marketing company does not have any information about campaign success in TEST traffic and uses a random user ad matching, which is equivalent to an AUC of 50%. Thus, the proposed LR model has a business value.

XG is the fastest method, followed by LR, while RF requires a substantial computation. DSP platforms have real-time requirements, which should be lower than 10 ms for matching users to ads. While we did not use optimized infrastructure and code, several of the XB and LR data-driven models do follow the real-time constrains, even when constantly updating the training model. For instance, for TEST data, LR needs an average of $15.6/3000$=5 ms to issue a prediction, while XG requires a shorter time of 1 ms.

V. Conclusions

There is an increasing interest in the domain of mobile performance marketing due to the massive usage of mobile devices (e.g., smartphones, tablets). Within this industry, Demand-Side Platforms (DSP) act as brokers, matching user traffic, coming from publishers, to advertisers. Acting globally, DSP generate big data related with ad clicks and conversions (product sales). Under this context, user Conversion Rate (CVR) prediction is a critical element of a DSP, allowing to better match user profiles to ads.

In this paper, we study user Conversion Rate (CVR) prediction using big data from a global mobile marketing company. Since the company data center receives big data, with millions of ad clicks and thousands of sales per hour, we design a stream processing engine to collect sample data from the company data center into our computational system. Several datasets with distinct duration times were collected and for two main traffic types: BEST and TEST. Then, we perform an extensive set of CVR prediction tests, under a two stage experimental design and using robust rolling window validation. The first stage explored five categorical transformations, five balanced training setups and six machine learning methods, which were applied to the oldest collected datasets. In the second phase, the best data-driven combinations were then tested on a larger set of unseen datasets.

Interesting predictive performances were achieved, ranging from reasonable (AUC of 61.2% for Logistic Regression and TEST traffic) to very good (AUC of 83.8% for Random Forest and BEST traffic). Thus, there is a potential value for an improved CVR user prediction in the analyzed mobile market. In particular, we achieved an AUC higher than 60% for the realistic TEST scenario, related with new ad campaigns. This is quite valuable for the analyzed DSP, since it currently employs a random user ad matching method for this traffic data type, which corresponds to an AUC of 50%.

Moreover, while we did not use a powerful computational infrastructure or an optimized code, several of the XGboost and Logistic Regression machine learning algorithms did produce real-time results (e.g., <10 ms) when performing several training and testing iterations (e.g., U =56).

In future work, we wish to improve the predictive performance results by putting an increased effort on feature engineering. For instance, by attempting to collect and extract a more richer set of attributes (e.g., related with user behavior after clicking the ad). We also wish to study scalability issues (e.g., use of the Apache Spark cluster-computing framework [21]). Currently, this work is part of an ongoing R&D project that involves a real business company and that will assume, in a later stage, the adaptation of the proposed data-driven approach to perform real-time user ad matches.

Acknowledgments

This article is a result of the project NORTE-01-0247-FEDER-017497, supported by Norte Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, through the European

TABLE IV
RESULTS FOR THE CATEGORICAL ENCODINGS (MEDIAN VALUES, BEST VALUES IN **BOLD**)

		AUC (in %)					Effort (in s)				
Traffic	**Variant**	CP10	IP10	R	C	I	CP10	IP10	R	C	I
	collected	84.52	84.49	**86.22**	50.54	84.66	**53.03**	62.93	54.79	66.61	61.42
BEST	realistic	82.98	85.15	**85.80**	49.80	85.36	**53.13**	82.93	57.99	77.04	63.14
	(R)	(60)	(60)	(60)	(30)	(60)	(60)	(60)	(60)	(30)	(60)
	collected	**64.30**	61.07	59.80	49.96	60.65	42.64	50.06	**40.58**	55.81	46.53
TEST	realistic	**69.47**	67.48	64.61	51.14	65.17	52.05	61.71	**47.85**	59.21	52.65
	(R)	(60)	(60)	(60)	(30)	(60)	(60)	(60)	(60)	(30)	(60)
	median	**76.4**	75.3	74.5	50.4	74.0	52.3	62.6	**50.3**	64.7	55.9

TABLE V
RESULTS FOR THE TRAINING SETUPS (MEDIAN VALUES, BEST VALUES IN **BOLD**)

		AUC (in %)					Effort (in s)				
Traffic	**Variant**	N	B	U	O	S	N	B	U	O	S
	collected	**84.13**	80.45	82.98	80.16	82.07	77.15	70.57	80.18	101.80	**25.13**
BEST	realistic	**83.89**	83.31	83.22	83.09	82.25	69.54	70.72	**62.31**	97.78	82.89
	(R)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)
	median	**84.01**	81.88	83.03	81.63	82.16	73.34	70.64	71.25	99.79	**54.01**
	collected	57.57	63.45	52.82	63.73	**64.55**	61.60	64.58	53.69	86.28	**15.47**
TEST	realistic	65.45	66.03	62.85	**66.17**	63.48	61.37	60.65	58.70	84.32	**47.67**
	(R)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)	(54)
	median	61.51	64.74	57.84	**64.95**	64.02	61.48	62.61	56.20	85.30	**37.57**

TABLE VI
RESULTS FOR THE LEARNING METHODS (MEDIAN VALUES, BEST VALUES IN **BOLD**)

		AUC (in %)						Effort (in s)					
Traf.	**Variant**	RF	LR	XB	OB	DS	RH	RF	LR	XB	OB	DS	RH
	collect.	**91.76**	83.95	89.99	87.81	68.10	67.34	90.26	16.21	**7.44**	116.78	95.13	94.52
BEST	realistic	**89.75**	83.78	88.10	87.83	79.24	67.56	100.61	17.93	**9.66**	127.44	106.43	90.73
	(R)	(40)	(40)	(50)	(40)	(50)	(50)	(40)	(40)	(50)	(40)	(50)	(50)
	median	**90.75**	83.87	89.04	87.82	73.67	67.45	95.44	17.07	**8.55**	122.11	100.78	92.62
	collect.	**74.81**	63.38	60.47	64.87	46.67	55.84	83.76	14.90	**8.15**	87.56	78.64	68.92
TEST	realistic	**76.63**	73.06	65.58	68.16	45.74	59.31	97.14	16.44	**8.13**	89.41	80.20	73.28
	(R)	(40)	(40)	(50)	(40)	(50)	(50)	(40)	(40)	(50)	(40)	(50)	(50)
	median	**75.72**	68.22	63.02	66.52	46.2	57.58	90.45	15.67	**8.14**	88.48	79.42	71.10

Regional Development Fund (ERDF). This work was also supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013.

REFERENCES

[1] S. Silva, P. Cortez, R. Mendes, P. J. Pereira, L. M. Matos, and L. Garcia, "A categorical clustering of publishers for mobile performance marketing," in *International Joint Conference SOCO'18-CISIS'18-ICEUTE'18 - San Sebastián, Spain, June 6-8, 2018, Proceedings*, ser. Advances in Intelligent Systems and Computing, M. Graña, J. M. López-Guede, O. Etxaniz, Á. Herrero, J. A. Sáez, H. Quintián, and E. Corchado, Eds., vol. 771. Springer, 2018, pp. 145–154. [Online]. Available: https://doi.org/10.1007/978-3-319-94120-2

[2] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile networks and applications*, vol. 19, no. 2, pp. 171–209, 2014.

[3] M. Du, R. State, M. Brorsson, and T. Avenesov, "Behavior profiling for mobile advertising," *Proceedings of the 3rd IEEE/ACM Int. Conf. on Big Data Computing, Applications and Technologies (BDCAT)*, pp. 302–307, 2016.

[4] W. Zhang, S. Yuan, J. Wang, and X. Shen, "Real-time bidding benchmarking with ipinyou dataset," *arXiv preprint arXiv:1407.7073*, 2014.

[5] Q. Lu, S. Pan, L. Wang, J. Pan, F. Wan, and H. Yang, "A practical framework of conversion rate prediction for online display advertising," in *Proceedings of the ADKDD'17*. ACM, 2017, p. 9.

[6] W. Zhang, T. Du, and J. Wang, "Deep learning over multi-field categorical data," in *European conference on information retrieval*. Springer, 2016, pp. 45–57.

[7] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter*, vol. 6, no. 1, pp. 20–29, 2004.

[8] L. Tashman, "Out-of-sample tests of forecasting accuracy: an analysis and review," *International Forecasting Journal*, vol. 16, no. 4, pp. 437–450, 2000.

[9] N. Oliveira, P. Cortez, and N. Areal, "The impact of microblogging data for stock market prediction: using twitter to predict returns, volatility, trading volume and survey sentiment indices," *Expert Systems with Applications*, vol. 73, pp. 125–144, 2017.

[10] R Core Team, *R: A Language and Environment for Statistical Computing*, Vienna, Austria, 2016. [Online]. Available: https://www.R-project.org/

[11] K. Banker, *MongoDB in action*. Manning Publications Co., 2011.

[12] G. O. Campos, A. Zimek, J. Sander *et al.*, "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 891–

TABLE VII
SECOND PHASE RESULTS (MEDIAN VALUES, BEST VALUES IN **BOLD**)

			AUC (in %)			Effort (in s)		
Traffic	**Variant**	**Duration** (U)	**LR**	**XG**	**RF**	**LR**	**XG**	**RF**
BEST	collected	1 hour (62)	72.6	**73.5**^{a}	64.9	14.3	**0.6**	58.8
		1 day (99)	79.4	75.0	**83.8**^{b}	19.1	**0.6**	55.8
		1 week (70)	72.4	76.0	**80.3**^{b}	13.2	**0.5**	75.6
		median	74.3	74.9	**77.3**	15.2	**0.6**	62.3
	realistic	1 hour (61)	70.3	**71.3**^{a}	64.5	13.1	**0.5**	55.5
		1 day (60)	71.7	**72.2**^{a}	58.8	15.3	**0.6**	61.8
		1 week (44)	69.2	**71.2**^{a}	58.0	12.0	**0.5**	53.1
		median	70.4	**71.5**	60.0	13.4	**0.5**	56.5
TEST	collected	1 hour (24)	**68.9**	64.5	68.8	**86.4**	88.4	127.1
		1 day (55)	73.4	68.1	**77.2**^{c}	0.4	0.3	1.8
		1 week(24)	67.9	64.1	**68.4**	102.7	93.3	147.0
		median	68.9	65.3	**70.8**	69.0	**67.6**	100.8
	realistic	1 hour (16)	**61.3**	56.0	57.2	17.5	**3.4**	86.0
		1 day (55)	**71.0**	63.9	70.8	9.5	**1.9**	43.0
		1 week (16)	**61.2**	58.3	56.7	18.0	**3.7**	90.5
		median	**63.7**	59.1	60.5	15.6	**3.1**	76.4

a – XG is statistically significant when compared with RF but not LR
b – RF is statistically significant when compared with RF and LR
c – RF is statistically significant when compared with XG but not LR

927, 2016.

[13] N. Lunardon, G. Menardi, and N. Torelli, "ROSE : A Package for Binary Imbalanced Learning," *The R Journal*, vol. 6, no. June, pp. 79–89, 2014. [Online]. Available: https://journal.r-project.org/archive/2014-1/menardi-lunardon-torelli.pdf

[14] L. Torgo, *Data Mining with R, learning with case studies*. Chapman and Hall/CRC, 2010. [Online]. Available: http://www.dcc.fc.up.pt/{~}ltorgo/DataMiningWithR

[15] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[16] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD international conference*. ACM, 2016, pp. 785–794.

[17] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: massive online analysis," *Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010.

[18] P. Cortez, "Data Mining with Neural Networks and Support Vector Machines using the R/rminer Tool," in *10th Industrial Conference on Data Mining*, P. Perner, Ed. Berlin, Germany: LNAI 6171, Springer, Jul. 2010, pp. 572–583.

[19] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, 2006.

[20] M. Hollander, D. A. Wolfe, and E. Chicken, *Nonparametric statistical methods*. John Wiley & Sons, 2013.

[21] X. Meng, J. Bradley, B. Yavuz *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.