Intelligent Systems with Applications 17 (2023) 200185

Contents lists available at ScienceDirect

Intelligent Systems with Applications

journal homepage: www.journals.elsevier.com/intelligent-systems-with-applications

# Click prediction boosting via Bayesian hyperparameter optimization-based ensemble learning pipelines

Çağatay Demirel [a,b,*], A. Aylin Tokuç [c], Ahmet Tezcan Tekin [d]

[a] Computer Engineering Department, Istanbul Technical University, Maslak, 34467 Sarıyer, Istanbul, Turkey
[b] Donders Institute for Brain, Cognition and Behaviour, Kapittelweg 29, Nijmegen, 6525 EN, Netherlands
[c] Computer Engineering Department, Kadir Has University, Istanbul, Turkey
[d] Management Engineering Department, Istanbul Technical University, Istanbul, Turkey

ARTICLE INFO

*Keywords:*
Machine learning
Ensemble learning
Meta-regressors
Bayesian optimization
Meta-search engines
Online travel agencies

ABSTRACT

Online travel agencies (OTA's) advertise their website offers on meta-search bidding engines. The problem of predicting the number of clicks a hotel would receive for a given bid amount is an important step in the management of an OTA's advertisement campaign on a meta-search engine because bid times number of clicks defines the cost to be generated. Various regressors are ensembled in this work to improve click prediction performance. After preprocessing, the entire feature set is divided into 5 groups, with the training set preceding the test set in the time domain, and multi-set validation is applied. The training data for each validation set is then subjected to feature elimination, and the selected models are next validated with separate ensemble models based on the mean and weighted average of the test predictions. Additionally, a stacked meta-regressor is designed and tested, along with the complete train set, whose click prediction values are extracted in accordance with the out-of-fold prediction principle. The original feature set and the stacked input data are then combined, and level-1 regressors are trained once again to form blended meta-regressors. All individually trained models are then compared pairwise with their ensemble variations. Adjusted $R^2$ score is chosen as the main evaluation metric. The meta-models with tree-based ensemble level-1 regressors do not provide any performance improvement over the stand-alone versions, whereas the stack and blended ensemble models with all other non-tree-based models as level-1 regressors boost click prediction (0.114 and 0.124) significantly compared to their stand-alone versions. Additionally, statistical evidence is provided to support the importance of Bayesian hyperparameter optimization to the performance-boosting of level-1 regressors.

## 1. Introduction

Millions of travellers book hotel accommodations over the Internet each year. Modern travellers rely on peer options, electronic word of mouth (eWOM), and peer reviews. Popular online travel websites offer reliable reviews and prices (Casaló et al., 2015). Therefore, customers choose to inspect and compare different options on meta-search sites like Kayak.com, Trivago, and TripAdvisor before booking their accommodations.

Online travel agencies (OTA's) advertise their website offers on meta-search bidding engines. If the OTA chooses to have a Cost-Per-Click (CPC) ad campaign, the OTA promises to pay a certain amount for each click a certain hotel gets from the platform under predefined conditions. The amount to pay per click is the OTA's *bid* amount. The problem of predicting the number of clicks a hotel would get for a certain bid amount is an important step in the OTA's advertisement campaign management on a meta-search engine, as $bid \times number\,of\,clicks$ defines the cost to be generated.

Predicting hotel searches, clicks, and bookings is a challenging task due to many external factors, such as seasonality, events, location, and hotel-based properties. Capturing such properties increases the accuracy of prediction models. Due to the high variance in daily OTA data, non-linear prediction methods and creating relevant features with a time-delayed data preprocessing approach are adopted in a work trying to forecast daily room sales for each hotel in a meta-search bidding platform (Aras et al., 2019).

Numerous regressor models are trained on large data sets gathered as a result of complex feature engineering, and numerical forecasts are

[*] Corresponding author at: Donders Institute for Brain, Cognition and Behaviour, Kapittelweg 29, Nijmegen, 6525 EN, Netherlands.
*E-mail address:* cagatay.demirel@donders.ru.nl (Ç. Demirel).

https://doi.org/10.1016/j.iswa.2023.200185
Received 30 July 2022; Received in revised form 7 December 2022; Accepted 14 January 2023
Available online 18 January 2023

then made. Although tree-based boosting regressors used as stand-alone ensemble models provide high performance, the performance-boosting effect of ensemble learning (Lei et al., 2010) is increasingly being investigated. It is a machine-learning model combination that gets decisions from various models to enhance the overall performance. The ensemble approach provides the stability and low-variety predictions of machine learning algorithms. It builds a set of decision-makers, namely classifiers and regressors, with various techniques, namely bagging, boosting or Bayesian averaging as final decisions (Dietterich, 2000).

Although ensembling is used as an umbrella term and is realized to be a performance booster in different data domains from a general perspective, it is a structure that is open to investigation due to its inclusivity to as many distinct pipeline-based designs as feasible within itself. In particular, alternative middle-layer feature engineering, and also substantiality of level-1 model characteristics and the impact of hyperparameters on meta-model prediction are open for further research.

The proposed study concentrates on regression-based meta-models because it predicts how many clicks the advertisement will get. A variety of ensemble models is examined for click prediction boosting in a broad perspective; performances of regressors trained as level-1 with their stand-alone versions are compared and the effectiveness of tree-based ensemble models (as an internal ensemble themselves) and non-tree-based models as level-1 regressors are found to be highly distinct. Moreover, the prediction performance of level-1 models with respect to hyperparameter optimization (HPO) is observed and the effectiveness of Bayesian optimization on the click prediction meta-regressors is verified.

## 2. Related work

In the literature, studies are focusing on the problem of predicting the Click-through-rate (CTR) of a sponsored display advertisement to be shown on a search engine related to a query. Click and CTR prediction is an ongoing research for both industry and academia (Fain & Pedersen, 2006, Jansen & Mullen, 2008, Ghose & Yang, 2009). Predicting the number of clicks as an aim of the proposed study is highly related to the CTR prediction problem.

Etsy, an online e-commerce platform, displays promoted search results, which are similar to sponsored search results and our problem with meta-search bidding engines. CTR prediction is utilized in the system to determine the ranking of the ads (Aryafar et al., 2017). The authors found out that different features capture different aspects, so they classified the features as being historical and content-based. They train separate CTR prediction models based on historical and content-based features, separately. Then, these individual models are combined with a logistic regression model. They reported AUC, Average Impression Log Loss, and Normalized Cross-Entropy metrics to compare the models to non-trivial baselines on a large-scale real-world dataset from Etsy, demonstrating the effectiveness of the feature engineering in the proposed study.

Besides, numerous approaches could be seen in the machine learning-based click prediction literature. In one study, state-of-the-art prediction algorithms, Extreme Gradient Boosting (XGBoost) (Chen & Guestrin, 2016) regressor, and a minimum Redundancy-Maximum Relevance (mRMR) (Torralba & Oliva, 2002) feature selection algorithm was executed to predict the daily clicks to be received per hotel, using a large OTA's data from Turkey (Cakmak et al., 2019). The data set received from the meta-search bidding engine contained both numerical and categorical features. The number of clicks as the multiplication of the predicted click-through rate (CTR) and the predicted hotel impression were modelled. The highest R-Squared values obtained in the prediction of individual-hotel-based CTR and impression values are both achieved using XGBoost.

Another study aimed to forecast how many impressions and clicks a hotel will acquire as well as how many rooms it will sell via a meta-search bidding engine (Tekin & Cebi, 2020). The given model predicts how much money an OTA's hotels will make the following day. The authors demonstrate that by incorporating OTA-specific information into prediction models, the generalization of models improves and better results are obtained. They applied XGBoost, random forest, gradient boosting, deep neural networks (DNN), and generalized linear models (GLM) (Nelder & Wedderburn, 1972). The most successful model to predict bookings is gradient boosting, applied on a dataset enriched by features that can summarize the trends in the target variable well.

The demand for hotel rooms in the hotel industry in Turkey between the years 2002-2013 is estimated using Autoregressive integrated moving average (ARIMA) (Box & Pierce, 1970) by Efendioğlu and Bulkan (2017). In their study, they determined the hotel room capacity according to the cost of the unsold rooms and the ARIMA distribution. They also reported that the hotel room demand in the country could be affected by outer factors such as political crises and warnings about terrorism. This work shows the non-deterministic nature of hotel room demand and how unpredictable factors suddenly affect the click prediction problem.

In order to predict ad clicks, Google makes use of logistic regression with improvements in the context of traditional supervised learning based on a Follow-The-Regularized-Leader (FTRL-Proximal) online learning algorithm (McMahan et al., 2013) for better sparsity and convergence. Microsoft's Bing Search Engine proposes a new Bayesian online learning algorithm for CTR prediction for sponsored search (Graepel et al., 2010), which is based on a *probit* regression model that maps discrete or real-valued input features for probability estimation. The scalability of the algorithm is ensured through a principled weight pruning procedure and an approximate parallel implementation. Yahoo adopts a machine learning framework based on Bayesian logistic regression to predict click-through and conversion rates (Chapelle et al., 2015), which is simple, scalable, and efficient. Facebook combines decision trees with logistic regression (He et al., 2014), generating 3% better results in click prediction compared to other methods.

An ensemble model is proposed by Wang et al. to predict the CTR of advertisements on search engines (Wang et al., 2012). Firstly, they tried several Maximum Likelihood Estimation (MLE)-based methods to exploit the training set; including Online Bayesian Probit Regression (BPR) (Smith & LeSage, 2004), Support Vector Machine (SVM), and Latent Factor Model (LFM) (Agarwal & Chen, 2009) and optimized them by selecting the most descriptive features. They have created a rank-based ensemble model using the outputs of BPR, SVM, and MLE. The results are ensembled using harmonic means to generate the final blending submission. The proposed model's output shows an on average 0.013 improvement over the individual models.

Ensemble learning techniques implemented by King et al. to investigate whether they could increase the profitability of pay-per-click (PPC) campaigns (King et al., 2015). They applied voting, bootstrap aggregation (Bagging) (Breiman, 1996), stacked generalization (or stacking) (Zirpe & Joglekar, 2017), and meta cost (Domingos, 1999) techniques to four base classifiers: Naïve Bayes, logistic regression, decision trees, and Support Vector Machines. The research in this work analyzed a data set of PPC advertisements placed on the Google search engine, aiming to classify PPC campaign success. They used average accuracy, recall, and precision metrics to measure the performance of both base classifiers and ensemble models. They also introduced the evaluation metric of total campaign portfolio profit and illustrated how relying on overall model accuracy can be misleading. They conclude that applying ensemble learning techniques in PPC marketing campaigns can achieve higher profits. In another study, Bisht and Susan (2021), probabilistic graphical models such as decision belief are trained with a voting soft-based ensemble model to predict click prediction. It is observed that the voting-based model outperforms the performance of individual models.

Eight ensemble methods were proposed by Ling et al. to accurately estimate the CTR in sponsored search ads (Ling et al., 2017). A single model would lead to sub-optimal accuracy, and the regression models all have different advantages and disadvantages. The ensemble mod-

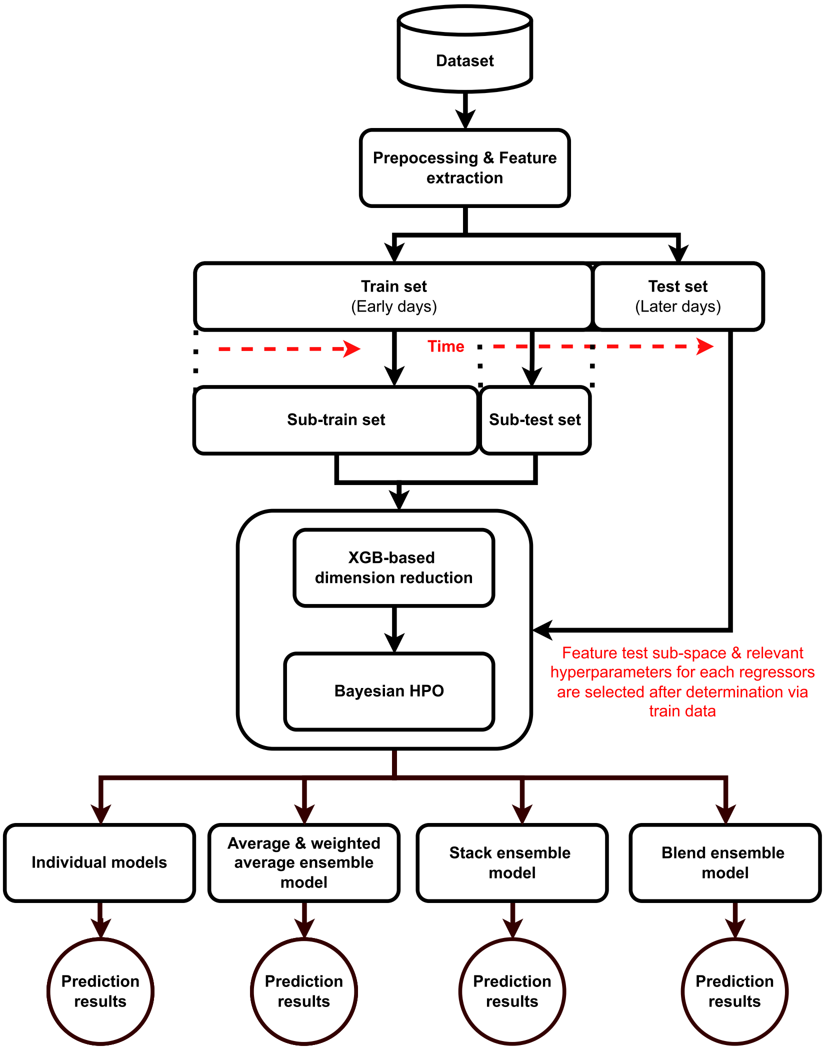

Dataset
Prepocessing & Feature extraction
Train set
(Early days)
Test set
(Later days)
Time
Sub-train set
Sub-test set
XGB-based dimension reduction
Bayesian HPO
Feature test sub-space & relevant hyperparameters for each regressors are selected after determination via train data
Individual models
Average & weighted average ensemble model
Stack ensemble model
Blend ensemble model
Prediction results
Prediction results
Prediction results
Prediction results

**Fig. 1.** Overview of the System. The main train set is divided into two subsets (train and test) to assess the importance of features. These are used to determine the most representative feature subspace by testing with the individual dataset that should be isolated from the actual test set. Accordingly, Bayesian HPO is applied to each individual model via training with a sub-train set. The dimensionality of the main test set is reduced over a predefined feature subspace, and the model is tested over five different model pipelines, including individual ten regressor models, simple averaged and weighted averaged ensemble models, and stack and blend ensemble pipelines.

els are created via bagging, boosting, stacking, and cascading. The training data is collected from historical ads' impressions and the corresponding clicks. The Area under the Receiver Operating Characteristic Curve (AUC) and Relative Information Gain (RIG) metrics are computed against the testing data to evaluate prediction accuracy. They conclude that boosting is better than cascading for the given problem. Boosting neural networks with gradient-boosting decision trees turned out to be the best model in the given setting. They conclude that the model ensemble is a promising direction for CTR prediction; meanwhile, domain knowledge is also essential for ensemble design.

## 3. Proposed system with methods

There are five primary components in the proposed system. The complete system's flow diagram is depicted in Fig. 1. To summarize, queries are used to retrieve the dataset from the database. Preprocessing is used to extract time-domain seasonal decomposition features with suitable data cleaning in the next stage. Individual regressors are then subjected to hyperparameter tuning. After splitting the entire training data into the sub-train sub-test and selecting some of the models ranked according to their validation scores, ensemble models are trained and tested to generate click predictions for each validation set of the total data. Multiple evaluation metrics are extracted from 35 proposed regressors and an adjusted $R^2$ score is used as an evaluation indicator of ensemble models.

### 3.1. Dataset generation and data preprocessing

The data is retrieved from a major OTA company based in Turkey. Contents of the meta-search platform's daily reports are combined with the data retrieved from the OTA. The dataset contains features including bid, average booking value, hotel impression, top position share, cost per click, number of stars of a hotel, rating, gross revenue as numerical, and regions of hotels, and hotel types as categorical features. Some of the columns are eliminated during the data analysis phase, as they contain a high ratio of missing values. In this study, we have replaced the missing values with the most common value and the average of the related feature for categorical and numerical features, respectively.

In addition to OTA's data, some external features are added to the dataset in order to explain the state of the economical and seasonal properties of the environment. Some simple external data examples are daily weather information and daily exchange rates. Data enrichment improves the quality of the dataset. The closeness of the related day to the next public holiday and the length of the holiday are also added as additional numerical variables.

In order to improve the accuracy and generalization ability of the prediction model, additional features are generated from the data following a sliding window (time-delay) approach. Multiple statistical values: the minimum, maximum, average, standard deviation and skewness of numerical values for some specific time periods (such as the last 3, 7, and 30 days) are calculated and used as input features for prediction. The aim of adding such features is to improve the accuracy and generalization ability of the prediction model.

Feature space is enriched with the seasonal decomposition of some time-series features. Seasonal decomposition is a naïve decomposition model that generates additive components by breaking the original feature into three. The output of the algorithm is T: Trend, S: Seasonality, and e: Residual, where $Y[t] = T[t] + S[t] + e[t]$. The seasonal component is first removed by applying a convolution filter to the data. The average of this smoothed series for each period is the returned seasonal component (Avazov et al., 2019). Decomposed seasonality, trend, and residual values are added to the dataset as new features.

As a final step, feature one-hot encoding is proposed for some of the string-based features and binarized. In the last step, the feature set is normalized with min-max scaling to force values to be between 0 and 1.

### 3.2. XGBoost-based recursive feature elimination

XGBoost is part of a gradient-boosting decision tree which operates via the regularization of the tree framework. By using gradient boosting to create the boosted trees and collect the feature scores in an effective manner, each feature's significance to the training model is indicated (Zheng et al., 2017). The calculation of the feature importance of every feature $F_n$ is shown in Eq. (1).

$$F_n(T) = \sqrt{\frac{1}{E}\sum_{e=1}^{E} \hat{i}^2(T_e)} \tag{1}$$

There is a subdivision of each node into two regions at every node $e$ for each feature $n$ as a part of the feature space $F_n$ from a given single decision tree $T$. The maximally forecasted score boosting rate $\hat{i}^2$ represents the metric of squared error shifts of the cost function from the given XGBoost regression outcome of an additive tree $T_e$. The summation of the squared importance over all trees $E$ proposes the summarization of the square importance of the given feature $n$. Accordingly, the root mean squared importance manifests the absolute importance factor of the feature.

The estimation of such an improvement depends on replacing the actual feature value in space with random noise to determine a relative magnitude shift in the final regression performance. Running multiple trees simultaneously provides a better understanding of the average importance of the feature.

In the next step, the customized recursive feature elimination algorithm is used to minimize the feature space (Yan & Zhang, 2015). Algorithm 1 shows the procedure of the flow. The goal is to cover the features ($feature\_subspace$) that represent best the feature importance levels in descending order. To avoid the complexity of the classical recursive-based feature elimination due to the large feature space, the initial feature importance values are considered as bias factors for the features. Given that the randomization factor of the selected features will be auto-biased in the subspace, such a specialization significantly reduces the elimination process. $r^2\_score$ value of a new $feature\_subspace$ is calculated within every iteration until convergence occurs ($r^2\_temp$ value stops being exceeded by $r^2\_score$). Again XGBoost regressor is selected as the feature sub-space evaluator. Classical recursive feature elimination (RFE) (Misra & Yadav, 2020) techniques have also been tried. However, the process takes too long as feature sub-sets are randomly selected in each iteration (within the large feature space). For this reason, it is not preferred in the proposed system.

**Algorithm 1:** Recursive XGBoost dimensionally reduction algorithm.

```
Data:
    $FI = sort\_descending(feature\_importances)$
    $r^2\_temp = 0$
Result:
    $feature\_subspace$
for $FI_0$ in $FI$ do
    $feature\_subspace = feature\_space\,(FI_0 < FI)$
    $r2\_score = XGB\_evaluation(train\_data, test\_data, train\_labels, test\_labels)$
    if $r^2\_score < r^2\_temp$ then
        return $(feature\_subspace)$
    else
        $r^2\_temp = r^2\_score$
    end
end
```

### 3.3. Bayesian HPO

HPO is an essential approach for some machine learning models to enhance prediction performance. There are a few algorithms for tuning hyperparameters. One of them is Grid Search (Bergstra et al., 2011) which tries each combination of given hyperparameter candidates of a model. Another optimization algorithm is known as random search (Karnopp, 1963), which randomly extracts hyperparameter combinations and tries to reach the local optima of a performance score. However, none of them is able to reach successful local optima of performance in a short period. Bayesian HPO (Nguyen, 2019) is a relatively more powerful and efficient algorithm for hyperparameter tuning. It aims to reach a global optimum in a much shorter time than a grid search. There is a probabilistic model of $f(x)$ that aims to be exploited to make decisions about where $X$ is accepted as the next performing function. This procedure helps to find the minimum of non-convex functions in a few epochs, which positively affects the performance. The evaluation metric to rank hyperparameter combinations through input data is $R - squared(R^2)$. $R - squared$ is a statistical measure that represents the proportion of the variance for a dependent variable that's explained by an independent variable or variables in a regression model. The formula of $R^2$ is shown in Eq. (2).

$$R^2 = 1 - \frac{ExplainedVariation}{TotalVariation} \tag{2}$$

$$R^2_{adjusted} = 1 - \left[\frac{(1 - R^2) * (n - 1)}{n - k - 1}\right] \tag{3}$$

Adjusted $R^2$ (Leach & Henson, 2007) (shown in Eq. (3)) is the modified version of $R^2$ where the score is adjusted based on the number of features $k$, and sample size of $n$. Considering that the average $R^2$ values of each model are extracted with multi-set validation, adjusted $R^2$ is selected as an evaluation metric to extract a more robust $R^2$ value, assuming that each set has a different feature space size (feature selection is independent for each validation fold).

### 3.4. Validation score-based model selection

In total, 11 different regression models are selected as candidate models for ensemble pipelines including Lasso (Tibshirani, 1996), Lasso Lars (Efron et al., 2004), Ridge (Hoerl & Kennard, 1970), Bayesian Ridge (Shi et al., 2016), Huber (Sun et al., 2020), Elasticnet (Zou & Hastie, 2005), Linear Regression (Su et al., 2012), XGB, LGBM (Ke et al., 2017), Catboost (Dorogush et al., 2018) and Randomforest (Cutler et al., 2012). The models chosen include ensemble models on their own based on bagging and gradient boosting, as well as linear regression and its regularized variants. Additionally, a model is chosen that uses a probabilistic approach to tackle regression issues, such as the Bayesian Ridge model. However, because the suggested system is not centered on real-time estimates, online learning models are not included. Each training set is divided into a subset of training and test sets and the validation score of each regressor model is ranked. Models with adjusted $R^2$ scores above 0.5 are selected as input to the ensemble models. The selected models are trained from scratch with the whole training set and added as input to the ensemble models. In this way, models that cannot be trained with a certain level of success are automatically eliminated.

### 3.5. Ensembling

If there are M models with errors extracted from the same dataset which are uncorrelated with them, the average error of a model is theoretically reduced by some factor by simply averaging the model outputs. On the other hand, if some of the model outputs have lower performance and are not fit to predict results as well as others, overall error may not be reduced or even increase in some cases.

#### 3.5.1. Average & weighted average of model outputs

The first and most basic ensembling approach is to take an average of various model outputs. There are two different averaging techniques for ensembling. The first one is taking a mean of predicted values. It provides a lower variance of predicted values since different algorithms proceed to predict various aspects of the input data set. The formula for an average of model outputs is shown in Eq. (4).

$$Avg_i = \frac{\sum_r^n pi_r}{n} \tag{4}$$

where $i$ is the $i^{th}$ sample, $r$ is regressor model, $pi_r$ is individual probability of given regressor and $n$ is the number of models used.

However, some machine learning models perform worse than others in terms of prediction, culminating in a poorer overall ensemble prediction performance than some individual regressor prediction performances. The fundamental reason for this is that weak regressors are given the same weight as other ones that provide decent individual performance. As a consequence, while taking an average of all estimations, the weighted average is also utilized in this study to eliminate the detrimental influence of low-performance models. Weights are produced using each model's individual $R^2$ score and scaled between 0 and 1 to standardize the weight of each regressor, ensuring that the sum of all weights is 1. This method allows models that predict higher performance to have a greater impact on the final prediction than models that predict lower performance. The formula of the weighted average of model outputs is shown in Eq. (5).

$$\begin{gathered} Wavg_i = \sum_r w_r * pi_r, \\ r \in R \, for \, i = 1 \, to \, N, \\ \sum_r w_r = 1 \end{gathered} \tag{5}$$

where $r$ is the chosen regressor model, $w_r$ is normalized individual $R^2$ performance of regressor. $pi_r$ is prediction result of regressor $r$ of $i$'th sample and $N$ is the number of models used.

#### 3.5.2. Stack ensemble model

Stack ensemble algorithm assembles individual results for different models to make an intermediate input dataset, and the final model is used to create a final regression result. In the proposed approach, selected models are trained to stack their extracted predictions, and for each validation set, all the stacked inputs are further trained with level-1 regressors. Eleven different stack ensemble models are created for each validation set after each model is trained independently as a level-1 estimator.

Stacking the individual predictions enables the analysis of the intermediate regressor model. The latter is, in turn, used for the linear weight of the results to create a learnable weighted average of provided predictions through each sample of input data. Overall ensemble model variations are indicated in Fig. 2 along with the associations between them.

#### 3.5.3. Blend ensemble model

The Stack ensemble method and the Blend ensemble algorithm (Xie et al., 2013) have similar designs. The separate outcomes of regressor models are assembled in the first stage. Additionally, the individual model outcomes are merged with a dimensionally reduced feature set (feature blending), which adds mediated features extracted as predicted clicks with knowledge of intended predictions to produce an expanded feature dimension. In this regard, the level-1 regressor is subsequently trained independently for every model using the blended dataset.

## 4. Analysis and results

### 4.1. Train & test partition

The whole dataset has a semantic connection in the time domain due to seasonal decomposition features, and for this reason, the test dataset is applied after the training set. This is an alternative multi-set validation instead of cross-validation, as it is thought to give a more realistic result due to the nature of the time series. Accordingly, the entire dataset is split into 5 sub-sets using a window size of 55% of the whole dataset and within each sub-set, distinct data are used as test sets and training sets of constant size immediately preceding the test sets are validated independently. Train-test partitioning scheme is shown in Fig. 3. The entire dataset covers 36 days of curated OTA data.

### 4.2. Validation results

In total, 11 different regression models are selected, of which 4 are decision tree-based ensemble baseline models that are XGB, LGBM, Catboost and Randomforest.

The performance of 35 different regressor models is measured for five validation sets and the evaluation metrics are derived and averaged accordingly. In Table 1, the evaluation metrics ($R^2$, adjusted $R^2$, root mean squared error (RMSE), mean absolute error (MAE)) of all the models are listed in descending order based on the 5-set average of adjusted $R^2$ values. The average test click values for the chosen dataset have a high standard deviation (41.55), which makes the RMSE and MAE errors appear to be increased. However, simplified results are extracted by removing the variation component using the $R2$ and reflectively with adjusted $R2$. Accordingly, blend ensemble with Huber as a level-1 regressor provided the highest performance (0.610) among others. These model results are followed by weighted average and average ensemble models, Lasso with blended ensemble, stacked ensemble with Huber, blended ensemble models with Ridge, Elasticnet and Linear regression as level-1 regressors, Ridge and Bayesian Ridge's stacked ensemble models, blend ensemble with Bayesian ridge, and blend ensemble with XGB (0.610, 0.602, 0.602, 0.600, 0.593, 0.593, 0.592, 0.590, 0.589, and 0.587 respectively). The 13 highest-performing models are all ensemble regressors, and XGB follows close behind with 0.582.

As seen from here, ensemble regressors lose their dominance starting from the 14th highest-performing model. The click prediction rate of stack and blend ensemble models with tree-based level-1 regressors (XGB, LGBM, Randomforest and Catboost) are found to be insignificant compared to their stand-alone versions. When employed as a level-1 regressor, the Catboost model performs better than both the stack and blend ensemble models (0.374, 0.426), especially in its stand-alone version (0.456). Comparing their stand-alone performances to their ensemble counterparts, LGBM, XGB, and Randomforest all rank in the middle of their groups.

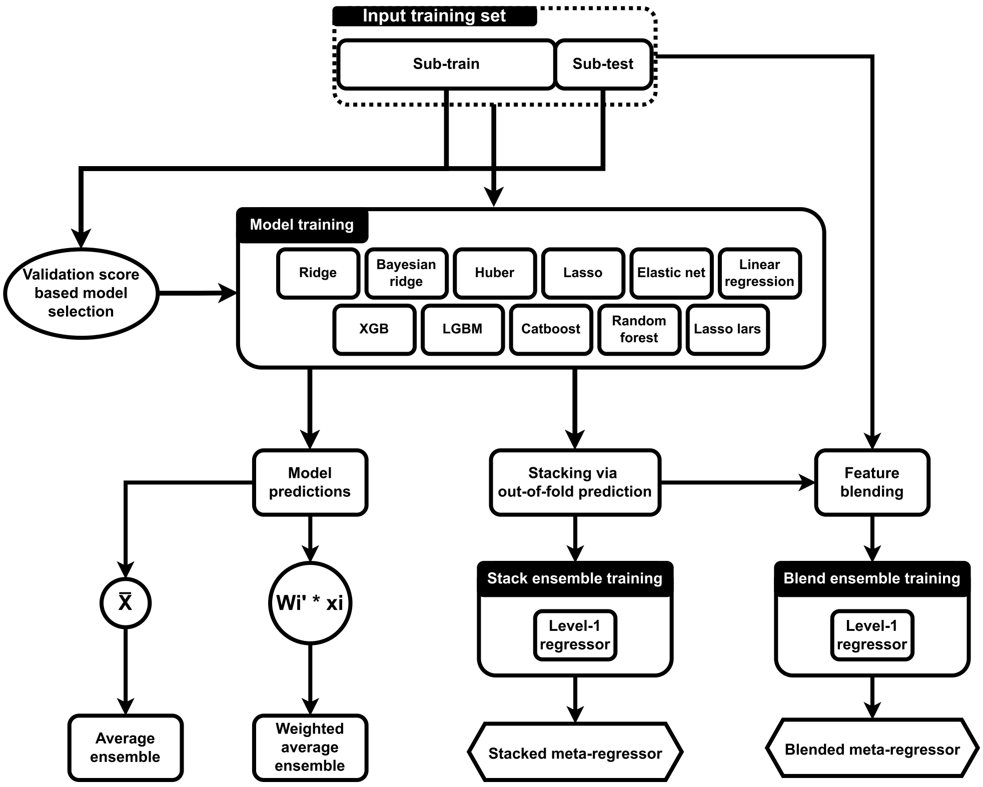


**Fig. 2.** Ensemble model pipelines. The ensemble models are built by averaging and weighted averaging the results for the models chosen from the validation scores. The click predictions of the models are extracted with out-of-fold prediction and stacked to be able to feed the level-1 regressor as a training set. The stacked data set is then blended with the dimensionally reduced training set and used to train the blended meta-regressor.

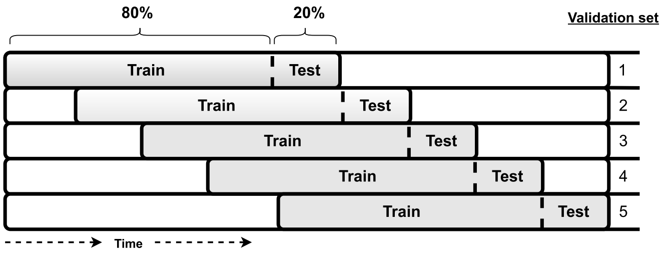


**Fig. 3.** Time-sensitive train-test split scheme for each validation set.

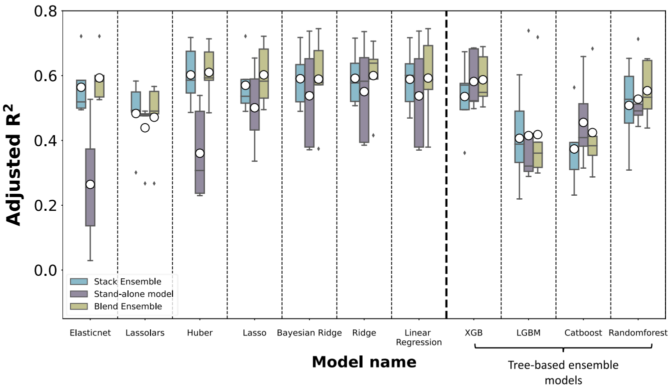


**Fig. 4.** Adjusted $R^2$ performance comparison among model type for each regressor. White circles are defining the average of the multi-set validation scores. Black dots are representing the outliers.

### 4.2.1. *Performance comparisons between ensemble and stand-alone models*

Further, all selected models are divided into two groups (tree-based ensemble and non-tree-based models), and the adjusted $R^2$ validation score distributions of each model are shown in Fig. 4. When utilized as level-1 regressors, it can be seen that all the non-tree-based models chosen perform better than their standalone equivalents. The ensemble variation of any tree-based model, however, does not follow this pattern. Owing to this, the ensemble variations of the two groups—tree-based and non-tree-based models—are statistically compared independently.

To examine the ensemble performance of each model type (tree-based, non-tree-based), one-way repeated measures ANOVA (RMA) within-subject is applied to separate the adjusted $R^2$ values of each validation set for each model into 3 groups, and RMA results are shown in Table 2. Within each model type, the sphericity test is used before RMA, but because the intragroup variance values are discovered to be different, Greenhouse-Geisser (GG) correction is employed on the significance values. As a result, there is no statistically significant difference between stack or blend ensemble models and stand-alone models of the tree-based model type **($p = .119$)**. The ensemble model groups and non-tree-based stand-alone models, however, differ from one another **($p < .001$)**. In order to determine whether groups varied from one another, post-hoc analysis with Bonferroni correction is performed on the adjusted $R^2$ scores of the non-tree-based models. Table 3 demonstrates that, despite the fact that there is no statistically significant difference between the stack and blend ensemble models **($p = 1$)**, the stand-alone model performance is significantly worse than the results of the stack and blend ensemble models (stack ensemble vs. stand-alone: **$p < .001$**; blend ensemble vs. stand-alone: **$p < .001$**). Fig. 5 compares the regression score distributions of both tree-based and non-tree-based models for all validation sets in a paired group setting. It can be seen that tree-based models do not contribute to the model performance as a level-1 regressor. However, in the comparison model contexts, the average score of each model with a non-tree-based level-1 regressor is

**Table 1**
Average validation results for each regressor model.

| Regressor type | Model name | $R^2$ | Adjusted $R^2$ | RMSEs | MAE | Training time [0-1] |
|---|---|---|---|---|---|---|
| Blend ensemble | Huber | 0.611 | 0.610 | 26.514 | 3.074 | 0.789 |
| Weighted average | Voting | 0.609 | 0.608 | 26.619 | 3.082 | 0.454 |
| Model average | Voting | 0.606 | 0.605 | 25.737 | 3.804 | 0.454 |
| Blend ensemble | Lasso | 0.603 | 0.602 | 25.481 | 4.358 | 0.762 |
| Stack ensemble | Huber | 0.603 | 0.602 | 26.916 | 3.103 | 0.771 |
| Blend ensemble | Ridge | 0.601 | 0.600 | 26.138 | 4.082 | 0.771 |
| Blend ensemble | Elasticnet | 0.594 | 0.593 | 25.748 | 4.436 | 0.766 |
| Blend ensemble | Linear regression | 0.593 | 0.593 | 25.999 | 3.889 | 0.768 |
| Stack ensemble | Ridge | 0.593 | 0.592 | 25.903 | 5.271 | 0.761 |
| Stack ensemble | Bayesian ridge | 0.592 | 0.590 | 25.905 | 4.530 | 0.975 |
| Blend ensemble | Bayesian ridge | 0.590 | 0.589 | 26.095 | 4.053 | 0.776 |
| Stack ensemble | Linear regression | 0.590 | 0.589 | 25.931 | 4.624 | 0.777 |
| Blend ensemble | XGB | 0.588 | 0.587 | 27.327 | 3.292 | 0.816 |
| Stand-alone | XGB | 0.583 | 0.582 | 27.534 | 3.263 | 0.038 |
| Stack ensemble | Lasso | 0.571 | 0.570 | 26.319 | 4.832 | 0.768 |
| Stack ensemble | Elasticnet | 0.565 | 0.564 | 26.511 | 4.608 | 0.761 |
| Blend ensemble | Randomforest | 0.555 | 0.554 | 27.508 | 3.368 | 1.000 |
| Stand-alone | Ridge | 0.552 | 0.551 | 26.888 | 3.973 | 0.001 |
| Stand-alone | Bayesian ridge | 0.539 | 0.538 | 27.165 | 3.966 | 0.001 |
| Stand-alone | Linear regression | 0.539 | 0.537 | 27.170 | 3.967 | 0.001 |
| Stack ensemble | XGB | 0.536 | 0.535 | 29.380 | 3.358 | 0.789 |
| Stand-alone | Randomforest | 0.529 | 0.527 | 28.571 | 3.388 | 0.016 |
| Stack ensemble | Randomforest | 0.509 | 0.508 | 29.736 | 3.491 | 0.776 |
| Stand-alone | Lasso | 0.502 | 0.501 | 29.207 | 4.081 | 0.001 |
| Stack ensemble | Lasso Lars | 0.484 | 0.483 | 30.091 | 4.975 | 0.771 |
| Blend ensemble | Lasso Lars | 0.473 | 0.472 | 30.807 | 4.255 | 0.777 |
| Stand-alone | Catboost | 0.457 | 0.456 | 31.356 | 3.567 | 0.089 |
| Stand-alone | Lassolars | 0.440 | 0.439 | 31.408 | 4.229 | 0.001 |
| Blend ensemble | Catboost | 0.426 | 0.424 | 31.940 | 3.588 | 0.873 |
| Blend ensemble | LGBM | 0.420 | 0.418 | 31.636 | 3.748 | 0.779 |
| Stand-alone | LGBM | 0.416 | 0.415 | 31.731 | 3.973 | 0.004 |
| Stack ensemble | LGBM | 0.408 | 0.407 | 32.962 | 3.721 | 0.768 |
| Stack ensemble | Catboost | 0.375 | 0.374 | 33.627 | 3.670 | 0.827 |
| Stand-alone | Huber | 0.362 | 0.361 | 32.162 | 3.688 | 0.012 |
| Stand-alone | Elasticnet | 0.266 | 0.264 | 36.659 | 5.989 | 0.001 |

**Table 2**
RMA within-subject adjusted $R^2$ comparisons among ensemble and stand-alone models for level-1 tree-based and non-tree based regressors.

| Model type | SS | df | MS | F | GG corrected p-value |
|---|---|---|---|---|---|
| Tree-based models | 0.020 | 1.343 | 0.015 | 2.482 | $p = .119$ |
| Non-tree based models | 0.332 | 1.257 | 0.265 | 18.847 | $p < .001$ |

**Table 3**
Post-hoc test for further model type comparisons among non-tree-based level-1 learners and non-tree-based stand-alone models. CI refers to confidence interval.

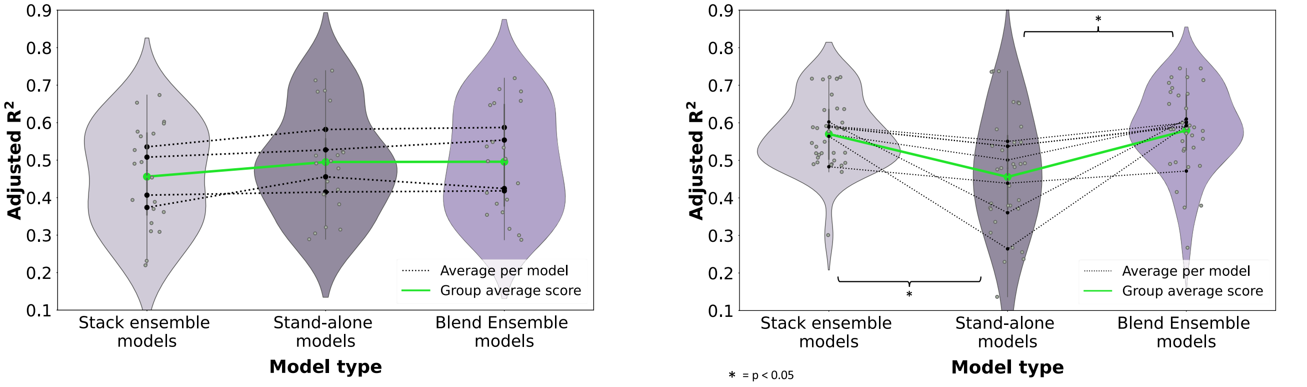
| Group 1 | Group 2 | Mean dif. | 95% CI for mean difference | | SE | t | $P_{Bonferroni}$ |
|---|---|---|---|---|---|---|---|
| | | | Lower | Upper | | | |
| Stack ensemble | Stand-alone | 0.114 | 0.059 | 0.169 | 0.022 | 5.087 | $p < .001$ |
| Stack ensemble | Blend ensemble | -0.010 | -0.065 | 0.045 | 0.022 | -0.434 | $p = 1$ |
| Stand-alone | Blend ensemble | -0.124 | -0.179 | -0.069 | 0.022 | -5.521 | $p < .001$ |

superior to the standalone version. JASP graphical statistical software (Love et al., 2019a) is used for statistical testing.

#### *4.2.2. Effect of HPO applied to level-1 regressor on ensemble model performance*

The effect of HPO on the level-1 regressors (restricted to stack and blend models) is further examined. Accordingly, among all the models, the ones (XGB, LGBM, Catboost, Randomforest, Elastic net and Bayesian Ridge) where multiple hyperparameters can be tuned are chosen for comparison between Bayesian and Randomsearch HPO. The same number of iterations (n = 80) is used for both HPOs. In addition, level-1 regressor performances without any HPO as a baseline factor are also extracted. For the validity of the comparison, the same stacked input layer is used for level-1 model training of all three groups.

As can be seen in Table 4, by applying one-way RMA within-subject comparison between Bayesian HPO, Randomsearch HPO and no HPO groups, it is observed that level-1 learners have a statistically significant effect on ensemble pipeline performance depending on HPO (sphericity is failed and GG corrected $\mathbf{p = .003}$). In the lower part of the same table, groups are subjected to post-hoc analysis applying post-hoc test and bonferroni-corrected significance values are extracted. Although there is an average increase of 0.041, there is no statistical difference between level-1 learners without HPO and level-1 models trained with Randomsearch HPO ($\mathbf{p = .354}$). However, Bayesian HPO performs significantly better than both level-1 learners without HPO and level-1 regressors



(a) Tree-based ensemble model performance comparison with their stand-alone forms

(b) Non-tree-based ensemble model performance comparison with their stand-alone forms

**Fig. 5.** Tree-based and non-tree-based grouped model performance comparison among model types. While all validation sets of each model are included in the distribution, no statistical difference is observed when the adjusted $R^2$ of the tree-based validations are compared pairwise with the stand and blend ensemble models where the same models are used as level-1 learner **(a)**. However, both stack and blend pipelines, where all non-tree-based stand-alone models are used as level-1 learners, achieve significantly higher average performance **(b)**. However, no difference is found between the stack and blend ensemble pipelines for non-tree-based level-1 learners.

**Table 4**
RMA within-subject comparisons of adjusted $R^2$ among HPO types of level-1 regressors in stack & blend ensemble models with additional post-hoc test.

**RMA**

| SS | df | MS | F | GG corrected p-value |
|---|---|---|---|---|
| 0.197 | 1.382 | 0.143 | 8.007 | p = .003 |

| Group 1 | Group 2 | Mean dif. | 95% CI for mean difference | | SE | t | $P_{Bonferroni}$ |
|---|---|---|---|---|---|---|---|
| | | | Lower | Upper | | | |
| No HPO | Random search | 0.041 | -0.023 | 0.105 | 0.026 | 1.583 | p = .354 |
| Random search | Bayesian optimization | -0.065 | -0.128 | -0.001 | 0.026 | -2.490 | p = .045 |
| No HPO | Bayesian optimization | -0.106 | -0.169 | -0.042 | 0.026 | -4.073 | p < .001 |

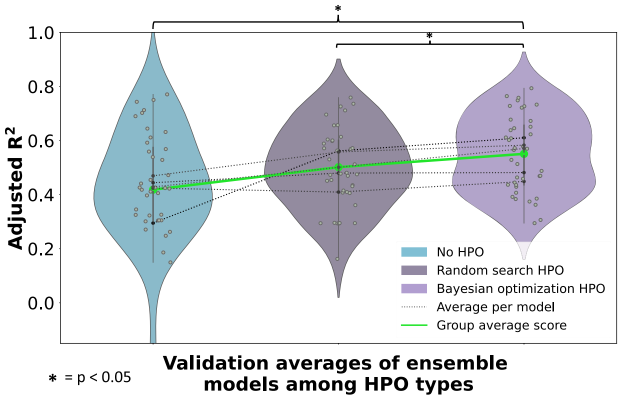


**Fig. 6.** Type of HPO effectiveness on stack & blend ensemble model performances.

trained with Randomsearch HPO (Randomsearch vs. Bayesian HPO: **p = .045**; No HPO vs. Bayesian HPO: **p < .001**). Visual representations of the performance distributions among HPO utilization on level-1 learners are shown in Fig. 6. According to this, averaged $R^2$ group distributions for level 1 learners with No HPO, Random search, and Bayesian optimization HPO are presented from left to right, and the group means are 0.453, 0.494 and 0.559, respectively, showing an upward trend in performance.

### 4.2.3. Ensemble training time assessments for practicality aspects

The normalized training time of each model is also listed in Table 1 (all seconds of training times are divided by the maximum). As a result, all stand-alone models are trained substantially more quickly than ensemble models overall. Before training the entire training set of data, the individual validation scores of each model are first computed. Only the chosen models are trained individually with the whole training set. The ensemble models are awaiting the necessary input data at this time frame. Average and weighted average ensemble models take relatively less to predict results than stack and blend ensemble models since they can reach the result directly based on the selected model predictions. In addition to the stacked prediction input for level-1 regressors, the original feature set and the input space grow in the blending process, so blend ensemble models generally take slightly longer to train than stack variants.

## 5. Discussions

In this work, various meta-regressor modifications that can enhance click prediction are formed, and comparative analyses are conducted to identify strategies for level-1 regressor success. Multiple statistical features are added to the extracted numerical features, substantially expanding the feature vector and requiring faster convergence during feature elimination. The complete feature set is then partitioned into five validation sets. Within each validation set, XGB feature ranking-based recursive dimensionality reduction is applied to select optimal features in the large feature vector with ideal time frame.

Sub-optimal models are selected via validation-based model ranking, and click prediction values of selected models are extracted. In order to estimate the voting-based performances, the average and the weighted average are taken. Additionally, a stacked meta-regressor is built via out-of-fold prediction, which is then applied to the training of several level-1 regressors. In order to produce blended meta-regressors, the same stacked input data is additionally mixed with the original dimensionally reduced feature set and utilized as training data for level-1 models.

All variations of model performances are subjected to comparisons. Accordingly, in the stack and blend ensemble models, the tree-based bagging and boosting regressors do not improve performance at level-1, however other models boost significantly compared to their stand-alone versions. However, it is also discovered that Bayesian HPO outperforms random search or non-optimized models in terms of level-1 regressor performance.

Overall, the findings indicate that level-1 performance is improved by the linear regression variants with L1 & L2 regularizers. It can be seen from the top-ranked blend ensemble with Lasso and Ridge models as regularized linear models contribute the most to the performance of meta-regressors. However there is no noticeable performance difference between the blend and stack ensemble models. The fact that the weighted average ensemble model and the slightly more primitive voting-based average model are among the top performers is notable.

While there may be different reasons why tree-based level-1 regressors do not contribute to the ensemble models this could be speculated that they capture ensemble nuance with multiple decision trees within themselves, and the ineffectivity is understandable for the stacked ensemble model because the intermediate input set is quite low-dimensional compared to the original feature vector for stacked meta-regressors. Thus, it can be expected that they cannot form marginal decision tree combinations relative to each other due to the lack of a diverse vector pool. However, given that the original feature data is also used to train level-1 regressors in blended ensemble models, it would be expected to improve the performance of blended meta-regressors in particular.

In this case, a more unusual restructuring of ensemble models can be considered. The concept of meta-learners is designed to provide the final outcome, yet there are possibilities to convert them into intermediate learners by inducing additional HPO mechanisms or additional meta-feature elimination due to forming the additive judgement on blended predictions on an originally untouched feature dimension (Chen et al., 2022). Articulating meta-learners as mediators would be an inception-based regularizer for intercommunication between multiple meta-models as a single pipeline, which might recalibrate incoming feature space with new model parameters to interact with. This procedure will unquestionably prolong training time, but it might considerably boost the contribution of bagging or gradient boosting based level-1 regressors on ensemble pipelines.

Considering the average training time, it could be seen that meta-regressors are considerably more cost-inefficient than voting-based ensemble models. Still, the model selection before stacking also extends this process. Model selection time can be reduced by training all models in parallel via training multiple models asynchronously via multi-core training pipelines, but while each selected model is stacked separately according to the out-of-fold prediction principle, it still consumes time due to a cross-validation-like fold predictions. If a real-time estimation is not required, practically there is no disadvantage of ensemble methods in estimating the clicks of the next day(s).

## 6. Conclusion

In summary, ensemble model variations are built from chosen regressors, and by comparing these meta-regressor and stand-alone models pairwise, the importance of ensemble approaches to click prediction is highlighted. Although it is noted that level-1 regressors based on tree-based ensemble models do not contribute to the techniques, it is found that level-1 models based on regularized linear regression variations are boosting factors on both stack and blend ensemble meta-regressions. However, using the same hyperparameters as the level-1 regressor, models with the Bayesian HPO on the initial feature set improved the performance of the ensemble model. Future studies will concentrate on the performance enhancement of the blended meta-regressor architecture and multi-level feature blending will be initiated with local feature elimination and HPO on each layer, and the effectiveness of gradient boosting trees on mid-level layers will be reassessed.

## CRediT authorship contribution statement

**Çağatay Demirel:** Conceptualization, Feature engineering, Machine learning analysis, Statistical analyis, Writing-Original draft preparation. **A. Aylin Tokuç:** Feature extraction, Writing-Original draft preparation. **Ahmet Tezcan Tekin:** Data collection, Data curation.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

The authors do not have permission to share data.

## References

Agarwal, D., & Chen, B.-C. (2009). Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '09* (pp. 19–28). New York, NY, USA: Association for Computing Machinery.

Aras, G., Ayhan, G., Sarikaya, M. A., Tokuç, A. A., & Sakar, C. O. (2019). Forecasting hotel room sales within online travel agencies by combining multiple feature sets. In *Proceedings of the ICPRAM 2019 - 8th international conference on pattern recognition applications and method*.

Aryafar, K., Guillory, D., & Hong, L. (2017). An ensemble-based approach to click-through rate prediction for promoted listings at etsy. In *Proceedings of the ADKDD'17*. New York, NY, USA: Association for Computing Machinery.

Avazov, N., Liu, J., & Khoussainov, B. (2019). Periodic neural networks for multivariate time series analysis and forecasting. In *2019 international joint conference on neural networks (IJCNN)* (pp. 1–8).

Bergstra, J. S., Bardenet, R., Bengio, Y., & Kégl, B. (2011). Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, & K. Q. Weinberger (Eds.), *Advances in neural information processing systems: Vol. 24* (pp. 2546–2554). Curran Associates, Inc.

Bisht, K., & Susan, S. (2021). Weighted ensemble of neural and probabilistic graphical models for click prediction. In *2021 the 5th international conference on information system and data mining* (pp. 145–150).

Box, G. E. P., & Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association*, *65*, 1509–1526.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *24*, 123–140.

Cakmak, T., Tekin, A. T., Senel, C., Coban, T., Uran, Z. E., & Sakar, C. O. (2019). Accurate prediction of advertisement clicks based on impression and click-through rate using extreme gradient boosting. In *Proceedings of the ICPRAM 2019 - 8th international conference on pattern recognition applications and method*.

Casaló, L. V., Flavián, C., Guinalíu, M., & Ekinci, Y. (2015). Do online hotel rating schemes influence booking behaviors? *International Journal of Hospitality Management*, *49*, 28–36.

Chapelle, O., Manavoglu, E., & Rosales, R. (2015). Simple and scalable response prediction for display advertising. *ACM Transactions on Intelligent Systems and Technology*, *5*.

Chen, L., Ding, Y., Pirasteh, S., Hu, H., Zhu, Q., Ge, X., Zeng, H., Yu, H., Shang, Q., & Song, Y. (2022). Meta-learning an intermediate representation for few-shot prediction of landslide susceptibility in large areas. *International Journal of Applied Earth Observation and Geoinformation*, *110*, Article 102807.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, KDD '16* (pp. 785–794). New York, NY, USA: ACM.

Cutler, A., Cutler, D. R., & Stevens, J. R. (2012). *Random forests*. Boston, MA: Springer US (pp. 157–175).

Dietterich, T. G. (2000). *Ensemble methods in machine learning*In *Multiple classifier systems* (pp. 1–15). Berlin, Heidelberg: Springer Berlin Heidelberg.

Domingos, P. (1999). Metacost: A general method for making classifiers cost-sensitive. In *Proceedings of the fifth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 155–164).

Dorogush, A. V., Ershov, V., & Gulin, A. (2018). Catboost: Gradient boosting with categorical features support. arXiv preprint, arXiv:1810.11363.

Efendioğlu, D., & Bulkan, S. (2017). Capacity management in hotel industry for Turkey. In *Handbook of research on holistic optimization techniques in the hospitality, tourism, and travel industry* (pp. 286–304). IGI Global.

Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. *The Annals of Statistics*, *32*, 407–499.

Fain, D. C., & Pedersen, J. O. (2006). Sponsored search: A brief history. *Bulletin of the American Society for Information Science and Technology*, *32*, 12–13.

Ghose, A., & Yang, S. (2009). An empirical analysis of search engine advertising: Sponsored search in electronic markets. *Management Science*, *55*, 1605–1622.

Graepel, T., Candela, J. Q. n., Borchert, T., & Herbrich, R. (2010). Web-scale Bayesian click-through rate prediction for sponsored search advertising in Microsoft's bing search engine. In *Proceedings of the 27th international conference on international conference on machine learning, ICML'10* (pp. 13–20). Madison, WI, USA: Omnipress.

He, X., Pan, J., Jin, O., Xu, T., Liu, B., Xu, T., Shi, Y., Atallah, A., Herbrich, R., Bowers, S., & Candela, J. Q. n. (2014). Practical lessons from predicting clicks on ads at Facebook. In *Proceedings of the eighth international workshop on data mining for online advertising, ADKDD'14* (pp. 1–9). New York, NY, USA: Association for Computing Machinery.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*, 55–67.

Jansen, B. J., & Mullen, T. (2008). Sponsored search: An overview of the concept, history, and technology. *International Journal of Electronic Business*, *6*, 114–131.

Karnopp, D. C. (1963). Random search techniques for optimization problems. *Automatica*, *1*, 111–121.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in neural information processing systems: Vol. 30*. Curran Associates, Inc.

King, M. A., Abrahams, A. S., & Ragsdale, C. T. (2015). Ensemble learning methods for pay-per-click campaign management. *Expert Systems with Applications*, *42*, 4818–4829.

Leach, L. F., & Henson, R. K. (2007). The use and impact of adjusted r2 effects in published regression research. *Multiple Linear Regression Viewpoints*, *33*, 1–11.

Lei, S., Xinming, M., Lei, X., & Xiaohong, H. (2010). Financial data mining based on support vector machines and ensemble learning. In *2010 international conference on intelligent computation technology and automation: Vol. 2* (pp. 313–314).

Ling, X., Deng, W., Gu, C., Zhou, H., Li, C., & Sun, F. (2017). Model ensemble for click prediction in bing search ads. In *Proceedings of the 26th international conference on world wide web companion, WWW '17 companion* (pp. 689–698). Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee.

Love, J., Selker, R., Marsman, M., Jamil, T., Dropmann, D., Verhagen, J., Ly, A., Gronau, Q. F., Šmíra, M., Epskamp, S., Matzke, D., Wild, A., Knight, P., Rouder, J. N., Morey, R. D., & Wagenmakers, E.-J. (2019a). Jasp: Graphical statistical software for common statistical designs. *Journal of Statistical Software*, *88*, 1–17.

McMahan, H. B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., Chikkerur, S., Liu, D., Wattenberg, M., Hrafnkelsson, A. M., Boulos, T., & Kubica, J. (2013). Ad click prediction: A view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '13* (pp. 1222–1230). New York, NY, USA: Association for Computing Machinery.

Misra, P., & Yadav, A. S. (2020). Improving the classification accuracy using recursive feature elimination with cross-validation. *International Journal on Emerging Technologies*, *11*, 659–665.

Nelder, J. A., & Wedderburn, R. W. M. (1972). Generalized linear models. *Journal of the Royal Statistical Society. Series A. General*, *135*, 370–384.

Nguyen, V. (2019). Bayesian optimization for accelerating hyper-parameter tuning. In *2019 IEEE second international conference on artificial intelligence and knowledge engineering (AIKE)* (pp. 302–305).

Shi, Q., Abdel-Aty, M., & Lee, J. (2016). A Bayesian ridge regression analysis of congestion's impact on urban expressway safety. *Accident Analysis and Prevention*, *88*, 124–137.

Smith, T. E., & LeSage, J. P. (2004). A Bayesian probit model with spatial dependencies. In *Spatial and spatiotemporal econometrics*. Emerald Group Publishing Limited.

Su, X., Yan, X., & Tsai, C.-L. (2012). Linear regression. *WIREs: Computational Statistics*, *4*, 275–294.

Sun, Q., Zhou, W.-X., & Fan, J. (2020). Adaptive Huber regression. *Journal of the American Statistical Association*, *115*, 254–265.

Tekin, A. T., & Cebi, F. (2020). Click and sales prediction for digital advertisements: Real world application for otas. In C. Kahraman, S. Cebi, S. Cevik Onar, B. Oztaysi, A. C. Tolga, & I. U. Sari (Eds.), *Intelligent and fuzzy techniques in big data analytics and decision making* (pp. 205–212). Cham: Springer International Publishing.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B, Methodological*, *58*, 267–288.

Torralba, A., & Oliva, A. (2002). Depth estimation from image structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*, 1226–1238.

Wang, X., Lin, S., Kong, D., Xu, L., Yan, Q., Lai, S., Wu, L., Chin, A., Zhu, G., Gao, H., et al. (2012). *Click-through prediction for sponsored search advertising with hybrid models*In *KDD workshop*.

Xie, Z., Singh, A., Uang, J., Narayan, K. S., & Abbeel, P. (2013). Multimodal blending for high-accuracy instance recognition. In *2013 IEEE/RSJ international conference on intelligent robots and systems* (pp. 2214–2221).

Yan, K., & Zhang, D. (2015). Feature selection and analysis on correlated gas sensor data with recursive feature elimination. *Sensors and Actuators. B, Chemical*, *212*, 353–363.

Zheng, H., Yuan, J., & Chen, L. (2017). Short-term load forecasting using emd-lstm neural networks with a xgboost algorithm for feature importance evaluation. *Energies*, *10*.

Zirpe, S., & Joglekar, B. (2017). Negation handling using stacking ensemble method. In *2017 international conference on computing, communication, control and automation (ICCUBEA)* (pp. 1–5).

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B, Statistical Methodology*, *67*, 301–320.