International Journal of Information Technology & Decision Making
© World Scientific Publishing Company

International Journal of Information Technology & Decision Making VOL. 22, NO. 02

A Deep Learning-Based Decision Support System for Mobile Performance Marketing

Luís Miguel Matos, Paulo Cortez, Rui Mendes, and Antoine Moreau

https://doi.org/10.1142/S021962202250047X

In Mobile Performance Marketing (MPM), monetary compensation only occurs when an advertisement results in a conversion (e.g., sale of a product or service). In this work, we propose an Intelligent Decision Support System (IDSS) to automatically select mobile marketing campaigns for users. The IDSS is based on a computationally efficient mobile user conversion prediction model that assumes a novel Percentage Categorical Pruning (PCP) categorical preprocessing and an online deep Multilayer Perceptron (MLP) reuse model (MLPr). Using private (non publicly available) business MPM data provided by a marketing company, the MLPr model outperformed an offline multilayer perceptron and a logistic regression, obtaining a high quality class discrimination when applied to sampled (85% to 92%) and complete (90% to 94%) data. In addition, the MLPr compared favorably with other Machine Learning (ML) models (e.g., Random Forest, XGBoost), as well as with other deep neural networks (e.g., diamond shaped). Moreover, we designed two strategies (A — best campaign selection; and B — random selection among the top candidate campaigns) to build the IDSS, in which the predictive deep learning model is used to perform a real-time selection of advertisement campaigns for mobile users. Using recently collected big data (with millions of redirect events) from a worldwide MPM company, we performed a realistic IDSS evaluation that considered three criteria: response time, potential profit and advertiser diversity. Overall, competitive results were achieved by the IDSS B strategy when compared with the current marketing company ad assignment method.

Keywords: Big Data; Categorical Transformation; Classification; Conversion Rate (CVR); Deep Multilayer Perceptron; Intelligent Decision Support System (IDSS).

1. Introduction

The Internet worldwide expansion has created huge marketing opportunities, such as improving marketing campaigns for e-commerce applications[1], obtaining consumer profiles from Facebook[2] and even to detect possible malicious website links[3]. In particular, the massive adoption of 4G or Wi-Fi connected smartphones and tablets has increased the value of Mobile Performance Marketing (MPM). This domain contains specific characteristics that distinguishes it from other digital marketing forms (e.g., Email or social media marketing).[4] Firstly, it is specifically addressed to mobile devices. Secondly, financial remuneration only occurs when an advertisement performs well (e.g., there is a product purchase). Thus, real-time bidding is implemented by using a Demand-Side Platform (DSP) that matches users to ads.[5,4] The DSP is used as a broker between publishers and advertisers. Publishers are web content owners or creators (e.g., online games, news portal) that attract a massive audience of users. The web content is financially supported by using dynamic link ads provided by the DSP. Before accessing the web content, users are required to click the dynamic ad link. Once the link is clicked (redirect event), the DSP diverts the client to a particular advertiser. When there is a conversion (sales event), the DSP returns a percentage of the sale revenue to the publishers. A DSP typically generates big data, due to its worldwide usage and high velocity in which the ad clicks are generated. Under this context, a key DSP issue is the prediction of the user Conversion Rate (CVR), often modeled as a binary classification task and where the goal is to estimate if there will be a purchase when the user clicks a dynamic link and then views an ad.[6,7,8]

Within the mobile performance marketing domain, the user CVR prediction goal is a complex real-world task due to six main reasons. First, it involves big data, since typically millions of clicks are generated every hour. Second, only a tiny amount of user clicks are converted into a sale. Third, and in contrast with other digital marketing CVR prediction tasks (e.g., marketing web page), a limited set of data features is available due to technological constraints and privacy issues associated with the mobile devices and DSP platform. For instance, web page ads often use cookies to store the history of an individual user but this mechanism is not available when working with DSPs and thus it is not possible to identify a single user. Fourth, data features are mostly categorical, often presenting a large cardinality with hundreds or thousands of levels. Fifth, there are market dynamics over time, such as DSP technological changes, addition of new publishers, advertisers and campaigns. Sixth, there are DSP response time constraints (e.g., the DSP analyzed in this paper requires a user ad selection within a time limit of 10 ms).

As detailed in Section 2, the generic digital marketing user CVR prediction task has been mostly modeled using linear models (e.g., logistic regression), assuming the one-hot categorical encoding and static offline learning environments. Also, most studies focus only on classification performance but not on computational effort, which is a critical issue within the mobile performance marketing domain, since DSPs require constant model updates and real-time predictions (due to the already mentioned market dynamics and DSP response constraints). Finally, the majority of the related work addresses only the user CVR prediction task. Thus, there is scarce focus on how the predictive models can be integrated into a DSP, in order to automatically perform user to ad matches and provide a business value.

In Matos *et al.*,[9] a deep multilayer perceptron model was proposed for mobile user CVR prediction, outperforming the logistic regression algorithm when working with sampled data, collected in the year 2018. The model used a newly proposed Percentage Categorical Pruned (PCP) categorical encoding and was adapted to work in dynamic environments (reuse model), allowing constant updates through time. In this paper, we present a rather extended version of our previous work[9] and that covers the whole research performed to design an Intelligent Decision Support System (IDSS)[10] capable of providing value for the MPM domain. In particular, we test the proposed deep learning model with more recent (purely unseen) and complete big data that was collected in the year 2019. Moreover, we integrate the obtained data-driven deep model into an IDSS, allowing to realistically simulate the deep learning usefulness for the MPM domain. More specifically, the proposed deep learning method is adapted for a real-time user ad selection, under two selection strategies: **A** - best campaign selection; and **B** - random selection among the top candidate campaigns. The two IDSS strategies are compared with the ad matching procedure currently adopted by the analyzed DSP (a worldwide mobile marketing company), in terms of three MPM business domain criteria: response time; potential profit, in terms of expected increase of sales; and advertiser diversity, measured in terms of campaign Variety and a proposed True Saved Space indicator. The goal is to show that the proposed IDSS can provide value for the MPM domain by increasing its financial remuneration while maintaining an interesting advertiser diversity, which can benefit several DSP stakeholders (e.g., DSP company, publishers, advertisers).

This paper is organized as follows. Section 2 presents the related works in the field of user CVR prediction. Next, Section 3 details the DSP collected data, categorical transformation, prediction methods and evaluation procedure. Then, Section 4 presents the experimental results. Next, Section 5 discusses the main research and practical implications. Section 6 presents the conclusions and future work directions. Finally, Appendix A presents the list of all acronyms used throughout this work.

2. Related Work

The state-of-the-art works are summarized in Table 1 in chronological order. Each study is characterized in terms of: if the Mobile Performance Marketing (**MPM**) was addressed; the task **Type** (CTR – Click Through Rate or CVR); **Goal** (e.g., BC – Binary Classification); Categorical Processing Method (**CP**); Machine Learning (**ML**) algorithm and type of learning (**Offline** or **Online**); and evaluation procedure (**Eval.**) and **Measure**.

In online advertising, there are two main prediction tasks: Click Through Rate (CTR),[11,8] predicting if an ad link is clicked when a user views a webpage or app; and CVR,[7] estimating if there will be a sale when an ad is viewed. Under the analyzed MPM domain, the CTR probability is always 100% since users need to click an ad before accessing the publisher content (e.g., news portal). Thus, it only useful to perform the user CVR prediction task in this domain. Besides our approach, there is only one study that addresses the MPM domain.[6] It should be noted that DSP Data is very sensitive due to business issues and thus similarly to our approach, Du et al.[6] worked with private (non publicly available) datasets. Turning to the modeling **Goal**, the majority of the related works assume a Binary Classification (BC). In a few cases, a regression modeling (Reg.) is used (e.g., [12,13]), where a regression (numeric output) ML algorithm and measure (e.g., RMSE – Root Mean Squared Error) is adopted to model or evaluate a user CTR or CVR class probability ($\in [0.0, 1.0]$). We highlight that showing only a predictive CTR or CVR value is not enough to demonstrate the utility of the ML models in their application domain. As explained in Section 1, the computational effort and business value of the ML models are two important dimensions that also impact the MPM domain. To measure these dimensions, besides addressing the BC goal, in this work we assume a novel realistic DSP simulation (sim.) that considers our proposed CVR ML model (under two campaign selection schemes). In the related work, there was only one study that also performed a realistic digital marketing simulation,[14] but not for the MPM domain.

Regarding the ML algorithms, the CTR and CVR prediction tasks have been mostly performed by using models that assume a linear correlation between the input features,[15,6] such as linear Poisson regression[16] and Logistic Regression (LR).[14,13,17] Since 2014, more flexible learning methods were proposed, such as: CTR – Gradient Boosting Decision Trees (GBDT)[14]; and CVR – GBDT[18] and Random Forest (RF).[6,13] Due to the remarkable success of deep learning in several competitions (e.g., computer vision, natural language processing)[19], these models were recently proposed for CTR[15,20,11,8,21] and CVR[7,22,21] prediction, under distinct learning architectures: ResNet and Convolutional Neural Networks;[11] deep multilayer perceptrons;[15,20] and Entire Space Multi-Task Model.[7]

Our previous work[9] was the first study that proposed deep learning for mobile performance marketing user CVR prediction. It also addressed other relevant issues that are reflected in Table 1:

- Most CTR or CVR prediction studies tend only to consider prediction classification measures and not the computational effort.[6,15,7] For instance, the deep learning models proposed in[15] are more complex than the LR method, although the classification only improved slightly (e.g., 0.1 percentage points). In contrast, we assessed the obtained deep learning models regarding predictive performance and computational effort.
- Most studies only consider static offline learning scenarios by using a single random (H1) or temporal (T1) train and test holdout split.[14,6,15,7] In contrast, we used a realistic Rolling Window (RW) scheme, which considered several training and test iterations through time, to evaluate the all data-driven models. We have also proposed a reuse learning mode that is more suited for dynamic time changes (Online learning), since it learns from previously trained neural networks.
- Most CTR or CVR works use the popular one-hot encoding (1H) to handle categorical inputs,[14,6,15,7] which heavily increases the computational effort for high cardinality input features. Instead of using the standard one-hot encoding, we pro-

Table 1. Summary of the related work.

Study	**MPM**	**Task**[a]	**Goal**[b]	**CP**[c]	**ML**[d]	**Offline**	**Online**	**Eval.**[e]	**Measure**[f]
16		CTR	BC	1H	LR, LPR	✓		H1	AUC
14	✓	CTR	BC, Sim.	1H	LR, GBDT	✓		T1	AUC, RMSE
6	✓	CVR	BC	1H	LR,RF, NB	✓		T1	AUC
15		CTR	BC	1H, Emb. Layers	MLP, FM, LR	✓		n.d.	AUC
12		CVR	Reg., BC	1H	EC, GBDT	✓	✓	Hn	PR,RC, MAPE
17		CVR	Reg.	n.d.	LR, MF	✓		H1	RMSE,AUC, NDCG
20		CTR	BC	1H	MLP, LR	✓	✓	Hn	AUC, LogLoss
7		CTR, CVR	BC	1H	MLP, GBDT	✓		H1	AUC,F1
11		CTR	Reg.	1H	CNN, MLP	✓		H1	MSE
22		CTR	BC	1H	EM-DL	✓		T1	AUC
21		CTR, CVR	BC	LE	RNN, CNN	✓		T1	AUC, ACC
8		CTR	BC	1H	MLP, AutoGroup	✓		H1	AUC, LogLoss
13		CVR	Reg.	n.d.	LR,RF, DT	✓		H1	RMSE,MAE, ACC,F1
This work	✓	CVR	BC, Sim.	PCP	MLP	✓	✓	RW	AUC,CE,CVR (%) TSS,Entropy

[a] CTR – Click Through Rate; CVR – Conversion Rate.

[b] BC – Binary Classification; Reg. – Regression; Sim. – Simulation.

[c] n.d. – Not Disclosed; LE – Label Encoding; 1H – One-Hot Encoding; PCP – Percentage Categorical Pruning.

[d] AutoGroup – Automatic Feature Grouping; CNN – Convolutional neural network; DT – Decision Tree; FM – Factorisation Machine; EC – Evolutionary Computing; EM-DL – Ensemble Model - Deep Learning CNN + RNN + MLP; GBDT – Gradient Boosting Decision Tree; LPR – Linear Poisson Regression; LR – Logistic Regression; MF – Matrix Factorization; MLP – Multi-layer Perceptrons; NB – Naive Bayes; RF – Random Forest; RNN – Recurrent Neural Network.

[e] n.d. – Not Disclosed; H1 - single random Holdout train and test split; Hn - multiple random Holdout train and test splits; RW – Rolling Window; T1 - single Time ordered holdout train and test split.

[f] ACC – Accuracy; AUC – Area Under Curve; CE – Computational Effort; CVR (%) – Conversion Rate percentage; F1 – F1-Score; MAE – Mean Absolute Error; MAPE – Mean Absolute Percentage Error; MSE – Mean Squared Error; NDCG – Normalized Discounted Cumulative Gain; PR – Precison; RC – Recall; RMSE – Root Mean Squared Error; TSS – True Saved Space

.

posed a new variant termed PCP transform that substantially reduces the memory and computational requirements of the predictive models (see Section 3.2).

While presenting novel features, our previous work[9] contained two main limitations. First, it analyzed only sampled data, collected in 2018 by using a developed data stream engine. The sampled data, termed collected data, contained a higher sales ratio than what would be expected to occur with the real DSP. This issue was handled by creating another dataset, called realistic, with an undersample of the sales events. Nevertheless, such undersampling is synthetic and does not accurately reflect the true data distribution values (see Table 2). Second, and similarly to other related works, it did not realistically measure how the predictive models could be used in a real DSP environment to select mobile advertisements for users. This paper extends our previous work by handling both these limitations (the last row of Table 1 summarizes the novel aspects of our approach). Rather than working with sampled data, we had access to larger and complete datasets, collected in the year of 2019. Furthermore, we perform a realistic DSP simulation that considers our best user CVR prediction model (when using the sampled 2018 data). Using a rolling window scheme, the model is iteratively trained and tested through time using the complete 2019 DSP data. Besides measuring the predictive performance (AUC), we also track the computational effort and relevant business indicators, such as CVR (%) and ad diversity measures (e.g., TSS). In particular, we propose two strategies to make use of the deep learning user CVR prediction model, defining an IDSS that is capable of assigning users to ads in real-time.

3. Materials and Methods

3.1. *Mobile Marketing Data*

We worked with data from OLAmobile, a worldwide mobile performance company responsible for its own DSP. The analyzed DSP records two main event types: redirect, each time a user clicks a dynamic ad; and sales, when there is a conversion. Also, it works under two traffic modes: TEST – used to measure the performance of new campaigns; and BEST – with only the best TEST performing ads and that corresponds to most of the DSP data. We designed a data stream engine that allowed us to collect sampled data during a two week period, starting at 30th May of 2018. Since we worked with sampled data, the sales ratio is much higher than the expected real DSP ratio (e.g., 32% for collected BEST, as shown in Table 2). Therefore, in[9] we handled this issue by creating a realistic dataset, in which we randomly undersample[23] the number of sales obtained in the collected data such that a more realistic ratio is obtained: 1.5% for BEST and 0.5% for TEST traffic. More recently, we had a direct access to the complete from the company via a datacenter available on an Amazon server. The complete data includes all events received by the company between November 15^{th}, 2019, and November 18^{th}, 2019, for the BEST and TEST traffic modes. Table 2 summarizes the main characteristics of the DSP datasets. While related with a smaller time period (four days), the complete data contains a much higher number of examples, with millions of redirects and thousands of sales.

Table 2. Summary of the mobile marketing data.

Fetch method	**Year**	**Period**	**Name**	**Mode**	**Nº no conversions**		**Nº conversions**	
sampled	2018	2 weeks	collected	TEST	290,279	(90.7%)	29,599	(9.3%)
				BEST	328,028	(67.7%)	156,637	(32.3%)
			realistic	TEST	290,279	(99.5%)	1,600	(0.5%)
				BEST	328,028	(98.5%)	4,847	(1.5%)
complete	2019	4 days	complete	TEST	2,283,725	(99.0%)	22,769	(1.0%)
				BEST	4,076,375	(97.2%)	112,532	(2.8%)

Due to technological limitations and privacy concerns, the number of useful features is quite limited in this domain and corresponds to the attributes shown in Table 3. The attributes are related to different entities (column **context**): users, advertisers and publishers. All data attributes are categorical. Some features present a high cardinality (e.g., ad campaign). It should be further noted that DSPs tend to evolve through time, resulting in several data features changes. In the considered DSP, the city attribute was available in 2018 and thus used in[9,24]. Yet, this data attribute was discarded from the DSP in 2019 and thus it is not included in the complete data. As explained in Section 1, the MPM domain includes several dynamics over time, including DSP technological changes, which often impacts on the types of features collected. In our view, the removal of the city attribute from the complete 2019 data does not constitute a comparison study limitation. Rather, it consists of a real-world example of the DSP feature collection dynamics. And if the proposed user CVR model provides consistently high quality prediction results in both datasets (2018 and 2019), then this would be an interesting indication that the model is more robust to DSP changes.

Table 3. Summary of the DSP data attributes.

Context	**Attribute**	**Description** (a – TEST traffic, b – BEST traffic)	**Sampled**	**Complete**
user	country	user country: 198 to 225 levels (e.g., Russia, Spain, Brazil)	✓	✓
	city	user city: up to 13423 levels (e.g., Lisbon, Paris)	✓	✗
	region	region of the country: 23 levels (e.g., Asia, Europe)	✓	✓
	browser	browser name: 14 levels (e.g., Chrome, Safari)	✓	✓
	operator	mobile carrier or WiFi: up to 448 levels (e.g., Vodafone)	✓	✓
advertiser	vertical	ad type: 4 to 12 levels (e.g., video, mainstream, dating)	✓	✓
	campaign	ad product identification: up to 1741 levels	✓	✓
	special	smart link or special offer: up to 1101 levels	✓	✓
publisher	account	publisher type: 8 to 10 levels (e.g., app developer, webmaster)	✓	✓
	manager	publisher account manager: 10 to 34 categorical levels	✓	✓
target	*Y*	if there is a conversion: 2 levels (no, yes)	✓	✓

3.2. *Data Preprocessing*

Most CVR works adopt the one-hot encoding to transform This transform assumes one binary input per categorical level. For instance, the three levels {"a","b","c"} categorical attributes into numeric ones.[6,15] can result in the following one-hot encoding: "a" $\rightarrow$ (1,0,0), "b" $\rightarrow$ (0,1,0) and "c" $\rightarrow$ (0,0,1). However, as explained in Section 2, one-hot encoding creates a vast amount of inputs when the attribute cardinality is high, resulting in more computational effort (in terms of memory and training time) for the machine learning algorithms.

One-hot encoding is a widely used categorical transformation that is very simple to implement and thus it is used in diverse studies concerning CTR or CVR prediction.[16,14,6,15,12,7,11,22,8]

However, high cardinality input features often present a long tail effect.[25,26] Thus, when applied to these features, the one-hot encoding creates computational issues concerning storage and sparseness that are dealt in some studies by using deep autoencoders to preprocess the data.[25] However, these autoencoders introduce a significant overhead since they entail a previous computational training phase (for the data preprocessing step) prior to the application to the predictive learning task. Furthermore, the MPM area is highly volatile and these attributes change quickly, which would require a continuous retraining of the autoencoders. Another alternative is to assume similarities among the string categorical values, such as by adopting a min-hash encoder.[26] Yet, this assumption is only valid in some specific domains (e.g., when encoding a job, the numeric value for "police aide" should be close to the value assigned to a "police officer"), which is not the case of the MPM domain, since it assumes mostly computer generated codes.

In the analyzed mobile marketing domain, there are several high cardinality features that are very sparse. Thus, in [9] we proposed the use of the PCP transform, which is a reduced form of an one-hot encoding. It works by first sorting the feature levels according to their frequency in the training data. Then, the least frequent levels (summing up to a threshold percentage of P) are merged into a single category denoted as "Others". Similarly to the one-hot transform, this category is also used to represent unseen levels in test data. Finally, the one-hot encoding is applied using the reduced set of levels, which includes the most significant levels and the "Others" label.

The goal of the PCP transform is to substantially reduce the input memory and processing requirements while keeping the most relevant levels. For instance, the effect of this preprocessing method is exemplified in Fig. 1 for the campaign attribute, which as a total of 1,268 distinct levels for the TEST traffic. For this attribute and when $P = 10\%$, PCP selects only the most frequent 141 levels (dashed vertical line in Fig. 1), merging the other 1,127 infrequent levels into the "Others" label. Thus, the PCP transform results in a total of 142 binary inputs (141 + "Others"), which is much less than the 1,268 binary inputs required by the standard one-hot transform (reduction of $\frac{1268-142}{1268} = 89\%$).

The advantage of PCP over other encoding methods (e.g., autoencoders) is its simplicity, since it prunes a large number of infrequent values and it does not entail a large preprocessing effort. Moreover, it is a more universal transform (in contrast with the min-hash encoder), since it enforces no constraints on the types of attributes used.

Finally, even though embedding layers were are also considered in CTR works[15], it was used to process textual data, such as comments, reviews or keywords. In our case this approach is not useful since none of the DSP data attributes are related with textual features. As shown in Table 3, we only analyze categorical features (handled by using the PCP transform).

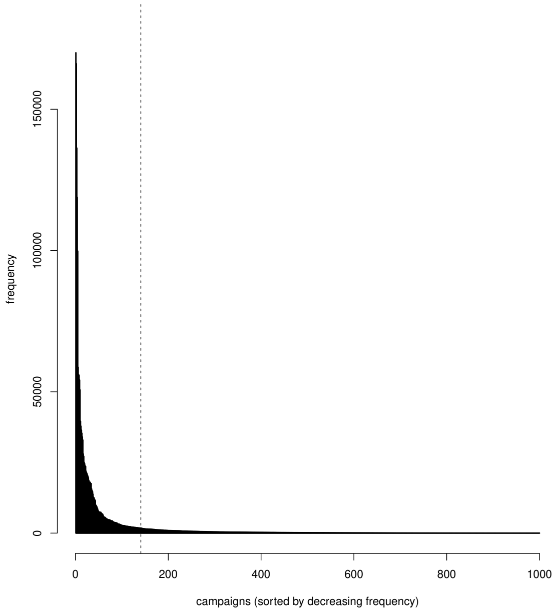

Fig. 1. Example of advertiser campaign most frequent values for TEST traffic (x-axis shows only the 1,000 most frequent levels of a total of 1,268 categorical values; y-axis presents the frequency of the x-axis levels).

3.3. *Multilayer Perceptrons*

In this work, we handle the user mobile marketing CVR prediction as a binary classification task by adopting a Deep Learning model based on Multilayer Perceptron (MLP), also known as Deep Feedforward Neural Network (DFFN).[19] We assume this learning model for the user CVR prediction of the MPM domain because when compared with other binary classification algorithms (e.g., LR, NR, DT), deep learning models (including DFFN) tend to provide better predictive performances when fed with big data.[27,28,29] In effect, deep learning methods have obtained competitive results in a diverse range of classification competitions.[19] The adopted MLP consists of a fully connected network with several hidden layers. Each node in one layer contains only direct weighted connections to the nodes of the next layer. It should be noted that MLP is a popular deep learning architecture for modeling tabular data[15,20,21], which corresponds to our case (see Table 2). While there are other types of Deep Learning architectures, they tend to work better with other input attribute types, such as images (CNN) or temporal data (RNN).

Let $(L_0, L_1, ..., L_H, 1)$ denote a vector with the layer sizes, where $L_0 = I$ is the input layer size, H is the number of hidden layers, and there is one output node. Each MLP node computes:

$$\begin{aligned} z_{k:m,j} &= w_{m:0,j} + \textstyle\sum_{i\in\{1,...,L_{m-1}\}} w_{m:i,j} \cdot a_{k:m-1,i} \\ a_{k:m,j} &= f(z_{k:m,j}) \end{aligned} \tag{1}$$

where $z_{k:m,j}$ denotes the weighted sum of the j-th node of layer m and for example k, $w_{m:i,j}$ the weight connection from node i (of previous layer) to node j (of current layer m), $a_{m,j}$ the activation value for the same node and f the activation function. The $w_{m:0,j}$ weights are known as bias. For the input layer ($m = 0$), $a_{k:0,i} = x_{k,i}$ (the input values).

The design of the MLP structure often involves heuristics and trial-and-error experiments.[19,29] Since the number of inputs is quite large in this domain and there is just one output, in this work we opted to use a a triangular shaped MLP, in which each subsequent layer size is smaller: $L_0 > L_1 > ... > L_H > 1$. A geometrical expression was used to set the triangular shape of the number of neurons in each layer:

$$L_m = L_1 \cdot \alpha^{m-1} \tag{2}$$

where $m \in \{1, ..., H\}$ represents the number of the hidden layer and α denotes a constant multiplier value. The advantage gained is that it is easier to set the full triangular MLP structure by fixing a few parameters: L_1, α and H.

Based on the results of previous experiments, as presented in Table 5, we decided to use the same structure as presented in[9,24]. This structure assumes a large number of layers, which is justified by the theoretical and empirical evidence that depth provides learning benefits.[19,30,28,31,27] Thus, our model has a fixed MLP structure with $L_1 = 1024$, $\alpha = 0.5$ and $H = 8$: $(I, 1024, 512, 256, 128, 64, 32, 16, 8, 1)$. In all hidden layers ($m \in \{1, ..., 8\}$) we used the popular ReLU activation function, due to its fast training and good convergence properties.[19,31]

During the training phase, we used the AdaDelta gradient function,[32] which is an efficient stochastic gradient decent method.[33] We used two approaches to avoid overfitting: dropout and earlystopping. Dropout randomly ignores weighted connections and it was applied on the hidden layers $m = 5$ and $m = 7$ with the values of 0.5 and 0.2. Earlystopping was performed by monitoring the binary crossentropy loss function on a validation set (with 30% of the training data). The training algorithm was stopped when the validation error increased or after a maximum of 100 epochs. Table 4 summarizes the MLP architecture and training setup.

The reset mode follows a standard offline learning procedure where a new MLP model is fully initialized with random weights when new training data is available. In contrast, the proposed reuse approach assumes an online learning approach. Thus, the weights of the previously trained MLP are first stored. As previously explained, new training data often contains unseen input levels (e.g., new ad campaign). If that is the case after preprocessing the data (e.g., one-hot or PCP), then the corresponding new input nodes and weights (using

Table 4. Summary of the main characteristics of the proposed deep MLP.

	L_1	**Hidden layers (H)**	**Dropout layers**	**Dropout values**	**Hidden layer act. function**	**Output layer act.function**	**Structure**
Architecture:	1024	8	2	0.5 (m=5) 0.2 (m=7)	ReLu	Sigmoid	$(I, 1024, 512, 256, 128, 64, 32, 16, 8, 1)$
Training:	**Loss Function**: Binary crossentropy				**Optimizer Function**: Adam		**Number of Epochs**: 100

random initialization) are added to the previously trained MLP model. Fig. 2 exemplifies this procedure by showing the first two MLP layers when one input level (node I from layer $m = 0$) is added under the reuse mode. Next, the whole new MLP is retrained using the AdaDelta gradient function.

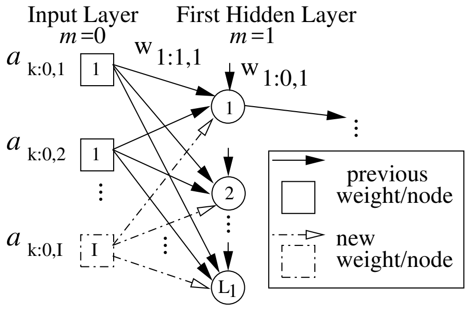

Fig. 2. Example of a new input level added during the reuse mode.

Once an accurate user CVR predictive model is build, it can be used to assign advertisements to users. When a user triggers a redirect event, the publisher and user context is known (Table 3). Thus, the DSP task is to select the "right" advertisement from the set of active advertisements: $A \in \{a_1, a_2, ..., a_{n_A}\}$, where each advertisement (a_i) is defined in terms three attributes: vertical, campaign and special. Let p_i denote the predictive conversion probability for the advertisement a_i. In this work, we devise two main strategies to assign advertisements to users:

A - the best advertisement, set in terms of best the triple (vertical,special,campaign) that, when associated with the current publisher and user context, produces the highest conversion probability ($\arg\max(p_i)$);

B - performs a random selection within the top $B = 10$ best advertisements (the ones with the highest p_i values).

The **A** strategy will result in a higher conversion rates and thus the expected profit. However, it also tends to select a smaller diversity of campaigns, thus being associated with a smaller range of products and advertisers. By including some randomness in the campaign selection procedure, the second strategy (**B**) will increase the diversity of the assignment campaigns, making the DSP less dependent on a just a few advertisers.

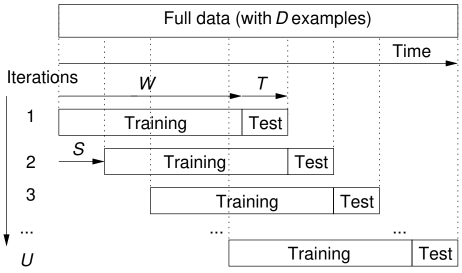

Fig. 3. Schematic of the rolling window procedure.

3.4. *Evaluation*

In this work, we compare several ML models,three that were proposed in[9] (two deep learning modes, namely MLP reset and reuse, and a baseline LR model) and five other ML algorithms commonly adopted by the related work. The learning models were trained and evaluated by using the robust rolling window validation[34], that simulates a real classifier usage through time, with several training iterations and test updates (Fig. 3). In the first iteration ($u = 1$), the classifier model is adjusted to a training window with the W oldest examples, and then predicts T test predictions. Next, in the second iteration ($u = 2$), the training set is updated by discarding the oldest S records and adding S more recent ones. A new model is fit, producing T new predictions, and so on. We assume a sliding window step of ($S = T$), which produces $U = D - (W + T)$ model updates (training and test iterations), where D is the data length (number of examples). To aggregate all $u \in \{1, ..., U\}$ execution results, we compute the median and average of the iteration values.

The analyzed DSP datasets are related with two different time periods (Table 2): two weeks (sampled) and four days (complete). Therefore, the rolling window procedure assumes two setups. For the collected data, the procedure is controlled by the number of data examples, with $W = 100,000$, $T = S = 5,000$, resulting in U =38 to U =75 model updates (depending on the analyzed {collected,realistic} and {TEST,BEST} combination). As for the complete data, we adjusted the rolling window to be controlled by fixed time periods, which is more close to what would occur in a real DSP environment. Thus, the training window corresponds to all training examples produced during a two-day period, while the predictive model is tested on a hourly basis. This leads to $U = 48$ training and testing model updates.

During the execution of the two main rolling window procedures, we store the computational effort (time elapsed, in seconds), which is aggregated by considering average values. The predictive classification performance was measured using the receiver operating characteristic (ROC) curve and the well known Area Under the Curve (AUC) indicator on the rolling window test data.[35] It is common to interpret the quality of the AUC values as[36]: 0.5 – equal to a random classifier; 0.6 – reasonable; 0.7 – good; 0.8 – very good; 0.9 – excellent; and 1 – perfect. The overall AUC value is obtained by computing the median of the rolling window iterations. In this case, we use the median because it is less sensitive to outliers when compared with the average. Moreover, it is adopted by the Wilcoxon non parametric statistic,[37] which is used to check if paired median differences are statistically significant.

The second rolling window experiments, using the complete and more recent DSP data, are also used to simulate the advertisement selection and compare the proposed IDSS strategies (**A** and **B**) with the OLAmobile DSP assignment method (termed **O** strategy). The IDSS strategies use the selected user CVR prediction model to estimate the expected conversion probability p_i for a candidate advertisement a_i. In each iteration of the rolling window, the hourly test data is used to define the set of active advertisements (A, total of n_A distinct values). Moreover, the same test data is used to define the redirect context (user and publisher attributes). We further note that the target output (Y) of the test data corresponds to the **O** strategy selection. The OLAmobile method selects the campaign that achieved the highest average past sales for a user context (the four user attribute values from Table 3). When a user arrives for the first time to the DSP, the selected campaign is the most profitable one, generally the first row of the fixed table. Recurrent visits of the same user, result in moving down on that table, which means moving to the second most profitable campaign, then the third one and so on.

During the advertisement selection experiments, we record the computational effort (in ms). One important restriction of the DSP is that a dynamic link needs to be issued within a maximum response time of 10 ms. We also measure the conversion probability (p_i), assigned to the selected advertisement (a_i). Finally, we compute two diversity measures: Variety and True Saved Space (TSS). Let $X =< x_1, x_2, ..., x_T >$ denote a sequence that contains the assigned advertisements for the T test set user redirects, where $x_i \in \{a_1, ..., a_{n_A}\}$ (e.g., $X =< a_1, a_2, a_1, a_3, a_2, ... >$). The Variety is a simple indicator that counts the number of distinct advertisements that are included in X. As for TSS, it is a newly indicator that is based on the information entropy concept proposed by Claude Shannon[38]:

$$H(X) = -\sum_{i=1}^{n_A} P(X = a_i) \log_2(P(X = a_i)) \tag{3}$$

where $P(X = a_i) \in [0, 1]$ is computed as the proportion of a_i assignments included in X. The entropy computes the number of bits necessary to represent information and it is often used as a diversity measure. The higher the entropy, the more diverse is the X distribution. However, this entropy depends on the number of possible outcomes. In our case, this corresponds to the number of active advertisements (n_A), which varies every hour during the rolling window simulation. To handle this issue, we propose the TSS measure that computes the percentage of saved space when compared with the equally probable assignment method (the most diverse scenario):

$$TSS = 1 - \frac{H(X)}{\log_2(n_A)} \tag{4}$$

where $H(X)$ is the entropy of the analyzed campaign distribution and $\log_2(n_A)$ is the entropy for the equally probable assignment. The lower the TSS value, the higher will be the diversity of the strategy.

4. Results

All experiments were implemented in Python, using the popular Keras library[39]. The categorical processing was executed using a newly implemented Python module, named Categorical Attribute traNsformation Environment (CANE)[40]. We used a 2,3 GHz Intel Core i9 processor, where each classification experiment was executed in a unique core.

4.1. *User CVR prediction results (sampled data)*

In this subsection, we present the initial DSP user CVR prediction results that were obtained when using sampled data from the year of 2018. Table 5 shows the results that were obtained in[9] in terms of predictive accuracy (median AUC of the test sets produced by the rolling window procedure) and computational effort (average preprocess, train and test time, in seconds). Using the sampled datasets, several configurations were compared: two preprocessing methods (1H - one-hot encoding; PCP); two conversion/redirects ratios (collected and realistic); two traffic modes (TEST and BEST); and three machine learning algorithms (LR - Logistic Regression; MLP reset - a new MLP is trained in each rolling window iteration; and MLP reuse – which retrains the previously used MLP). The best overall results were obtained by the PCP MLP reuse model (with the setup presented in Table 4 and henceforth denoted as **MLPr**), allowing to obtain a much faster model (when compared with 1H) while achieving a high quality classification discrimination (AUC values).

Next, we extend the 2018 data user CVR prediction comparison study by considering five additional ML methods often adopted by the related CTR and CVR prediction works (e.g., see Table 1): NB, DT, RF, GBDT and XGBoost (XG). The five ML methods were adopted assuming their default hyperparameter values, as implemented in the scikit-learn and XGBoost Python modules.[41,42] In this comparison, we assume the same rolling window procedure (as executed in Table 5) and consider two types of DSP traffic (TEST and BEST) from the 2018 collected datasets (which includes a larger amount of redirect records). Also, for all ML methods we employ the PCP input categorical transform, in order to provide a fair comparison with the selected PCP MLP reuse method (MLPr). The ML comparison results are presented in Table 6, which confirms that the selected MLPr model is competitive and valuable for the MPM domain. For both traffic types (TEST and BEST), MLPr provides the highest median AUC value with statistically significant differences when compared with all other ML methods (the only exception is for RF and BEST traffic). Moreover, the MLPr model is computationally lighter when compared with RF (second best AUC performing method), requiring around six time less computation. Furthermore, it should be noted that MLPr is the only method in the comparison that is naturally suited for online learning, which is an important aspect given the dynamics of the MPM domain data.

4.2. *Deep learning hyperparameter study (sampled data)*

When assuming deep learning models, there is a large set of hyperparameters that need to be adjusted for a specific task (e.g., adopted neural network structure, learning algo-

Table 5. Results obtained in our previous work (best value per dataset is signaled by using a **boldface** font).

Preprocess	Data	Traffic	Model	Median AUC	Average Effort (s)
1H	collected	TEST	MLP reset	0.90^{ac}	205.45
			MLP reuse (MLPr)	**0.92**^{c}	152.63
			LR	0.72	**142.31**
		BEST	MLP reset	0.88^{ac}	228.26
			MLP reuse (MLPr)	**0.89**^{c}	140.96
			LR	0.78	**132.79**
	realistic	TEST	MLP reset	0.76^{c}	**131.49**
			MLP reuse (MLPr)	**0.88**^{bc}	134.87
			LR	0.51	144.38
		BEST	MLP reset	0.82^{c}	123.04
			MLP reuse (MLPr)	**0.86**^{c}	127.03
			LR	0.53	**124.82**
PCP	collected	TEST	MLP reset	0.88^{ac}	21.12
			MLP reuse (MLPr)	**0.92**^{c}	16.93
			LR	0.67	**16.79**
		BEST	MLP reset	**0.88**^{ac}	21.64
			MLP reuse (MLPr)	**0.88**^{c}	14.98
			LR	0.77	**13.99**
	realistic	TEST	MLP reset	0.77^{c}	**14.56**
			MLP reuse (MLPr)	**0.85**^{bc}	14.75
			LR	0.50	15.68
		BEST	MLP reset	0.80^{c}	**12.62**
			MLP reuse (MLPr)	**0.85**^{c}	12.67
			LR	0.50	12.80

a - statistically significant when compared with MLP reuse.
b - statistically significant when compared with MLP reset.
c - statistically significant when compared with LR.

rithm and its stopping criteria, how to handle overfitting). As explained in Section 3.3, the selection of these hyperparameters is often based on heuristics and trial-and-error experiments.[19,29] The particular setup for the proposed MLPr model was obtained in[9] by executing preliminary trial-and-error user CVR experiments using older sampled data (collected before May 2018). The experiments included different triangular structure and dropout combinations (e.g., $L_1 \in \{1000, 1024\}$, dropout rates $\in \{0.2, 0.3, ..., 0.5\}$). The final MLPr setup was obtained by monitoring the AUC value on the preliminary test sets. We highlight that this MLPr setup provided high quality results for all 4 sampled datasets (e.g., collected TEST) from Table 5. Nevertheless, to further back up the MLPr model selection, in this section we perform an additional hyperparameter study, which compares several

Table 6. Comparison of selected MLP reuse (MLRr) model with other ML methods (best value per traffic type is signaled by using a **boldface** font).

Traffic	Learning Mode	Model	Median AUC	Average Effort (s)
TEST	offline	NB	0.68	**1.62**
		RF	0.88	104.98
		DT	0.82	29.47
		GBDT	0.87	104.19
		XG	0.87	5.16
	online	MLPr	**0.92**[a]	16.93
BEST	offline	NB	0.66	**0.87**
		RF	0.88	88.90
		DT	0.85	5.53
		GBDT	0.86	84.70
		XG	0.86	4.18
	online	MLPr	**0.89**[b]	14.98

a - statistically significant when compared with other ML models.

b - statistically significant when compared with NB, DT, GBDT and XG.

MLP structures and dropout selections. Given the large number of selections involved, and to reduce the computational effort, the comparison assumes the average execution of three runs (training and test evaluations) for the first rolling window iteration data, using 2018 collected data and the PCP transform (as in Table 6).

Table 7 presents the distinct MLP structures that were compared. Column **Model** assumes in general the form MLPH, where H denotes the number of hidden layers. In particular, we have created several triangular MLPs (using Equation 2) with a varying number of hidden layers ($H \in \{1,5,8,11\}$) by fixing several α and L_1 parameters (as shown in Table 7). We also defined a diamond shape MLP $H = 8$ structure (MLP8d), similarly to what was proposed in,[15] which includes three growing hidden layer sizes (first $\alpha = 1.1$) and then four decreasing hidden layers (second $\alpha = 0.6$). The distinct MLPs were trained similarly to the proposed MLPr setup, which included the AdaDelta gradient function, early stopping with 0.5 and 0.2 dropout values applied on the 4th and 6th hidden layers (when available).

The MLP structure comparison results are shown in Table 8, revealing that the proposed model (MLP8) provides the best overall AUC values when considering both traffic types (TEST and BEST). The MLP8d and MLP11 structures also provide interesting predictive results (2 percentage points inferior for the TEST data, similar performance for BEST traffic) but at the expense of slightly higher computational requirements. In effect, more memory is needed to store the MLP8d and MLP11 networks. Also, these models tend to require more computation, although this behavior is not visible for MLP11 and TEST data. We note that due to the early stopping mechanism, it can occur that larger MLPs end their

Table 7. Distinct MLP structures that were compared.

Model	**H**	α	L_1	**MLP structure**
MLP1	1	—	103	$(I, 103, 1)$
MLP5	5	0.25	1024	$(I, 1024, 256, 41, 16, 4, 1)$
MLP8 (**MLPr**)	8	0.5	1024	$(I, 1024, 512, 256, 128, 64, 32, 16, 8, 1)$
MLP8d	8	(1.1, 0.6)	1024	$(I, 1024, 1127, 1240, 1363, 1024, 615, 369, 222, 1)$
MLP11	11	0.65	1024	$(I, 1024, 666, 433, 282, 183, 119, 78, 51, 33, 22, 14, 1)$

training before smaller MLPs.

Table 8. Comparison of MLP structures (AUC and effort values are computed as the average of three runs; best values are signaled by using a **boldface** font).

Traffic	**Model**	**AUC**	**Effort (s)**
TEST	MLP1	0.87	**11.94**
	MLP5	0.87	12.23
	MLP8	**0.90**	21.52
	MLP8d	0.88	22.20
	MLP11	0.88	20.51
BEST	MLP1	0.87	**11.82**
	MLP5	0.87	12.77
	MLP8	**0.88**	18.61
	MLP8d	**0.88**	26.30
	MLP11	**0.88**	19.49

Next, we compare several dropout combinations by performing a 0.1 increase or decrease in the MLPr selected values (0.5 and 0.2). Table 9 presents only the average AUC results, since the computational effort values were similar for all explored combinations (overall average around 20 s). The obtained results show that the selection of the “right” dropout values is not a critical factor that strongly influences the user CVR prediction performance, since several combinations provided similar results. More specifically, the (0.5,0.3) setup obtained the worst performances, while there are two interesting combinations, the proposed (0.5,0.2) setup and also the (0.4,0.3) combination, that resulted in the highest AUC values for both traffic datasets (TEST and BEST).

As a final note, we would like to stress that hyperparameters adopted by the model proposed in this work (MLPr) were selected using preliminary sampled data (collected before May 2018). Without further tuning the MLPr model, it consistently provided high quality prediction results for both traffic types (TEST and BEST) when applied to newer sampled data (collected in May 2018, Table 5) and even more recent complete data (collected in

Table 9. Dropout combination comparison (AUC values are computed as the average of three runs; best values are signaled by using a **boldface** font).

Traffic	Dropout combination	AUC
TEST	(0.4,0.2)	**0.90**
	(0.4,0.3)	**0.90**
	(0.5,0.2)	**0.90**
	(0.5,0.3)	0.86
BEST	(0.4,0.2)	0.87
	(0.4,0.3)	**0.88**
	(0.5,0.2)	**0.88**
	(0.5,0.3)	0.87

November 2019, as shown in Section 4.3). Such consistency attests the MLPr model as a very robust setup for the analyzed DSP domain. This probably occurs due to the AdaDelta training algorithm, which empowers the MLPr model to produce a superior user CVR performance when applied to different DSP dynamics over time. In particular, we highlight the MLPr TEST performances (e.g., Table 6 and Table 8). The analyzed DSP uses a purely random ad selection method when working with this traffic, which results in a much lower conversion rate (CVR%, which is 1.0% for TEST versus 2.8% for BEST, Table 2). By applying sooner our MLPr model, there is a potential to divert more traffic into the BEST mode, thus increasing the overall CVR%.

4.3. *IDSS simulation results (complete data)*

In this section, we present the results of the realistic IDSS simulation described in Section 3.4. Given Table 5 results, we have selected the MLPr model, which was further applied in this paper to the more recent and complete 2019 data. In these experiments, the rolling window assumes fixed time periods, with a two-day training window and an hourly testing. The obtained results are shown in Table 10. The complete datasets involve large training sets. On average, 1.5 and 0.9 millions of records are used to train the BEST and TEST traffic predictive models. Nevertheless, the required computational effort is reasonable, requiring around 162 and 107 seconds to process and train two days of complete DSP data. Moreover, an excellent predictive discrimination level was obtained by the reuse deep learning model (MLPr), obtaining AUC values of 90% (BEST traffic) and 94% (TEST data). We note that the deep learning model was designed in[9] using sampled data from the year of 2018. When applied to the complete and more recent data (year of 2019), the same deep structure kept the same high quality user CVR discrimination level, confirming that the proposed model is robust to dynamic changes.

The MLPr data-driven model was used in the advertisement assignment experiments. The analyzed average number of redirects and active advertisements are shown in Table 10.

Table 10. Rolling window results for the MLPr model and complete DSP data (average values over all $U = 48$ iterations).

	BEST	**TEST**
Preprocess Effort (s)	50.11	30.06
Training Time (s)	111.62	77.15
Training set size	1,573,742	903,213
Test set size	60,778	34,916
Active campaigns (n_A)	107	146
AUC	0.90	0.94

The ad assignment comparison is presented in Table 11. In the table, the Time measure refers to the computational effort (in ms) that is required to process a single redirect and perform an ad assignment. Both IDSS strategies (**A** and **B**) are fast, requiring around 4 ms, which is around half the DSP time limit of 10 ms. The CVR measures the estimated average conversion probability (in %). As expected, the conversion rates are higher for BEST traffic when compared with the TEST data. Overall, the best conversion is provided by the greedy **A** strategy, followed by the second IDSS method (**B**). The CVR differences are statistically significant when compared with the DSP strategy (**O**). The obtained gain is 8 (BEST) and 3 (TEST) percentage points when comparing **A** with **O**, and 5 (BEST) and 2 (TEST) percentage points when comparing **B** with **O**. On the other hand, the diversity measures (Variety and TSS) position the DSP assignment method (**O**) as the more diverse one, followed by the **B** strategy, which is more diverse than **A**.

Table 11. Advertisement assignment comparison (average over all $U = 48$ iterations).

	A		**B**		**O**	
	BEST	TEST	BEST	TEST	BEST	TEST
Time (ms)	3.933	4.183	3.935	4.215	-	-
CVR (%)	12.38	4.47	9.45	3.17	4.06	1.42
Variety	46	29	78	68	440	587
TSS	0.67	0.77	0.45	0.5	0.37	0.32

The final selection of the best DSP ad assignment method requires setting the right trade-off between conversion and diversity. Higher conversion methods will increase the revenue for the advertisers, publishers and DSP company. On the other hand, less diverse methods will reduce the number of advertisers and sold products, while also increasing the irritation of users (e.g., showing often the same ad). The obtained results were shown to the analyzed DSP company, which provided a very positive feedback and favored the **B** strategy as an interesting conversion versus diversity trade-off. Furthermore, in the complete 2019 datasets the TEST traffic corresponds to around 35% of the DSP redirect events. This type of traffic produces a much lower CVR (1.0%, Table 2) given that it is currently associated to a wider range of ads and a random DSP ad assignment method. By using the proposed MLPr model and **B** strategy, there is a strong potential in a real environment to either enhance the TEST CVR rate or to reduce the amount of redirects analyzed under TEST mode.

5. Research and Practical Implications

In terms of research implications, this work demonstrates the usefulness of deep multi-layer perceptrons to model the user CVR prediction task associated with the MPM domain. Indeed, when compared with state-of-the-art ML algorithms (e.g., LR, RF, GBDT, XG), the proposed MLPr obtained better user CVR predictive performances. Furthermore, this study has shown that the proposed PCP categorical encoding is also valuable for the MPM domain, which contains several categorical inputs with a high cardinality. Moreover, the MLPr model can be trained with big data while requiring an affordable computational effort (e.g., 111 s when fitted to 1.5 million DSP records). Also, the fitted MLPr can be adopted for a real-time DSP user ad assignment. In particular, when employing the **B** strategy (random selection among the top candidate campaigns), the proposed MLPr model can produce an interesting campaign conversion rate while maintaining a higher ad selection diversity. As for the newly proposed TSS measure, it consists of a scale independent measure of ad selection variety based on the percentage of saved space. While we only have shown its usefulness for the MPM domain, we believe it is a valuable diversity measure for other marketing application domains.

As for practical implications, we recommend an IDSS for the MPM domain that includes the MLPr model, preprocessed with the PCP transform, and associated with the **B** user ad selection strategy. As shown in this study, such IDSS is expected to highly benefit marketing companies that manage DSPS, producing an potential increased user conversion. Potential beneficiaries include also the other DSP stakeholders: advertisers – since they can increase their sales; publishers – since they will receive a higher revenue for the same user audience; and even users – since they will view ads that are more related with their preferences.

6. Conclusions

In this paper, we have worked with recent MPM big data provided by a marketing company (OLAmobile), conducting several computer simulations for user campaign selection based on deep learning MLP models. Using several business datasets (sampled and complete), collected in distinct years (2018 and 2019), we compared several preprocessing methods, ML algorithms and MLP configurations. Consistent high quality user CVR discrimination results were achieved by the proposed online MLP reuse (MLPr) model. Moreover, the MLPr was adapted to perform real-time advertisement assignments based on two strategies: **A** - selection of the highest conversion probability ad; and **B** - random selection within the top ten candidate ads. When compared with the current DSP company assignment method, the MLPr based **B** strategy obtained a significantly higher user CVR and an interesting ad diversity usage. The obtained results were shown to the MPM company, which provided a positive feedback. Indeed, in the future, the company plans to implement the proposed method in their DSP, which would allow us to attest the conversion and diversity performance in a real-world environment. We also intend to explore eXplainable Artificial Intelligence (XAI) techniques[43,44], and cluster based analytics[45] to better understand and assess the decisions performed by the user CVR predictive models and analyze subpatterns of the data that could be used to infer CVR behaviors and identify potential sales. Also, there is the possibility to implement incremental learning methodologies onto the proposed system similar to Tie *et al.* work[46] given the highly dynamic MPM environment.

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 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. Finally, we wish to thank the anonymous reviewers for their helpful suggestions.

References

1. Robin Marco Gubela, Artem Bequé, Stefan Lessmann, and Fabian Gebert. Conversion uplift in e-commerce: A systematic benchmark of modeling strategies. *International Journal of Information Technology and Decision Making*, 18(3):747–791, 2019.
2. Yu-Jen Hsu Yuh-Jen Chen, Yuh-Min Chen and Jyun-Han Wu. Predicting consumers' decision-making styles by analyzing digital footprints on facebook. *International Journal of Information Technology and Decision Making*, 18(2):601–627, 2019.
3. Tie Li, Gang Kou, and Yi Peng. improving malicious urls detection via feature engineering: Linear and nonlinear space transformation methods. *Information Systems*, 91:101494, 2020.
4. Madhu Bala and Deepak Verma. A critical review of digital marketing. *International Journal of Management, IT & Engineering*, 8(10):321–339, 2018.
5. Susana Silva, Paulo Cortez, Rui Mendes, Pedro José Pereira, Luís Miguel Matos, and Luís Garcia. A categorical clustering of publishers for mobile performance marketing. In Manuel Graña, José Manuel López-Guede, Oier Etxaniz, Álvaro Herrero, José Antonio Sáez, Héctor Quintián, and Emilio Corchado, editors, *International Joint Conference SOCO'18-CISIS'18-ICEUTE'18 - San Sebastián, Spain, June 6-8, 2018, Proceedings*, volume 771 of *Advances in Intelligent Systems and Computing*, pages 145–154. Springer, 2018.
6. Manxing Du, Radu State, Mats Brorsson, and Tigran Avanesov. Behavior profiling for mobile advertising. In Ashiq Anjum and Xinghui Zhao, editors, *Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, BDCAT 2016, Shanghai, China, December 6-9, 2016*, pages 302–307. ACM, 2016.
7. Hong Wen, Jing Zhang, Yuan Wang, Fuyu Lv, Wentian Bao, Quan Lin, and Keping Yang. Entire space multi-task modeling via post-click behavior decomposition for conversion rate prediction. In Jimmy Huang, Yi Chang, Xueqi Cheng, Jaap Kamps, Vanessa Murdock, Ji-Rong Wen, and Yiqun Liu, editors, *Proceedings of the 43rd International ACM SIGIR conference on research*

and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020, pages 2377–2386. ACM, 2020.

8. Huifeng Guo, Bo Chen, Ruiming Tang, Weinan Zhang, Zhenguo Li, and Xiuqiang He. An embedding learning framework for numerical features in ctr prediction. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, Aug 2021.
9. Anonymous Author1, Anonymous Author2, Anonymous Author3, and Anonymous Author4. Using deep learning for mobile marketing user conversion prediction. In *International Joint Conference on Neural Networks, IJCNN 2019 Budapest, Hungary, July 14-19, 2019*, pages 1–8, 2019.
10. David Arnott and Graham Pervan. A critical analysis of decision support systems research revisited: the rise of design science. *Journal of Information Technology*, 29(4):269–293, 2014.
11. Xueting Wang Bohui Xia, Hiroyuki Seshime and Toshihiko Yamasaki. Click-through rate prediction of online banners featuring multimodal analysis. *International Journal of Semantic Computing*, 14(1):71–91, 2020.
12. Quan Lu, Shengjun Pan, Liang Wang, Junwei Pan, Fengdan Wan, and Hongxia Yang. A practical framework of conversion rate prediction for online display advertising. In *Proceedings of the ADKDD'17*, page 9. ACM, 2017.
13. Luis Miralles-Pechuán, M. Atif Qureshi, and Brian Mac Namee. Real-time bidding campaigns optimization using user profile settings. *Electronic Commerce Research*, 2021.
14. Weinan Zhang, Shuai Yuan, Jun Wang, and Xuehua Shen. Real-time bidding benchmarking with ipinyou dataset. *arXiv preprint arXiv:1407.7073*, 2014.
15. Tianming Du Weinan Zhang and Jun Wang. Deep learning over multi-field categorical data - a case study on user response prediction. In *Advances in Information Retrieval - 38th European Conference on IR Research, ECIR 2016, Padua, Italy, March 20-23, 2016. Proceedings*, pages 45–57, 2016.
16. Ye Chen, Dmitry Pavlov, and John F. Canny. Large-scale behavioral targeting. In John F. Elder IV, Françoise Fogelman-Soulié, Peter A. Flach, and Mohammed Javeed Zaki, editors, *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009*, pages 209–218. ACM, 2009.
17. Lili Shan, Lei Lin, and Chengjie Sun. Combined regression and tripletwise learning for conversion rate prediction in real-time bidding advertising. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 115–123, New York, USA, 2018.
18. Quan Lu, Shengjun Pan, Liang Wang, Junwei Pan, Fengdan Wan, and Hongxia Yang. A practical framework of conversion rate prediction for online display advertising. In *Proceedings of the ADKDD'17, Halifax, NS, Canada, August 13 - 17, 2017*, pages 9:1–9:9. ACM, 2017.
19. Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
20. Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, Xiuqiang He, and Zhenhua Dong. Deepfm: An end-to-end wide & deep learning framework for CTR prediction. *CoRR*, abs/1804.04950, 2018.
21. Liyi Guo, Rui Lu, Haoqi Zhang, Junqi Jin, Zhenzhe Zheng, Fan Wu, Jin Li, Haiyang Xu, Han Li, Wenkai Lu, Jian Xu, and Kun Gai. A deep prediction network for understanding advertiser intent and satisfaction. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux, editors, *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 2501–2508. ACM, 2020.
22. Yumin Su, Liang Zhang, Quanyu Dai, Bo Zhang, Jinyao Yan, Dan Wang, Yongjun Bao, Sulong Xu, Yang He, and Weipeng Yan. An Attention-based Model for Conversion Rate Prediction with Delayed Feedback via Post-click Calibration. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages

3522–3528. ijcai.org, 2020.

23. Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004.
24. Anonymous Author1, Anonymous Author2, Anonymous Author3, and Anonymous Author4. Using deep learning for ordinal classification of mobile marketing user conversion. In Hujun Yin, David Camacho, Peter Tiño, Antonio J. Tallón-Ballesteros, Ronaldo Menezes, and Richard Allmendinger, editors, *Intelligent Data Engineering and Automated Learning - IDEAL 2019 - 20th International Conference, Manchester, UK, November 14-16, 2019, Proceedings, Part I*, volume 11871 of *Lecture Notes in Computer Science*, pages 60–67. Springer, 2019.
25. Pau Rodríguez, Miguel A Bautista, Jordi Gonzalez, and Sergio Escalera. Beyond one-hot encoding: Lower dimensional target embedding. *Image and Vision Computing*, 75:21–31, 2018.
26. Patricio Cerda and Gaël Varoquaux. Encoding high-cardinality string categorical variables. *IEEE Transactions on Knowledge and Data Engineering*, 2020.
27. Yann LeCun, Yoshua Bengio, and Geoffrey E. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
28. Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6231–6239, 2017.
29. Andrew NG. *Machine Learning Yearning, Technical Strategy for AI Engineers, In the Era of Deep Learning*. deeplearning.ai, 2020.
30. Shruti Jadon and Ankush Garg. *Hands-On One-shot Learning with Python: Learn to implement fast and accurate deep learning models with fewer training samples using PyTorch*. Packt Publishing Ltd, 2020.
31. Jojo Moolayil. *Learn Keras for Deep Neural Networks*. Apress, 2019.
32. Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.
33. Emmanuel Okewu, Philip Adewole, and Oladipupo A. Sennaike. Experimental comparison of stochastic optimizers in deep learning. In Sanjay Misra, Osvaldo Gervasi, Beniamino Murgante, Elena N. Stankova, Vladimir Korkhov, Carmelo Maria Torre, Ana Maria A. C. Rocha, David Taniar, Bernady O. Apduhan, and Eufemia Tarantino, editors, *Computational Science and Its Applications - ICCSA 2019 - 19th International Conference, Saint Petersburg, Russia, July 1-4, 2019, Proceedings, Part V*, volume 11623 of *Lecture Notes in Computer Science*, pages 704–715. Springer, 2019.
34. L.J. Tashman. Out-of-sample tests of forecasting accuracy: an analysis and review. *International Forecasting Journal*, 16(4):437–450, 2000.
35. T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.
36. Paulo Cortez Pedro José Pereira and Rui Mendes. Multi-objective grammatical evolution of decision trees for mobile marketing user conversion prediction. *Expert Systems with Applications*, 168:114287, 2021.
37. Myles Hollander, Douglas A Wolfe, and Eric Chicken. *Nonparametric statistical methods*. John Wiley & Sons, 2013.
38. Claude E Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.
39. Francois Chollet. *Deep Learning with Python*. Manning Publications Co., USA, 2017.
40. Luís Miguel Matos, João Azevedo, Arthur Matta, André Pilastri, Paulo Cortez, and Rui Mendes. Categorical attribute transformation environment (cane): A python module for categorical to numeric data preprocessing. *Software Impacts*, 2022.
41. Ekaba Bisong. Introduction to scikit-learn. In *Building Machine Learning and Deep Learning*

Models on Google Cloud Platform, pages 215–229. 2019.

42. Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM .
43. Paulo Cortez and Mark J. Embrechts. Using sensitivity analysis and visualization techniques to open black box data mining models. *Inf. Sci.*, 225:1–17, 2013.
44. Alfredo Nascita, Antonio Montieri, Giuseppe Aceto, Domenico Ciuonzo, Valerio Persico, and Antonio Persico. XAI Meets Mobile Traffic Classification: Understanding and Improving Multimodal Deep Learning Architectures. *IEEE Transactions on Network and Service Management*, 18(4):4225–4246, 2021.
45. Tie Lie, Gang Kou, Yi Peng, and Philip S. Yu. an integrated cluster detection, optimization, and interpretation approach for financial data. *IEEE Transactions on Cybernetics*, pages 1–14, 2021.
46. Tie Lie, Gang Kou, Yi Peng, and Philip S. Yu. classifying with adaptive hyper-spheres: An incremental classifier based on competitive learning. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(4):1218–1229, 2020.

Appendix A. Acronyms Used

Tables 12 and 13 present the full list of acronyms used in work.

Table 12. List of acronyms.

Acronym	Description
1H	One-hot encoding
ACC	Accuracy
AutoGroup	Automatic Feature Grouping
AUC	Area Under Curve
BC	Binary Classification
CANE	Categorical Attribute traNsformation Environment
CE	Computational Effort
CTR	Click Through Rate
CVR	Conversion Rate
DFFN	Deep Feedforward Neural Network
DL	Deep Learning
DSP	Demand Side Platform
DT	Decision Tree
EC	Evolutionary Computing
EM	Ensemble Model
F1	F1-Score
FM	Factorization Machine
GBDT	Gradient Boosting Decision Tree
H1	Single random Holdout train and test split
Hn	Multiple random Holdout train and test splits
IDSS	Intelligent Decision Support System
LE	Label Encoding
LPR	Linear Poisson Regression
LR	Logistic Regression
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
ML	Machine Learning
MF	Matrix Factorization
MLP	Multilayer Perceptron
MLP1	Multilayer Perceptron with 1 Layer
MLP11	Multilayer Perceptron with 11 Layers
MLP5	Multilayer Perceptron with 5 Layers
MLP8d	Multilayer Perceptron with 8 Layers Diamond Shaped
MLPr	MLP reuse

Table 13. List of acronyms (Table 12 continued).

Acronym	**Description**
MPM	Mobile Performance Marketing
MSE	Mean Squared Error
n.d.	Not Disclosed
NB	Naive Bayes
NDCG	Normalized Discounted Cumulative Gain
PCP	Percentage Categorical Pruned
PR	Precison
RC	Recall
Reg.	Regression
RF	Random Forest
RMSE	Root Mean Squared Error
ROC	Receiver Operating Characteristic
RNN	Recurrent Neural Network
RW	Rolling Window
Sim.	Simulation
T1	Single Time ordered holdout train and test split
TSS	True Saved Space
XAI	eXplainable Artificial Intelligence
XG	Xgboost