Predicting Anonymous Users and Success of Campaigns on Celtra's Platform

Urša Zrimšek

Abstract

In this report we are explaining and evaluating methods for predicting anonymous users that appear in on parts of Celtra's platform that can be used without logging in. We were able to train a Support Vector Classifier that can identify the users with 95.39% accuracy if it is identifying users from the past, and with 92.76% if it is trained on past data and identifies users real-time. We also tried predicting the number of ads that will be interacted with, but we weren't able to do it significantly better than we could do by predicting mean. The best predictions were reached with lasso regression.

Keywords

Celtra, anonymous users, platform usage, success of campaigns

uz2273@student.uni-lj.si, 63200441

Introduction

Celtra is a leading company in the area of automated creative campaigns for digital advertising.

In this report we will model two datasets, provided by Celtra company. One of them contains platform usage data and the other shows us advertisement traffic on Celtra servers.

There are some actions on the platform that users can take without logging in. That results in anonymous users, whose identity we don't know. Our first goal will be to build a model, that identifies the true user that is using the platform anonymously. In each row from the second dataset we can see how many ads were requested by the user, and how many of them had interaction when (if even) they reached the final device to be shown. We will try to build a model that will tell the user how many of the requested sessions will be interacted with.

The calculations that prove the findings from this report and the code to reproduce it can be found in Jupyter notebook `predictive_modeling.ipynb`. Exploration and visualization of both datasets (together with some calculations, that we also used here) are available in repository for Project 2.

Data representation

Celtra company provided an anonymized and sampled dataset on their platform usage data. Let's overview the attributes provided in both datasets.

Platform usage data

Platform usage activity is shown in a data frame with 622078 rows, each represented with 6 columns:

- `ACCOUNT`: unique ID for each company,
- `USER`: unique ID for each company employer,
- `SESSION`: unique ID of session using the platform,
- `ACTIVITYLOCATION`: part of the platform used,
- `ACTIVITY`: coarse grouping of `ACTIVITYLOCATION`,
- `TIMESTAMP`: time of usage.

If we look at unique values in each column of the data frame, we can see that in the 90 days of gathering this data, from 1.7.2020 to 29.9.2020, 11 different companies with 266 users were using Celtra platform, and they altogether participated in 62693 sessions. They used 48 different activity locations, and 10 different activities.

Sessions data

Sessions dataset is represented with 4823186 rows and 15 columns. Some of the columns are explained here:

- `UTCDATE`: UTC date of gathered data,
- `ACCOUNTID`: unique ID for each company,
- `CAMPAIGNID`: unique ID for each campaign,
- `CREATIVEID`: unique ID for each creative,
- `PLATFORM`: platform where ads are shown,
- `SDK`: software environment on device where ads shown.

After each ID column for account, campaign and creative we have a column with creation date of that account/campaign/creative. Last 6 columns of the data frame represent

numeric data about success of the sessions. First of them tells us the number of requests to the server from the users' devices, next one the number of successfully loaded ads and next the number of successfully shown ads. Then we have the number of ads that the internet users interacted with and after that the time (seconds) that the ads were visible to users. The last column tells us the number of ads that were attempted to be loaded, but it is always the same as the number of requested sessions, so we didn't use that column anymore in the further analysis.

More information about data can be found in [1].

Models

In this section we will describe what data we used (and how we prepared it) and what models we chose to tackle the problems of anonymous user and success of the campaigns.

Predicting anonymous users

To prepare this data for building a classifier, we first deleted all the rows in which the user is *(anonymous)*. This is any user that used the platform without logging in, so we have no idea what their real identity is, and that is why we can't use it for train or test set.

To prepare the data for training we transformed the time stamp column into two columns: `DATE`, that tells us the number of day in the year (1-365 or in our case 189-269) and `MIN` that tells us the minute of the day. We also took activity location and account id, and transformed it into binary columns with *one-hot encoding*. We dropped other two columns, since activity is a grouping of activity location, and unique ID's in session don't add any information for most of the users. At least not for those in our train and test set - for anonymous users from our original dataset we saw that there is 19179 sessions (of 217925 anonymous sessions all together) in which a user was anonymous and also logged in. For such sessions we can directly know the identity of the user, so in real life our algorithm should first check if any real user was already logged in in the observed session, and predict with a model otherwise.

Anonymous user was used by all 11 accounts on 7 activity locations. When using those activity locations, users still logged themselves in almost 77% of times. That is why we took 25% of all data that was gathered on this 7 locations for our test set. We separated the data in two different ways - once randomly and once in a point in time, so that we trained the model only on data that was gathered before the time of testing data. In the second case, the data gathered in time of testing data, but on locations on which the user has to be logged in, wasn't used at all. If we compare those two cases to a real life situation, first would be useful if we want to identify the anonymous users from past month (or a longer period), that is why we are testing on data gathered in between training data. The second case is useful when we want to train the model on past data and then identify the users at the moment of using the platform.

We used three different classifier models, first two from `scikit-learn` and last one from `xgboost` library:

- Support Vector Classification.
- Pipeline with 2 stages: Principal Component Analysis, Support Vector Classification.
- Extreme Gradient Boosting.

We first did a Randomized Search for SVC and pipeline, in which we trained this two models with different training parameters on smaller train set of only 20 users (because bigger sets took too much time to train). In table 1 we can see mean accuracies and their standard deviations on 10 different folds for each model. Accuracy is calculated as the portion of correctly classified users. Parameters that we were trying to set are type of kernel, kernel coefficient (only used in rbf kernel) and regularization parameter, respectively, if we look at the first column of table 1. For our second model, that first performs PCA and then trains SVC only on the 5 most important components, we set the number of components to 5 at the start, and then optimized only the previously mentioned parameters of SVC. We can see, that the best parameters (the highest mean accuracy and lowest standard deviation) are the same for both models. Their values are *radial basis function* kernel, kernel coefficient $\gamma = 0.1$ and regularization parameter $C = 5$. We used those for training the models on the data of all users when splitting train and test data randomly.

Table 1. Table of accuracies of models with different parameters, evaluated using 10-fold cross validation.

parameters	model	mean	std
lin, 0.001, 0.1	SVC	0.914199	0.005991
lin, 0.001, 0.1	PCA + SVC	0.979670	0.001526
rbf, 0.001, 1	SVC	0.926060	0.009962
rbf, 0.001, 1	PCA + SVC	0.977526	0.003246
rbf, 0.1, 1	SVC	0.990034	0.001959
rbf, 0.1, 1	PCA + SVC	0.989637	0.001224
rbf, 1, 1	SVC	0.961085	0.006608
rbf, 1, 1	PCA + SVC	0.984594	0.001693
rbf, 0.1, 5	SVC	0.991078	0.001656
rbf, 0.1, 5	PCA + SVC	0.989954	0.001067
rbf, 1, 0.1	SVC	0.660497	0.007683
rbf, 1, 0.1	PCA + SVC	0.923804	0.004131

We did 5-fold cross validation to set the parameters for splitting train and test data in a point in time and got the best results at *radial basis function* kernel, kernel coefficient $\gamma = 0.01$ and regularization parameter $C = 5$. We used those parameters to train the models when splitting train and test data in a point in time.

Predicting the number of sessions with interaction

In [1] we made a dataset that contains all the campaigns that started during data acquisition and also has the information about the number of days that passed until the first active day of the campaign. In Figure 5 we noticed that the average success of a campaign (percentage of requested sessions that were interacted with) could be dependent on the number of days the campaign was already active. In figure 6 we noticed, that the success is dependent on the platform on which the sessions were requested. In the project 2 notebook we can see the graph (5th in the Session visualizations) of average success on certain day of the year. Except for some spikes, it looks like success isn't dependent on day of the year.

That is why, for training the number of sessions predictor, we considered important the number of requested sessions, platform and number of days from the start. We transformed platform values with one-hot encoding, and dropped date and campaign and creative ids, since we want to be able to predict the success also for new campaigns and creatives. We separated train and test data in a point in time, so that our test data contained 20% of all data.

We've tried different models: linear regressions with different regularization techniques, decision trees optimized with different optimizing criterion and K nearest neighbors regression.

We also noticed that there is a lot of rows with low number of requested sessions - more than 30% has less than 10. Those rows are not representative, since one session that was interacted with doesn't show true success of that campaign and the success of such rows has a big variance. We tried training our models only on sessions with 10 requested sessions or more.

Results

Predicting anonymous users

The results of predicting anonymous users were very good, we can see them in table 2. The models are first evaluated on randomly chosen test set and then on test set containing the last part of the data (by point in time). Accuracy is calculated as the percentage of users in test set that were classified correctly. They are compared to a majority classifier, that always predicts the majority user (that occurs most often in train data). When evaluating them, we should bare in mind that even though SVC model has the best accuracy it took by far the most time to train and test. PCA is couple of times faster at training (for accurate timing we would need more isolated conditions than were present during the training) and XGBoost is more than 10 times faster than PCA. We should also consider that the accuracy of XGBoost could be much higher with better choice of training parameters - in this evaluation we only trained it with default parameters.

On figure 1 we can also see one advantage of XGBoost. In contrast to SVC, we can get the importance of features for the trained model. This helps us interprete how the model works and compare different models. We can also see which features

Table 2. Accuracy of models on differently split train/test sets.

Model	Random split	Time split
majority user	7.65%	7.61 %
SVC	95.39%	94.62 %
PCA + SVC	93.53%	92.76%
XGBoost	61.24%	37.82 %

out model considers as unimportant for identifying the user. Such features are printed in the notebook of this project.

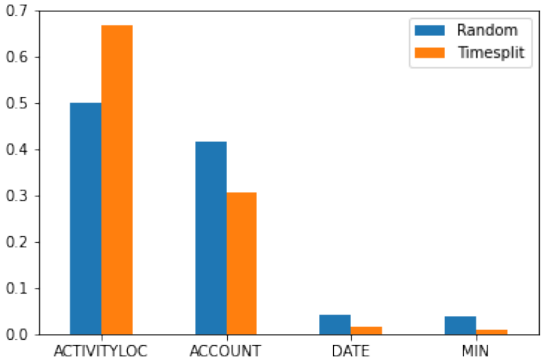

Figure 1. Importance of features in XGBoost model. The features are ordered by their importance and are shown for both XGB models, one trained on randomly chosen set and one on the set split by time. Since our model uses data transformed with one hot encoding, the importance of account and activity location is a sum of importance of each column representing those two attributes. In both models the minute of the day wasn't considered important at all.

Predicting the number of sessions with interaction

In this problem we didn't manage to find a model that improves much from a model that would always predict mean or median of the target variables from test data.

In table 3 we can see the values of the square root of mean squared error of different models. *RMSE* column represents RMSE on whole dataset, and *RMSE-big* represents RMSE on rows with 10 or more requests. First two rows represent models that always predict mean or median, then we have 3 linear regressions with different regularization, after that 3 decision tree models with different optimizing criterion, friedman MSE, MSE and MAE. Decision tree with optimization of MAE is trained only on a sample of our training set, since training on whole dataset takes more than a day. Last one is K-nearest neighbors regressor with $K = 5$.

In the table 3 we can see, that our only improvement with models were the regressions (best of them is lasso regression), but even those didn't bring much better results. If we look at

the size of coefficients in linear regressions we can see that it is probably not over fitting.

Table 3. Table of roots of mean squared error.

Model	RMSE	RMSE-big
mean predictor	295	354
median predictor	296	355
linear regression	287	344
lasso regression	287	344
ridge regression	287	344
decision tree f-mse	379	456
decision tree mse	381	456
decision tree mae	350	435
K nearest neigbours	379	456

Discussion

If training time is not an issue, the best model for identifying anonymous users, proposed to be used in production, is the SVC model. It still uses less than 0.1 second to predict one user, so once it is trained, the time shouldn't be a problem. If we want our model to be more interpretable, we should focus on XGBoost algorithm and try to improve that one. It was added to the analysis later, that is why it is not as optimized as other two models. If you are reproducing this study, we suggest that you start with also optimizing the parameters of XGBoost. When identifying the anonymous user for real, you should first check if there already exists a session that has the same id as the one of anonymous user, because in that case, we would already know the identity with 100% certainty and shouldn't predict it with a model.

We were not able to find a good model to predict the number of sessions with interaction. Since in [1], we saw that there is a connection between success and days the campaign was active, and also between the platform on which the sessions are requested, the reason is probably in the wrong choice of models or in wrong data preparation. Further we should analyze this models and try to figure out why they fail.

References

[1] Urša Zrimšek. Exploration od Celtra's Platform Usage Data. 2020.