TalkingData AdTracking Fraud Detection Challenge

Sun Hwa Ryu

Contents Table

0. Overview

Description

TalkingData, China's largest independent big data service platform, covers over 70% of active mobile device s nationwide. They handle 3 billion clicks per day, of which 90% are potentially fraudulent. The goal of the competition is to create an algorithm that predicts whether a user will download an app after clicking a mobile app ad.

Evalution

Submissions are evaluated on area under the ROC curve between the predicted probability and the observed target.

0. overview

variables

- ip : ip address of click
- app : app id for marketing
- device : device type id of user mobile phone
- 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 download

1. Data Exploration

Explore 100,000 data

- ip : 100000 non-null int64
- app : 100000 non-null int64
- device : 100000 non-null int64
- os : 100000 non-null int64
- channel : 100000 non-null int64
- click_time : 100000 non-null datetime64
- attributed_time : 227 non-null object
- is_attributed : 100000 non-null int64

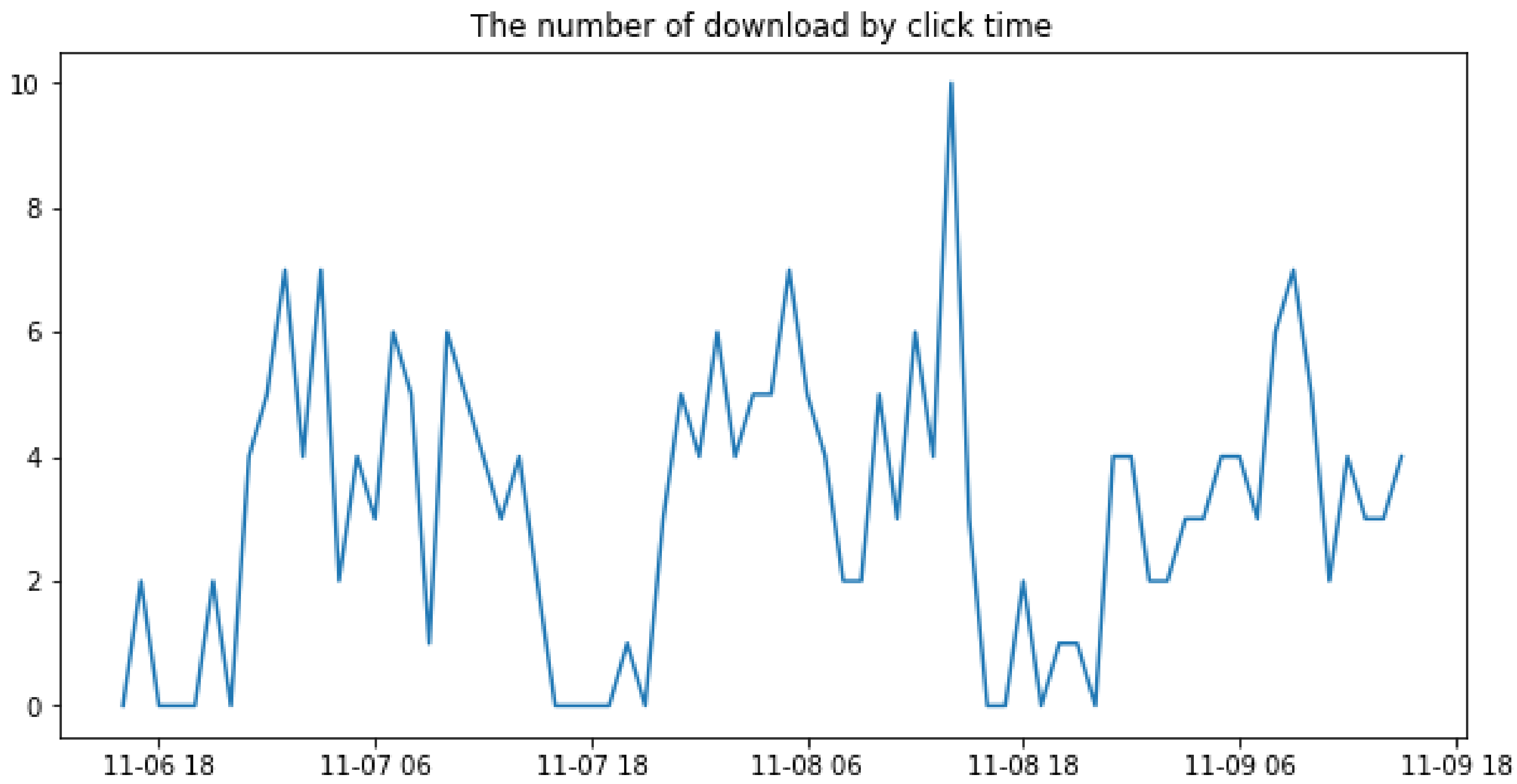
Check download frequency

- 0 : 99773
- 1 : 227

download proportion : 0.00227

1. Data Exploration

Check the number of download by click time



2. Data Preprocessing

Train all data

preprocessing

Train sample data

preprocessing

Make derived variables

Create derived variables in each train all dataset and train sample dataset.

A total of 14 derived variables are created.

- hour : hour from click time

2. Data Preprocessing

Train all data

preprocessing

Train sample data

preprocessing

Make derived variables

: download proportion

- ip_attr_prop : # by ip
- app_attr_prop : # by app
- device_attr_prop : # by device
- os_attr_prop : # by os
- channel_attr_prop : # by channel
- hour_attr_prop : # by hour
- tot_attr_ptop : the sum of the above 6 variables

2. Data Preprocessing

Train all data

preprocessing

Train sample data

preprocessing

Make derived variables

: download proportion

- ip_hour_prop : # by ip and hour
- ip_app_prop : # by ip and app
- ip_channel_prop : # by ip and channel
- hour_app_prop : # by hour and app
- hour_channel_prop : # by hour and channel
- tot_vv_prop : the sum of the above 5 variables

2. Data Preprocessing

Train all data

preprocessing

Train sample data

preprocessing

Check correlation

ip_attr_prop	: 0.438892
app_attr_prop	: 0.444209
device_attr_prop	: 0.201987
os_attr_prop	: 0.226293
channel_attr_prop	: 0.389942
hour_attr_prop	: 0.008851
tot_attr_ptop	: 0.532482

2. Data Preprocessing

Train all data

preprocessing

Train sample data

preprocessing

Check correlation

ip_hour_prop	: 0.582208
ip_app_prop	: 0.755585
ip_channel_prop	: 0.715354
hour_app_prop	: 0.457047
hour_channel_prop	: 0.416602
tot_vv_prop	: 0.739013

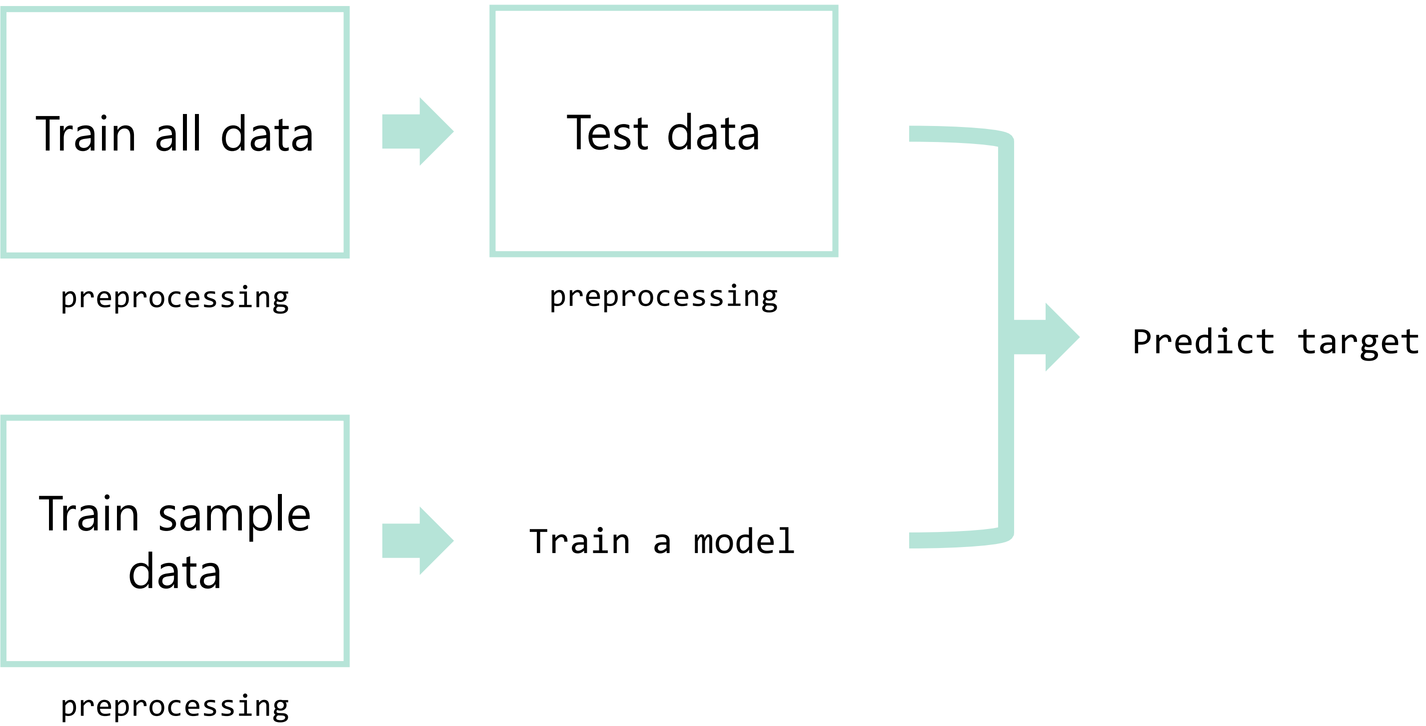
2. Data Preprocessing

Preprocess test data

Based on train all dataset except 'hour' variable, 13 derived variables are created in the test dataset.

Because train all dataset is the most data, the value of the test dataset can be filled without as many blanks as possible, thus creating derived variables in the test dataset using train all dataset.

3. Target Variable Prediction



3. Target Variable Prediction

Create features to use a model

- feat1 = ip_attr_prop, app_attr_prop, device_attr_prop, os_attr_prop, channel_attr_prop, hour_attr_prop, tot_attr_prop
- feat2 = ip_hour_prop, ip_app_prop, ip_channel_prop, hour_app_prop, hour_channel_prop, tot_vv_prop
- feat3 = feat1 + feat2
- feat4 = ip_attr_prop, app_attr_prop, channel_attr_prop, tot_attr_prop
- feat5 = feat4 + feat2
- feat6 = app_attr_prop, channel_attr_prop, hour_app_prop, hour_channel_prop

3. Target Variable Prediction

Predict target variable

- Linear Regression
- Ridge
- Logistic Regression
- Decision Tree
- Random Forest
- Gradient Boosting
- K-Nearest Neighbors
- Support Vector machines
- LightGBM

Skip because it takes too long

3. Target Variable Prediction

Predict target variable

❖ Linear Regression

samples / features	10m	20m	30m
feat1	0.9336475	0.3937085	0.9396936
feat2	0.7903207	0.7990348	0.8090254
feat3	0.6832881	0.6891693	0.6870306
feat4	0.9394377	0.9393066	0.9394337
feat5	0.6786381	0.6730954	0.6829231
feat6	0.9467690	0.9468087	0.9466697

3. Target Variable Prediction

Predict target variable

❖ Logistic Regression

C \ samples	10m	20m	30m
0.01	0.9518560	0.9518226	0.9518260
0.1	0.9517896	0.9518113	0.9517822
1	0.9517904	0.9517846	0.9517540
10	0.9517882	0.9517830	0.9517553

✓ feature : feat6

3. Target Variable Prediction

Predict target variable

❖ Decision Tree

max_depth \ samples	10m	20m	30m
3	0.9039194	0.9039806	0.9040380
4	0.9068583	0.9065484	
5	0.9379549	0.9245333	0.9310434

✓ feature : feat6

3. Target Variable Prediction

Predict target variable

❖ Random Forest

max_depth \ n_estimators	30	50	70
3	0.9117286	0.9325352	0.9325768
4	0.9446114	0.9444698	0.9481182
5	0.9511519	0.9506940	0.9506489

- ✓ feature : feat6
- ✓ sample : 10m
- ✓ max_features : 1

3. Target Variable Prediction

Predict target variable

- Gradient Boosting

max_depth \ n_estimators	30	50
3	0.9058254	0.9069254
4	0.9426463	0.9432340
5	0.9477711	0.9486383

- feature : feat6
- sample : 10m
- learning_rate : 0.01

3. Target Variable Prediction

Predict target variable

- LightGBM

samples / features	10m	20m	30m
feat1	0.9426481	0.9411704	0.9398357
feat2	0.8694790	0.8232350	0.8775217
feat3	0.8694790	0.8467034	0.8577380
feat4	0.9410401	0.9413678	0.9411245
feat5	0.8921562	0.8471011	0.8415991
feat6	0.9514271	0.9528658	0.9526517

4. Conclusion

Result

- Variables related to app and channel were important.
- The best score : 0.9666069

Realization

- It was more important to know which variables to use than which model to use.

Details

- https://github.com/FlowerSuNa/Ad_Tracking_Project/blob/master/README.md

Thank you.