RECOMMENDER ALGORITHM CLICK PREDICTION PROJECT

Report for CS7CS4

Robert Trew, Alec Barber Sch., David Scolard
Trinity College Dublin
trewr@tcd.ie, barberal@tcd.ie, dscolard@tcd.ie

November 27, 2019

A brief report on the state of a customer activity prediction in relation to advertisement interaction for Team 66. Included is a synopsis of Team 66's machine learning approaches, methods utilised and results collected. A discussion is presented on novel methods employed with motivation and justifications of their use outlined.

1 Introduction

The task as outlined by the module Kaggle page[1] is one of customer behavior prediction with respect to interaction with advertisements displayed by a host web page. The explained(target) variable in the task is a binary classification of whether or not a customer clicked on an advertisement for a given page visit, i.e. a value of 0 if a customer did not click an advertisement and 1 if they did. The explanatory variables(features) are a set of meta data collected about the webpage, along with some limited information for the target customer (time of interaction, unique visitor i.d. etc.). Importantly, the dataset was collected from multiple distinct websites that host similar genre content, but with potentially different audiences.

Considering the poor correlations of the explanatory variables to the explained variable in the dataset, the task is predominately one of dataset manipulation and adaption (see section 2). Besides this, experimentation on the use of equivalent but distinct explanatory variables was also researched (adopting click through rate or CTR as the explained variable and then using a threshold to decide when to classify as a click or not). Despite all attempts, no satisfying approach was identified that gave significant benefits to prediction in comparison to the all 0s prediction benchmark.

[1] https://www.kaggle.com/c/tcd-ml-comp-201920-rec-alg-click-pred-group/overview

2 Dataset Manipulation and Separation

The Kaggle dataset explanatory variables, as mentioned in section 1, can be described as highly uncorrelated with respect to the explained variable. Table 1 shows a sample of Pearson's Correlation coefficients for the 'JabRef' subset of the dataset. Note that features that are available in the testing set (i.e. not `clicks` and `CTR` etc.) all have a Pearson's correlation of less than 0.1 suggesting an extremely low dependence.

Explanatory Variable	Correlation Coefficient
recommendation_set_id	-0.017551
query_word_count	0.004459
query_char_count	0.002701
query_document_id	-0.016340
year_published	0.002824
number_of_authors	-0.002701
abstract_word_count	0.003643
num_pubs_by_first_author	0.002760
organization_id	NaN
hour_request_received	-0.002558
rec_processing_time	-0.001270
local_hour_of_request	-0.010412
number_of_recs_in_set	-0.003309
recommendation_algorithm_id_used	-0.010303
search_title	0.011859
search_keywords	0.004526
search_abstract	0.000431
clicks (not available in test set)	0.806164
ctr (not available in test set)	0.789273
set_clicked (not available in test set)	1.000000
user_id_num_visits	NaN
hour_request_received_sin	0.003418
hour_request_received_cos	0.001762

Table 1: *Sample Pearson's Correlation Coefficients for explanatory variables with respect to* `set_clicked` *explained variable.*

2.1 New features

In an attempt to generate more useful features, custom explanatory features were developed. The first of which was encoding time data to reflect its cyclic nature[2]. The 'hour_request_received' feature was adapted into two new features, which takes the sin and cos of the scaled time(real and imaginary components, think mapping time to the imaginary unit circle and identifying euclidean components). This has the benefit of relating closely hour 23, to hour 0. Otherwise, the machine learning algorithm would consider hour 23 as the extreme opposite to hour 0.

Another feature added was a count variable. `user_id` is a default feature giving unique identifiers to individual customers visiting the site. A new feature titled `user_id_num_visits` is a count of the number of unique records of each user id. This feature interestingly is non-causal in the sense that employs count data from the test set in the counts (i.e. if user 5 appears in both test and training set, then visits from both are accumulated into one statistic).

Binning is another operation that was applied to many features. Consider the `user_os` feature where there were a high number of low population categories. Binning was applied to make groups of Linux, Windows, IOS and Chrome OS, each of which containing a healthier population size in comparison to the original bins.

2.2 Dataset Separation

After informal investigations into the statistical properties of each website's datasets along with the fact that some websites contained features that the others did not, it was decided that building three separate models on each dataset was prudent (i.e. there was limited transferable knowledge). This had the added benefit of being able to employ feature selection more specifically for each website. Separation was only possible as each sub dataset had a sufficient number of data points to make reasonable models. As discussed further in section 4, better results were recorded using this method, giving some measure of empirical justification for its use.euclidean

A further advantage of dataset separation was the ability to accurately remove features that contained almost all N/A values. An example was the `user_java_version` feature, which in fact only had 7 non-N/A values over all datasets. Often it was found that mostly N/A valued features had a relatively high proportion of useful data for one website (typically 'JabRef') and all N/A values for the others.

2.3 Feature Selection

Pearson's Correlation coefficients were utilised to identify which features had a strong prediction capability in regards to the explained variable. This is useful in that it allows less active features to be dropped and reducing the effects of the curse of dimensionality.

[2] `https://ianlondon.github.io/blog/encoding-cyclical-features-24hour-time/`

2.4 Re-sampling

A major problem encountered when training models was the high imbalance present with regards to the explained variable. For all datasets ensembled, $\sim$1.75% of data points gave a 1 as a output (2.05% for JabRef, 1.04% for Blog and 1.06% for MyVolts). This is problematic as imbalanced datasets have a tendency to introduce bias in predictions. This was found by experimentation, where before any re-sampling was introduced, almost all (if not all) predictions were 0. In an effort to counter this imbalance, SMOTE, an up and down sampling algorithm was employed [3]. SMOTE identifies statistical properties of the minority classification and generates new datapoints so as the dataset is fully balanced. Ideally this should remove the bias of the algorithm without introducing any new information or removing potentially useful majority classification data points (down sampling).

3 Model Training and Adaption

3.1 Classification and Regression

By default, the outlined task is one of binary classification. An alternative approach of changing the explained variable to click through rate (`CTR`) was investigated. This has the added benefit of changing the problem from classification problem to a regression problem. When mapping CTR predictions to `clicked_set` for the submission to the Kaggle site, a simple threshold was adopted. This threshold parameter is key to the benefit of regression as it allows for a method of specifying the importance of recall over precision (i.e. if it is more important to get all positives even if it means getting a lower precision score). Simple classification does not allow for this preference.

A problem with regression approach was the inability for SMOTE to re-sample with a continuous explained variable. A lesser known algorithm SMOGN was adopted [2]. SMOGN performs the same re-sampling task that SMOTE executes on regression tasks.

3.2 Models

The first model used was the scikit-learn[3] implementation of the RandomForest Classifier[4]. This model was chosen as a starting point due to it's robustness as mentioned by Roy *et al.* [4]. Being CPU bound, while providing good results, this model was very slow to train.

The next consideration was the Gaussian Mixture Model[5], as inspiration was taken from the lecture series on unsupervised learning as part of the course. While the model was quick to train, the results produced were not as good as with the Random Forests.

[3] scikit-learn
[4] RandomForestClassifier
[5] Gaussian Mixture Model

In the income prediction challenge, good success was found using models from CatBoost[6], which are enabled for GPU training. The CatBoost Classifier model was used to good effect here. A validation set was used when training the model to try and combat over-fitting. As suggested in the challenge brief, regression is also a valid approach to this problem. The CatBoost Regressor model from the previous challenge was used to predict the *click through rate*, and a threshold of 0.1 was found to give best results. This regression approach was promising, but the Classifier model was still found to be superior. CatBoost Classification models were trained using the Cross-entropy (Log Loss) loss function was used on both the training and validation sets.

Currently all models had been trained on the dataset as a whole. The model classes and framework was then expanded to introduce an individual model for each organisation in the dataset, i.e. JabRef, MyVolts and Blog. This provided initial improvements, with the same processing applied to each provider dataset. It was investigated using curated feature processing per provider, but little joy was to be found.

When testing model performance off-line, the following metrics were used:

- F1 Score
- Mathews Correlation Coefficient [1]

In the end, the best performing model multiple RandomForest Classifiers, with no special pre-processing to the datasets, beyond what was designed for handling the full dataset. This design achieved a score of 0.78116 on the private leaderboard, by predicting 57 positives.

4 Results and Discussion

In general, with respect to the F1 metric, the performance of the model can be categorised as limited. A score of 0.77798 can be achieved by simply submitting all 0s[7]. In comparison, our best result was 0.78053(on public dataset). This is a relative improvement of only 0.00216. This suggests that given the current dataset, there is very little that can be done to make useful predictions. Further from this point, it can be speculated that the factors that contribute to a person clicking a title are clearly not shown in the feature space of the dataset. For the most part, information supplied is generic and specific to the article. It would be expected that data on the user themselves (age, interactions with adds in the past etc.) would have a far greater prediction power than information on the browser JavaScript version etc.

Good performance was recorded when using dateset separation for each website. As outlined early in this report, each website had significantly different statistical properties (jabref for example

[6] catboost.ai

[7] Interestingly, predicting all 0s or all 1s will always be optimal given prior ignorance about the dataset, this is called the Bayes Optimal Predictor

had a click rate twice that of the other two sites for example), this meant that the advantages of utilising transferable knowledge between the models was lesser than the negative knowledge transfer achieved by the different in model types.

Bibliography

[1] BOUGHORBEL, S., JARRAY, F., AND EL-ANBARI, M. Optimal classifier for imbalanced data using matthews correlation coefficient metric. *PLOS ONE 12*, 6 (06 2017), 1–17.

[2] BRANCO, P. O., TORGO, L., AND RIBEIRO, R. P. Smogn: a pre-processing approach for imbalanced regression.

[3] CHAWLA, N. V., BOWYER, K. W., HALL, L. O., AND KEGELMEYER, W. P. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research 16* (2002), 321–357.

[4] ROY, M.-H., AND LAROCQUE, D. Robustness of random forests for regression. *Journal of Nonparametric Statistics 24*, 4 (2012), 993–1006.