

Machine Learning Engineer Nanodegree

Capstone Project

Keyun Shang

March 31st, 2018

TalkingData AdTracking Fraud Detection Challenge (from Kaggle¹ competition)

Project Overview

For companies that advertise online, an overwhelming volume of click fraud can lead to misleading click data and wasted money. Although companies keep trying to find the behaviour pattern of click fraud, the huge amount data makes it almost impossible to be analyzed manually. In recent years, the emerge of high computation power and machine learning has given us the ability to use computer for predicting the user behavior.^{2 3}

TalkingData is china's biggest independent data service platform and had suffered from huge volumes of fraudulent traffic. Through building an IP blacklist and device blacklist, they have successfully prevented click fraud for app developers. However, now they want to step ahead of fraudsters and hope to find an user behaviour pattern that predicts if a user will download an app after clicking a mobile app advertisement, which is the aim of the capstone project.

Problem Statement

Their problem is a binary classification problem. The input are a set of features including ip address, app, device, os, channel, click time etc., the goal is to predict whether a user download an app after clicking a mobile app advertisement.

Metrics

As this is a Kaggle competition project, the best way for evaluation would be the Kaggle score for the test set. However, it is not appropriate in this study due to the following reasons:

¹ <https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection>

² <https://pdfs.semanticscholar.org/1518/6bd13f3f5a33598be9930d2e093055bb7c93.pdf> (A Novel Ensemble Learning-based Approach for Click Fraud Detection in Mobile Advertising)

³ http://ink.library.smu.edu.sg/cgi/viewcontent.cgi?article=2989&context=sis_research (Detecting Click Fraud in Online Advertising: A Data Mining Approach)

- 1) The number of submissions per day is limited by Kaggle.
- 2) If the Kaggle project expires before we complete the study, we can no longer obtain Kaggle score.

For academic purpose, we will use F-beta score as the metric instead.

$$F_{\beta} = (1 + \beta^2) \cdot ((precision \cdot recall) \div (\beta^2 \cdot precision + recall))$$

II. Analysis

Data Exploration

Input data fields

Each row of the training data contains a click record, with the following features.

- ip: ip address of click.
- app: app id for marketing.
- device: device type id of user mobile phone (e.g., iphone 6 plus, iphone 7, huawei mate 7, etc.)
- os: os version id of user mobile phone
- channel: channel id of mobile ad publisher
- click_time: timestamp of click (UTC)
- attributed_time: if user download the app for after clicking an ad, this is the time of the app download
- is_attributed: the target that is to be predicted, indicating the app was downloaded

Note that ip, app, device, os channel are encoded values.

Exploratory Visualization

The plots below show the number of unique values per feature as well as how the target values are distributed. The data are 100,000 randomly-selected rows of training data.

Fig.1 A plot showing the number of unique values for features like ip,app, device, os, channel. Note that the number of unique values of ip is very high.

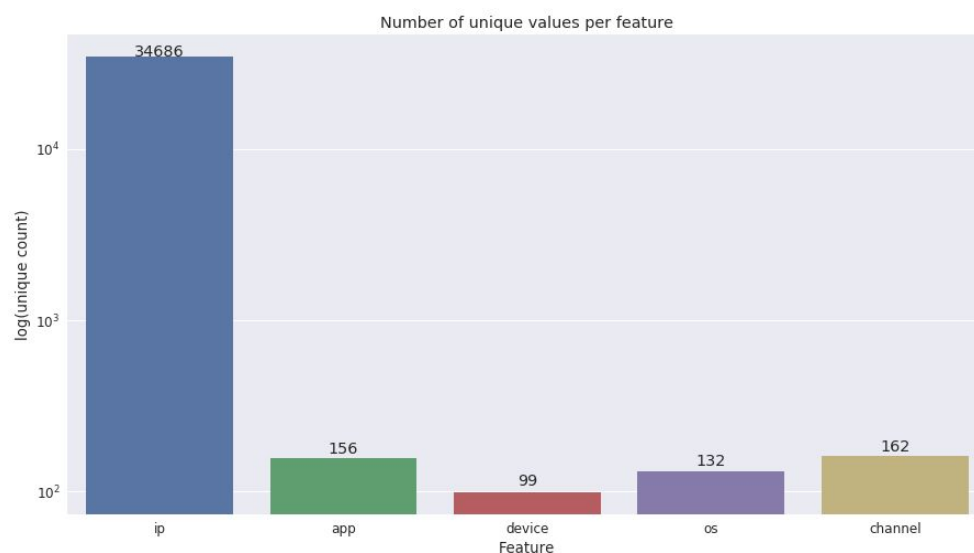


Fig.2 A plot showing the distribution of the probability of the target value. Note that this distribution presents a large class imbalance, as the probability of positive target values is only 0.25%. We need to deal with the imbalance in data processing.



Algorithms and Techniques

We will use the following common algorithms as below to train the model for comparison.

- Decision tree
- Naive Bayes
- Logistic Regression

All of the three algorithms have their advantages and disadvantages. For instance, Decision tree is simple to understand and interpret, but Decision-tree learners can create over-complex trees that lead to overfitting.⁴ Naive bayes learner is computationally fast and can work well with high dimensions, but it relies on the strong independent assumption between features.⁵ Logistic regression is easy to implement and very efficient in training, but it is basically a generalized linear model and therefore it doesn't work well with non-linear problem.⁶

Through an initial evaluation, we will pick up one best model and tune it with grid search method.

Benchmark

As there is no available benchmark model for this project at present, we take use the naive predictor as the benchmark model. Naive predictor is a simplest predictor that predicts the data with current values. Any new model should be better than or at least the same with it.

III. Methodology

Data Preprocessing

Target value data upsampling

As data exploration shows, there exists a large class imbalance in the training data. We will improve this situation by adding the positive target value data. The probability of positive target value in the dataset is about 0.25% , we increase it to 20% by upsampling the positive target data.

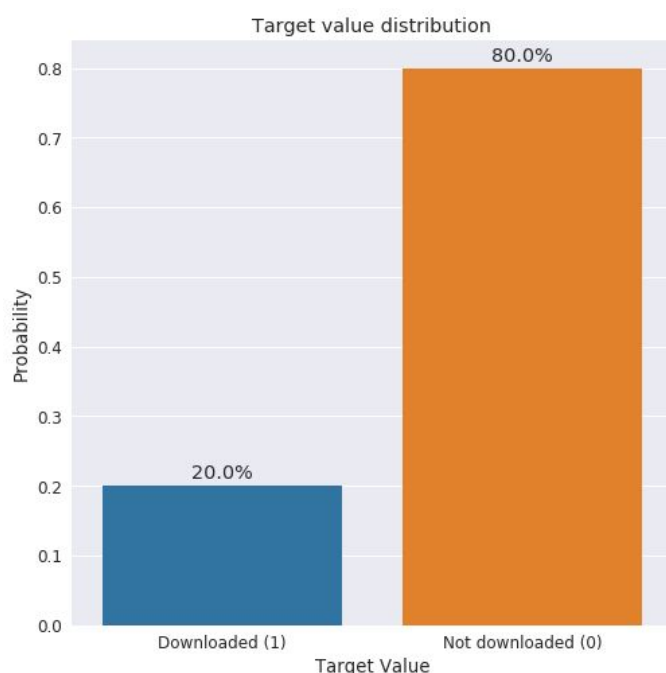
Fig.3 A plot showing the distribution of the probability of the target value after data upsampling.

⁴ <http://scikit-learn.org/stable/modules/tree.html>

⁵ http://scikit-learn.org/stable/modules/naive_bayes.html

⁶

<https://www.quora.com/What-are-the-pros-and-cons-of-using-logistic-regression-with-one-binary-outcome-and-several-binary-predictors>



One-hot Encoding

The features like app, device, os channel, can be seen as categorical data. Although they are numerical data, allowing the model to assume a nature ordering between categories could lead to a poor performance. Therefore, we applied one-hot encoding to these categorical data. Finally, we got a total of 6,574 features.

The ip address can also be seen as categorical data and theoretically we should include it into one-hot encoding. However, as the data exploration shows, ip address has a large amount of unique values even just in sample of training data, one-hot encoding to this feature could lead to a significant performance issue. Therefore, we removed ip address from our feature space in data processing.

Shuffle and split data

Although TalkingData has provided a dataset covering approximately 200 million clicks over 4 days⁷, training with all of the data could be relatively time and memory consuming. Therefore, I decided to take use of a part of these data only. In fact, the amount of the data I used for training is as following.

- Total number of records: 25,000
- Records with positive label value: 5,000
- Records with negative label value: 20,000

⁷ <https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection/data>

Furthermore, I split the training data with into the training/testing sets a ratio of 80:20. For model tuning, I split the training set again into training/validation set with a ratio of 80:20.

Implementation

The implement process can be divided into following steps :

1. Create learners for decision tree, GaussianNB, logistic regression.
2. For each learner,
 - a. fit the learner to the train set
 - b. Get the predictions on the first 5,000 data on the training subset
 - c. Get the predictions on the testing subset
 - d. Compute the accuracy and F-beta score of the first 5,000 data on the training subset
 - e. Compute the accuracy and F-beta score on the test set
3. Plot graphs to compare the training time, prediction time, accuracy score and F-beta score between different learners.

Refinement

Based on the result of the initial evaluation, I chose the logistic regression as the best model. Then, I optimized the model with grid search method.

For models with regularization, like logistic regression, the best hyper parameters to grid search is "C". Here I used a logspace , `np.logspace(-3,3,7)` , for this parameter tuning.

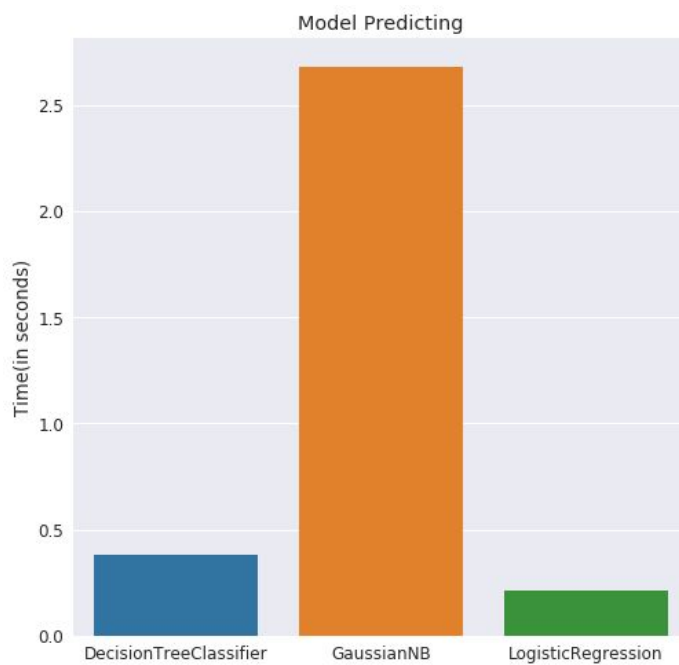
IV.Results

Model Evaluation and Validation

The evaluation to the three models are as the graphs below. The metrics for evaluation include model training time, model predicting time, accuray score and F score on training and sub testing subset.

**Fig.4**

A graph showing the training time of three models

**Fig.5** A graph showing the predicting time of three models

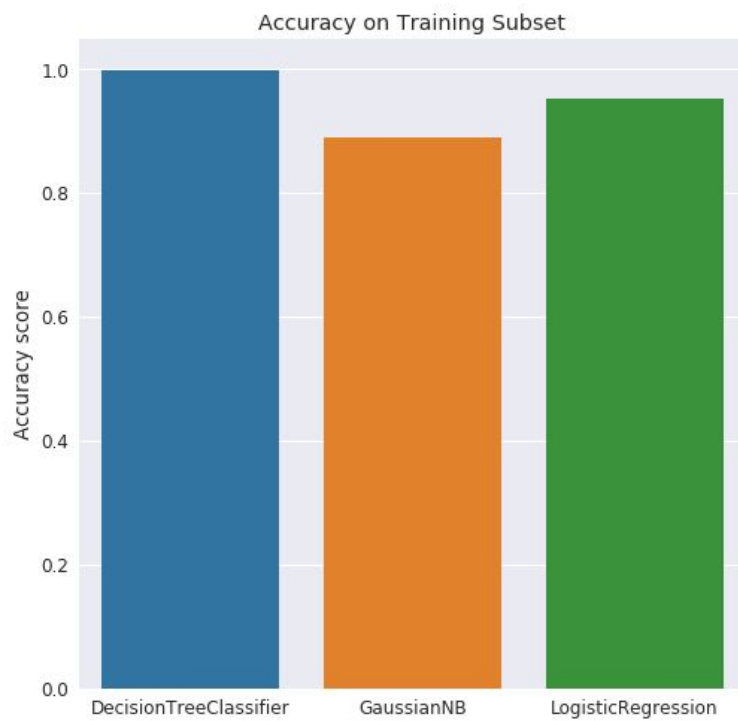


Fig.6 A graph showing the accuracy on training subset



Fig.7 A graph showing the accuracy on testing subset



Fig.8 A graph showing F-beta score on training subset



Fig.8 A graph showing F-beta score on testing subset

Further with grid search method, I run an optimization to the best model. The accuracy and F-beta score of the best model is as following.

	Unoptimized model	Optimized model
Accuracy score	0.9560	0.9560
F-beta score	0.9223	0.9223

Justification

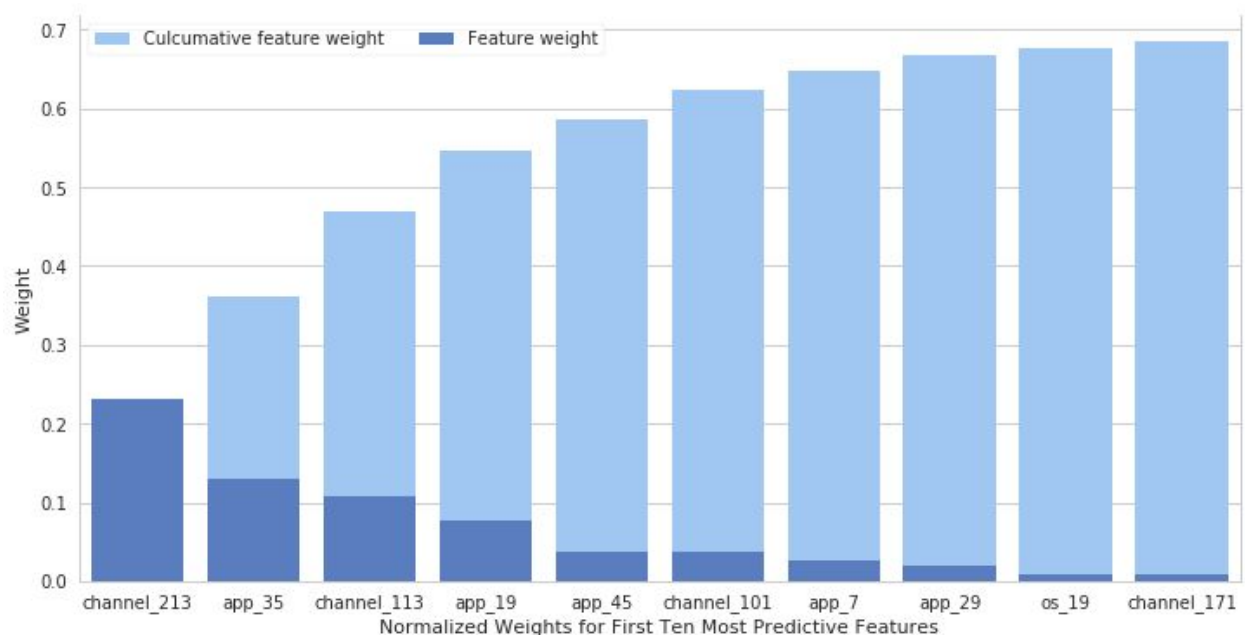
Based on the evaluation of the training time , predicting time, accuray score and F-beta score, the logistic regression can be considered as the best of the three. The reason is,

- Both training and prediction time is the least.
- The accuracy and F-beta score is the highest.

V.Conclusion

Free-Form Visualization

Fig.9 Normalized weights for first ten most predictive features.



As **Figure 9** shows, we can see that the cumulative feature weight of the first ten most predictive features is near to 0.7. In other words, the top ten important features contribute more than half of the importance of all features present in the data. This suggests that we can reduce the feature space and simplify the information required for model learning, if the training time and predicting time are concerns. The table below shows a comparison of accuracy and F-beta score between the model of full features and the model with top ten important features.

	Full-feature model	Reduced-feature model
Accuracy score	0.9560	0.9460
F-beta score	0.9223	0.9022

Reflection

Eventually I successfully built a model with a F-beta score over 0.90. Meanwhile, I also analyzed the top ten most important features which contribute more than half of the importance of all features. The importance analysis could help company better understand user behaviour.


Additionally, the process used for this project can be summarized as the following steps:

1. The data are explored and observed and pre-processed
2. A benchmark was created
3. Based on the pre-processed data, three models are trained and evaluated with specified metrics
4. The best model was selected and optimized

I found the step 1 is the most difficult. Since the large amount of categorical features could lead to a large memory consuming in data processing, I had to drop some features like ip address, and implement a customized one-hot encoding method instead of using third-party library to ensure that data processing wouldn't run out of the memory.

Improvement

Although TalkingData provides with a huge data for training, I didn't use all of the data due to the hardware limitation. With better hardware such as larger memory and faster CPU/GPU, we can include much more data into the training process, which may produce better performance than now.



Another thing we can do is to observe and try to understand more about some of the features. For instance, in this project, we dropped ip address because it had a large amount of unique values. However, we could analyze the count of user clicks by ip and include it into feature space.

For academic purpose, in this project, I evaluated decision tree, naive bayes and logistic regression. Actually, there are other algorithms like random forest, that work quite well in real-world situations and it also deserves a try.