TalkingData AdTracking Fraud Detection Challenge

CONTENTS

01

Competition Overview

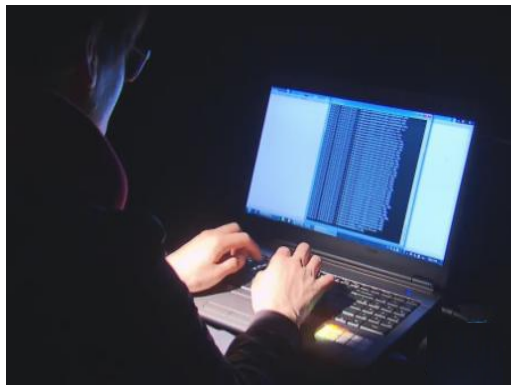
01. Competition Overview

Description

- 온라인 광고 회사는 클릭만으로도 광고 비용이 상승 할 수 있다.
- 사기성 클릭이 전체 클릭의 90% 이상이 잠재적인 사기성 클릭이다.
- 기존에는 클릭 수가 많은 추적하여 IP와 device를 이용한 블랙리스트를 이용한 후발적 조치만 하였다.
- 좀 더 빠르게 사기꾼을 색출하여 데이터 낭비를 방지하고 싶다.

01. Competition Overview

Description



kaggle

(8) "도와줘!"

(7) 사기꾼?

(3) 클릭. 클릭.

TalkingData

(2) "광고 클릭 수 세줘."

(4) "4천만 클릭임."

NAVER

Daum

(3) 클릭. 클릭

(7) 사기꾼?

(6) "똑바로 세!"

(1) "클릭 당 1원 광고 콜?"

(5) "고객님 결제 금액은 4천만원 입니다."

netmarble

01. Competition Overview

Evaluation

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

Submission File

For each `click_id` in the test set, you must predict a probability for the target `is_attributed` variable. The file should contain a header and have the following format:

```
click_id,is_attributed
1,0.003
2,0.001
3,0.000
etc,
```

02

Analysis environment

02. Analysis environment

1. 대용량 데이터 작업환경

train.csv
7GB

test.csv
800MB

대용량 데이터!
개인 노트북으로 분석 불가

Microsoft Azure

Azure 가상머신을 사용해보자!

02. Analysis environment

1. 대용량 데이터 작업환경

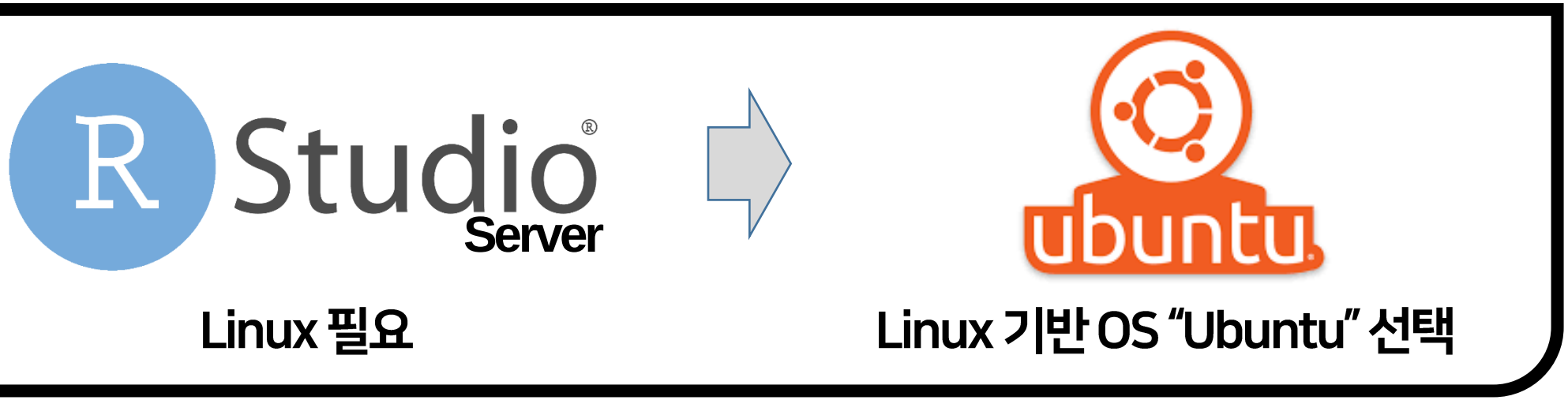
Q) 어떤 가상머신을 구현해야 할까?

- 요구사항

1. 큰 데이터를 처리할 작업공간
2. 작업공간의 공유
3. R언어 사용의 편리성

- Rstudio Server

웹에서 가상머신 IP주소로
간편하게 Rstudio에 접근 가능
Linux환경필요 (Windows 미지원)



02. Analysis environment

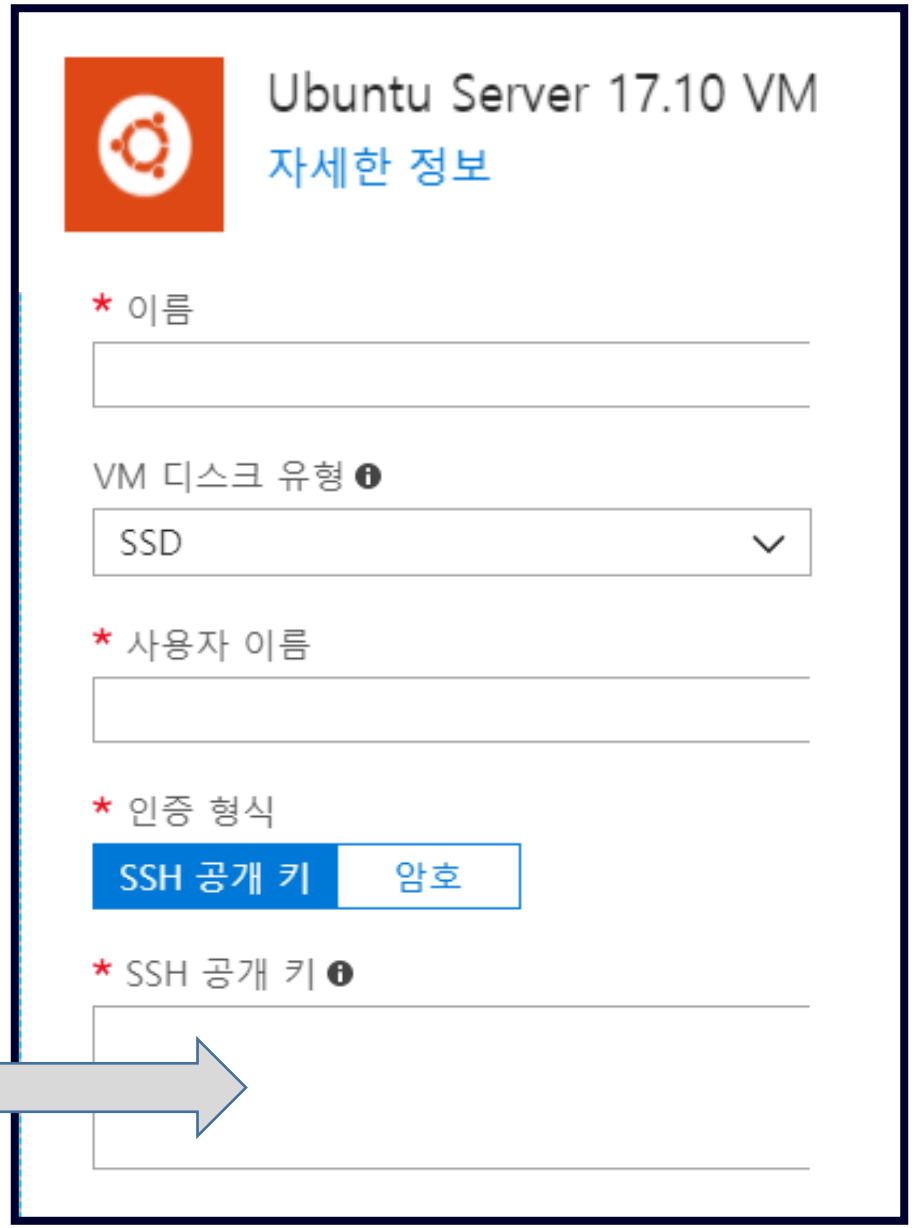
2. Azure로 가상머신 구현하기

@) Azure에서 Ubuntu 구현하기

1. Linux에 접근하기 위해 PuTTY 라는 프로그램을 설치
2. PuTTYgen 실행 후 키 생성
3. Azure에서 Ubuntu 선택 후 SSH 공개 키에 키 입력

…완성!

PuTTygen SSH Key

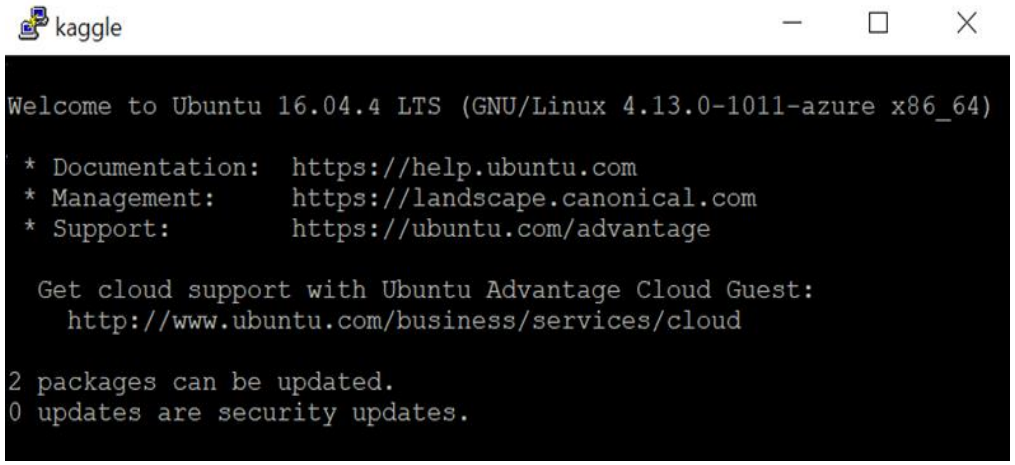


02. Analysis environment

2. Azure로 가상머신 구현하기

@) Ubuntu에 R 환경 구현하기

1. PuTTy를 실행하여 가상머신 IP로 Ubuntu 실행
2. Linux 명령어로 R과 Rstudio Server설치
3. Azure에서 Rstudio Server의 포트 8787 열어주기
4. 인터넷 주소창에 가상머신 IP:8787로 R server 접근 후 Linux ID로 서버 로그인



02. Analysis environment

3. Ubuntu에 R작업환경 구축하기

커널의 여러 기법들을 사용하려 하였으나, 우분투의 기본 지원 R버전이 낮아서 문제가 발생 -> 우분투의 Sources.list 에 R repository 추가해서 R버전 업데이트

You need to add R's repository to your system:

1. Use your favorite text editor (I'm using `gedit` as an example) to open `/etc/apt/sources.list`:

```
sudo -H gedit /etc/apt/sources.list
```

2. Add this line to the file (if this is slow, use another mirror. You may also want to change `precise` into the codename for your Ubuntu version --- e.g., `trusty` for 14.04):

```
deb http://cran.rstudio.com/bin/linux/ubuntu precise/
```

추가적으로, 패키지 설치를 위하여 Git과 Cmake를 설치
-> LightGBM 등의 모델링 패키지 설치가능

```
library(devtools)
options(devtools.install.args = "--no-multiarch") # if yo
install_github("Microsoft/LightGBM", subdir = "R-package"
```

02. Analysis environment

3. Ubuntu에 R작업환경 구축하기

...였지만 끝난 것이 아니었다. 문제발생!

- 문제1) 가상머신이 체험판이라 7G데이터를 분석하기에는 메모리가 부족하다..

- 방안)

1. data.frame이 아닌 data.table의 fread()로 데이터 불러오기.
2. 필요 없는 변수 없애기. (attributed_time)
3. 전체 데이터를 작은 데이터로 줄여서 사용하기.
4. 코드 중간중간에 가비지 컬렉션 gc() 자주 사용해서 메모리 관리하기.

- 문제2) EDA 시각화에 시간이 오래 걸린다 .

커널 EDA를 참고하고 train_sample데이터로 확인해보기

=> 비록 모델링 자체에 1억 8천건의 데이터를 전부 사용하지는 못하고 있지만
데이터를 불러오고 모델링 할 수 있는 환경 구축 성공!

03

EDA

03. EDA

Data fields

- Ip : 클릭한 ip 주소
- App : 마케팅용 앱 id
- Device : 사용자의 핸드폰 기종 id
- Os : 사용자 전화의 os 버전 id
- Channel : 모바일 광고 게시자의 채널 id
- Click_time : 클릭 시각
- Attributed_time : 앱 다운로드 시간
- Is_attributed : 1 :다운로드 / 0 : 사기

03. EDA

Data fields

1억 8천 4백만……

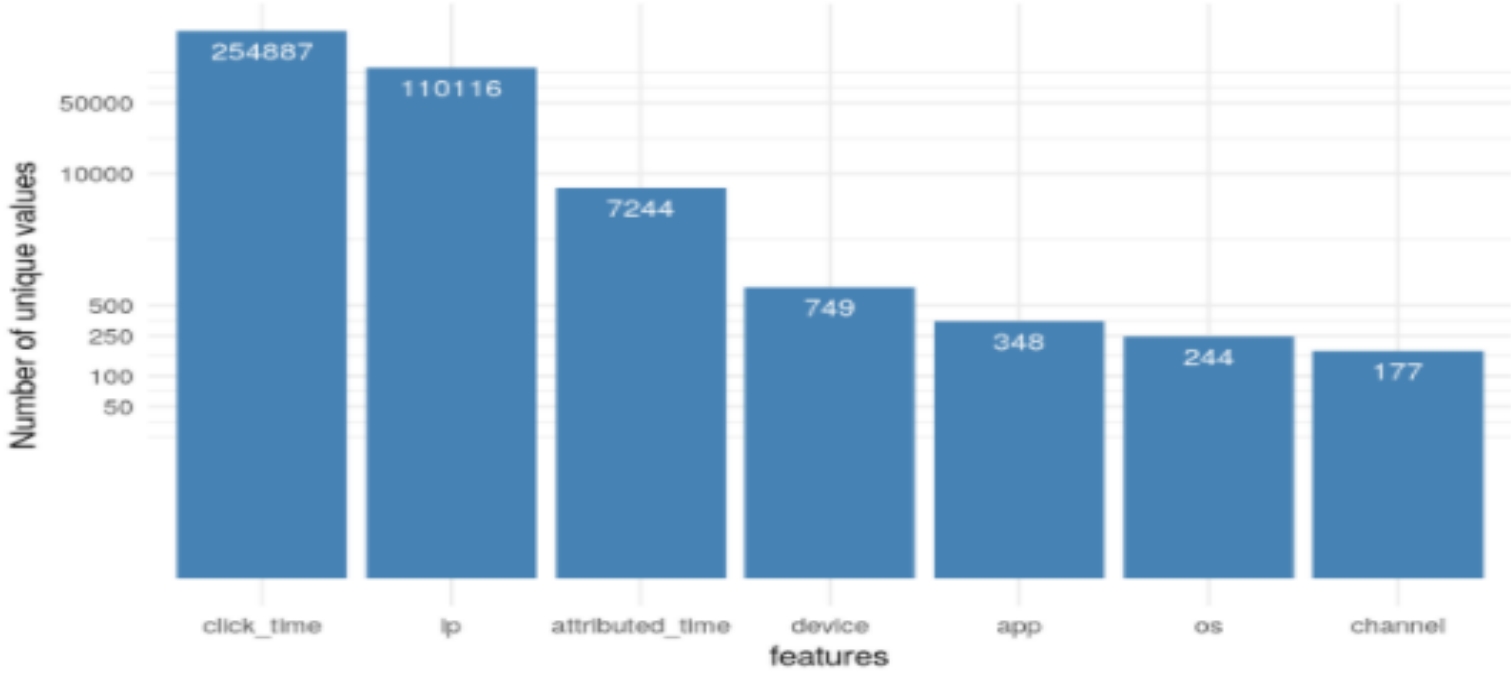
```
Classes ‘data.table’ and 'data.frame':  184903890 obs. of  8 variables:
 $ ip             : int  83230 17357 35810 45745 161007 18787 103022 114221 165970 74544 ...
 $ app            : int  3 3 3 14 3 3 3 3 3 64 ...
 $ device         : int  1 1 1 1 1 1 1 1 1 1 ...
 $ os             : int  13 19 13 13 13 16 23 19 13 22 ...
 $ channel        : int  379 379 379 478 379 379 379 379 379 459 ...
 $ click_time     : chr  "2017-11-06 14:32:21" "2017-11-06 14:33:34" "2017-11-06 14:34:12" "20
17-11-06 14:34:52" ...
 $ attributed_time: chr  "" "" "" "" ...
 $ is_attributed  : int  0 0 0 0 0 0 0 0 0 0 ...
 - attr(*, ".internal.selfref")=<externalptr>
```

- 관측치 수 : 1억 8천4백9십만 3천8백9십
  -> 데이터가 많아서 학습에 오랜 시간이 걸린다. 혹은 불가능하다. (샘플링)

- 변수 class : 비 식별화 되어있는 변수들.
  -> 시간 빼고 모두 범주형 변수라고 보는 것이 적절하다.

03. EDA

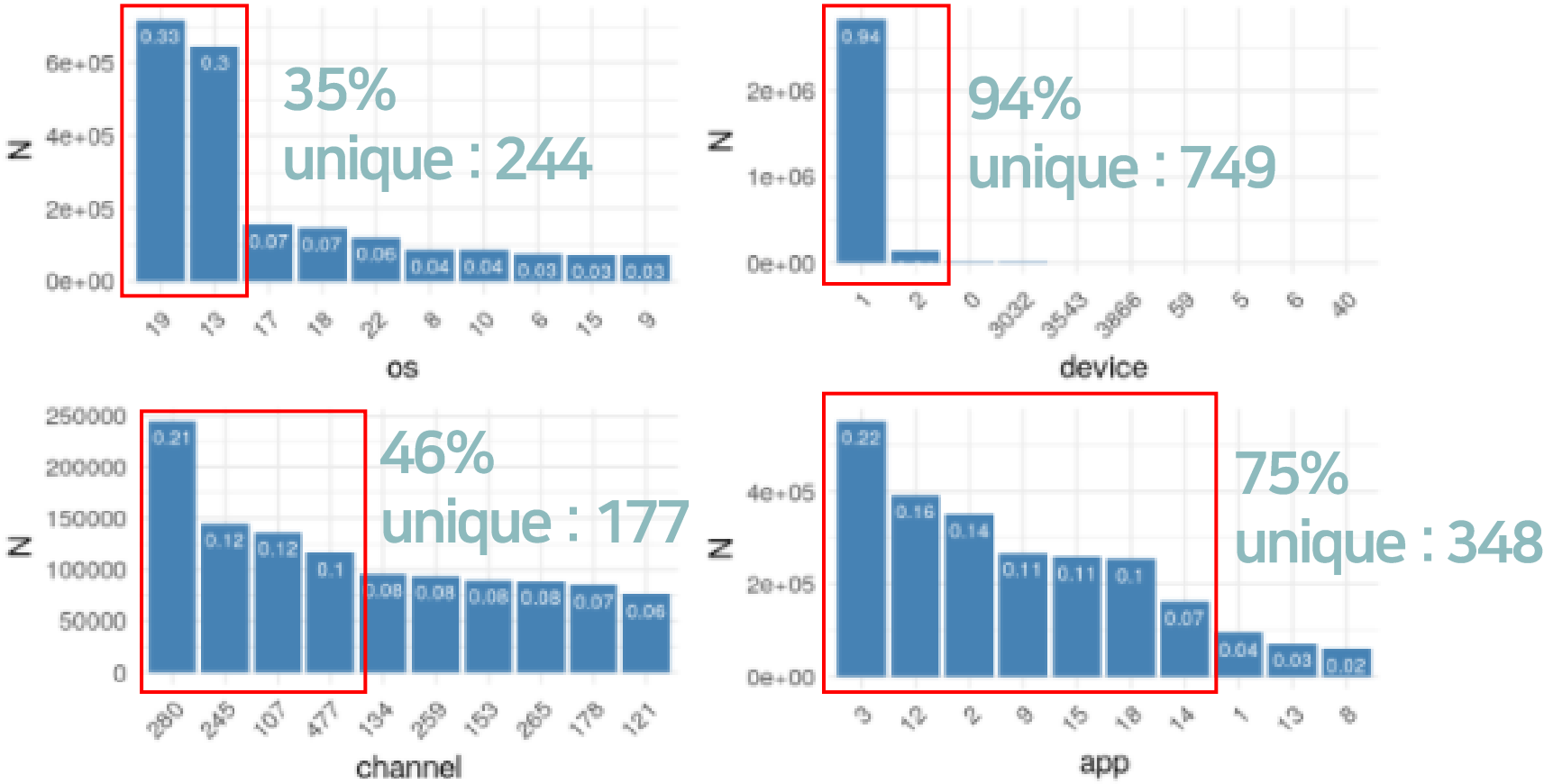
Unique(train)



- 가장 많은 unique 변수를 가지고 있는 ip가 11만개 (샘플 300만개 중에서)
  -> 상당히 많은 중복이 발생하였고, 각 변수의 분포를 알아볼 필요가 있다.

03. EDA

Count feature를 이용한 분포 그래프

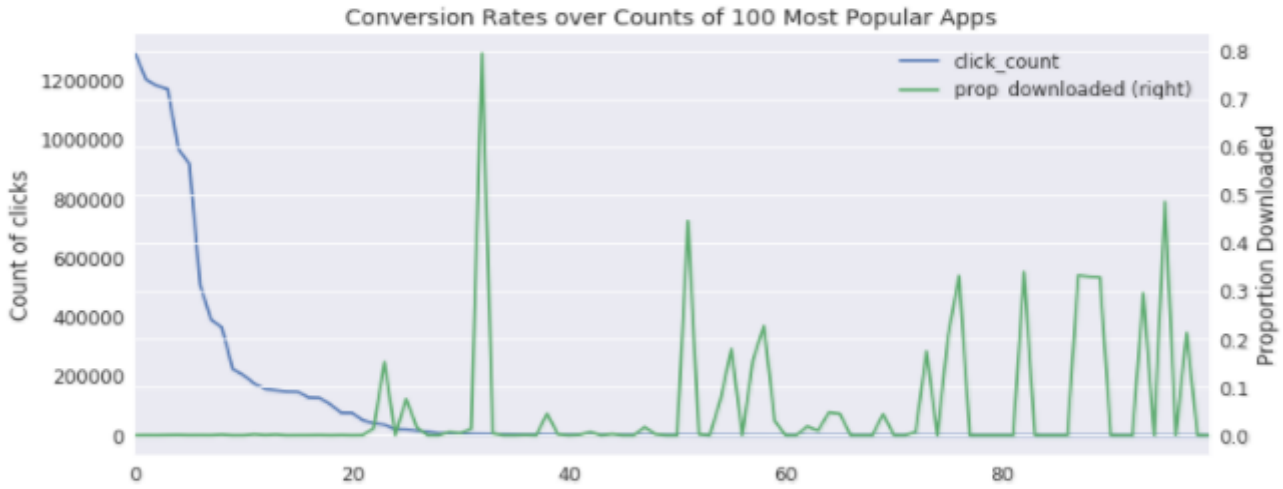


03. EDA

1차 자료와 2차 자료

Conversions by App

Check 100 most popular apps by click count:



2017년 중국 스마트폰 시장 점유율
(단위:%)

스마트폰 제조사	1분기	2분기	3분기	4분기
화웨이	18.9	21	19	19.6
오포	18.7	19.1	18.8	18.4
비보	16.8	14.6	15.2	16.6
샤오미	7.1	12.9	12.8	12.5
애플	7.7	5.2	7.2	13.3
지오니	5.5	5.3	4.9	3.6
메이쥬	4.5	4	3.4	2.4
삼성전자	3.1	2.7	2	1.7
ZTE	3.1	2.7	2	1.5
시아오라지아오	1.9	1.8	1.5	1.1
기타	12.6	10.6	13.2	9.2

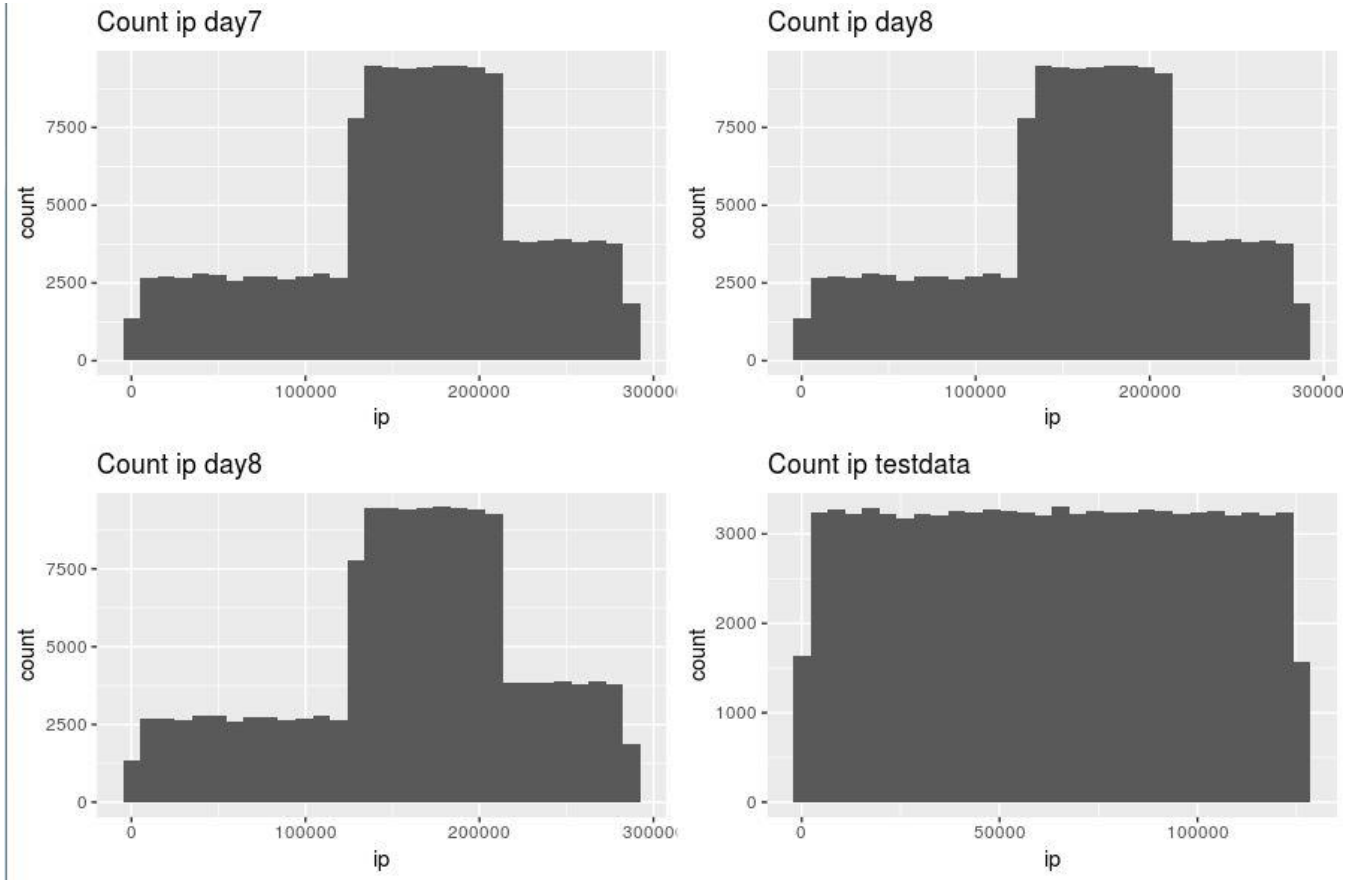
자료:스트래티지 애널리틱스(SA)

App **Device**

- 특정 집단 혹은 개인이 비정상적인 데이터를 만들고 있고, 이들은 일정한 Device를 사용하고 특정 App의 광고 수를 인위적으로 늘리고 있다.

03. EDA

About ip count feature



- ip가 고르게 분포한 형태로 Test data가 변경됨
  -> ip count feature의 사용을 재고해야 한다.

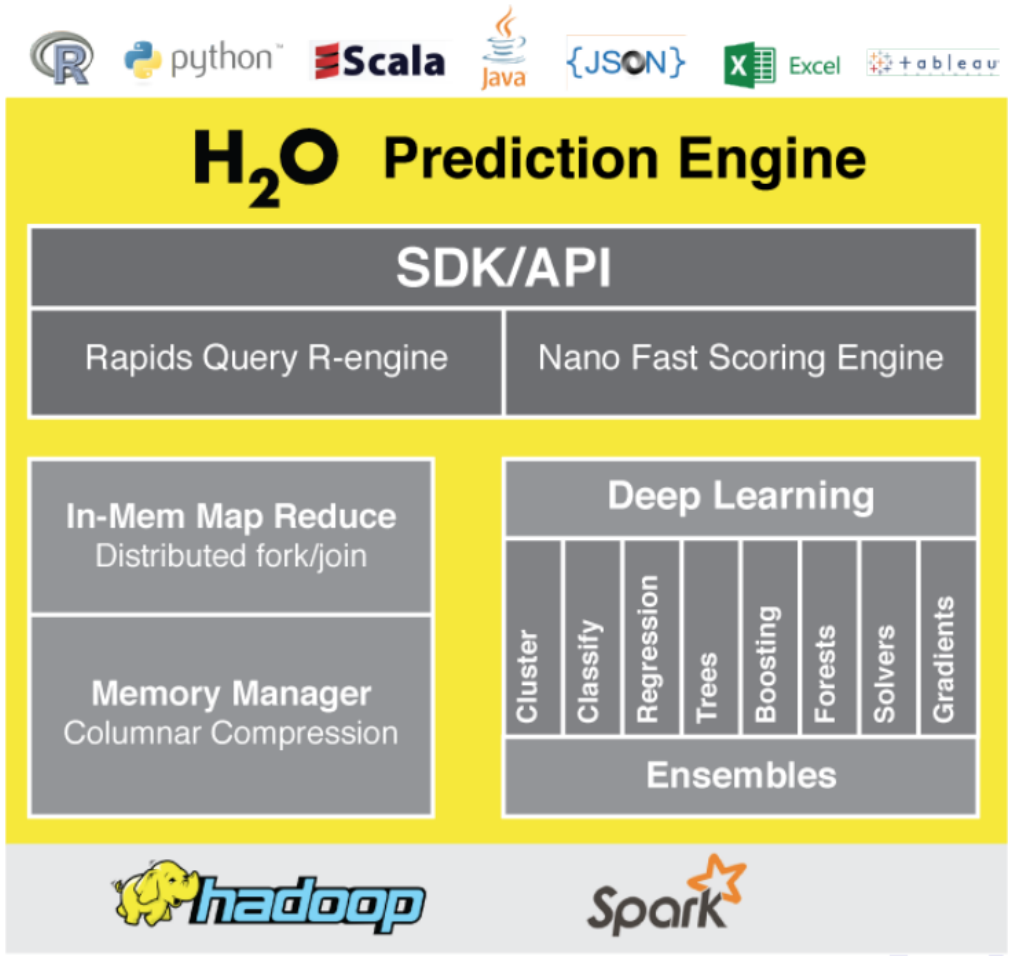
04

H2O Package

04.H2O Package

H2O 패키지 소개

빅데이터 분석을 위한
JAVA기반의
오픈소스
머신러닝/AI 플랫폼

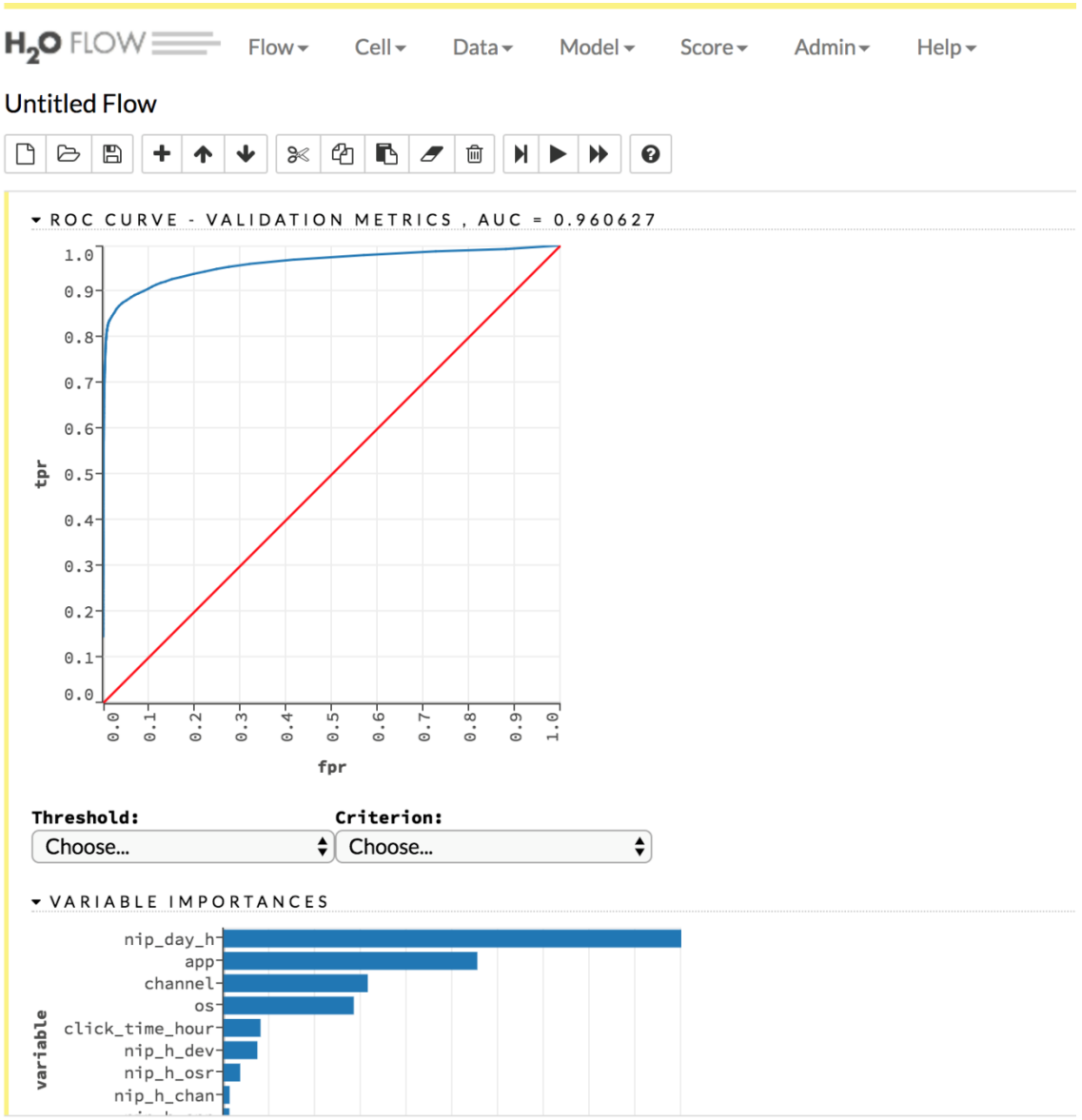


H2O is an open source, in-memory, distributed, fast, and scalable machine learning and predictive analytics platform that allows you to build machine learning models on big data and provides easy productionalization of those models in an enterprise environment. [http://docs.h2o.ai/h2o/latest-stable/h2o-docs/welcome.html]

04.H2O Package

H2O 패키지의 장점

- **빠른속도** (분산 병렬 처리)
- **웹 인터페이스 제공** (H2O Flow)
- **확장성이 좋다.**
- **모델 배포가 편리**
- **사용하기 쉬움**



[H2O Flow]

04.H2O Package

H2O 패키지 사용하기

```
# h2o 클러스터를 R과 연결
h2o.init(nthreads = 4, max_mem_size = "18g")

# h2o 상에 데이터 올리기
training <- as.h2o(training, destination_frame = "training")
validation <- as.h2o(validation, destination_frame = "validation")
test_h2o <- as.h2o(test, destination_frame = "test_h2o")
h2o.ls() # 올라갔는지 확인

target <- "is_attributed"
features <- names(training)[!names(training) %in% target]
```

h2o.init() : h2o 클러스터와 R 연결

(thread 갯수, 최대사용메모리 지정)

as.h2o() : 데이터를 h2o 상에 올림

(또는, h2o.importFile() 로 파일을 바로 올릴 수 있음)

key <fctr>
depth_grid_model_2
depth_grid_model_3
depth_grid_model_4
depth_grid_model_5
gbm_model
glm_model
modelmetrics_GBM_model_R_1521834389751_18@8903738402016773956_on_training@-45614059431649

04.H2O Package

H2O 패키지 사용하기

h2o.gbm() : Gradient Boosting model 생성

```
# GBM 모델 생성
gbm_model <- h2o.gbm(x = features, y = target, training_frame = training, model_id = "gbm_model")
```

```
  |============================================================| 100%
```

```{r}
summary(gbm_model)
```

```
Model Details:
==============

H2OBinomialModel: gbm
Model Key:  gbm_model
Model Summary:
  number_of_trees number_of_internal_trees model_size_in
1              50                       50
  min_depth max_depth mean_depth min_leaves max_leaves
1         5         5    5.00000         28         32
  mean_leaves
1    31.62000

H2OBinomialMetrics: gbm
** Reported on training data. **

MSE:  0.001736985
```

h2o.predict() 로 예측

```
gbm2 # AUC:  0.9612291
preds <- h2o.predict(gbm2, test_h2o)
head(preds)
summary(preds$predict, exact_quantiles=TRUE)
```

	predict <fctr>	p0 <dbl>	p1 <dbl>
1	0	0.9994166	0.0005833523
2	0	0.9993592	0.0006407654
3	0	0.9996849	0.0003150630
4	0	0.9996712	0.0003288168
5	0	0.9996914	0.0003086371
6	0	0.9994270	0.0005730025

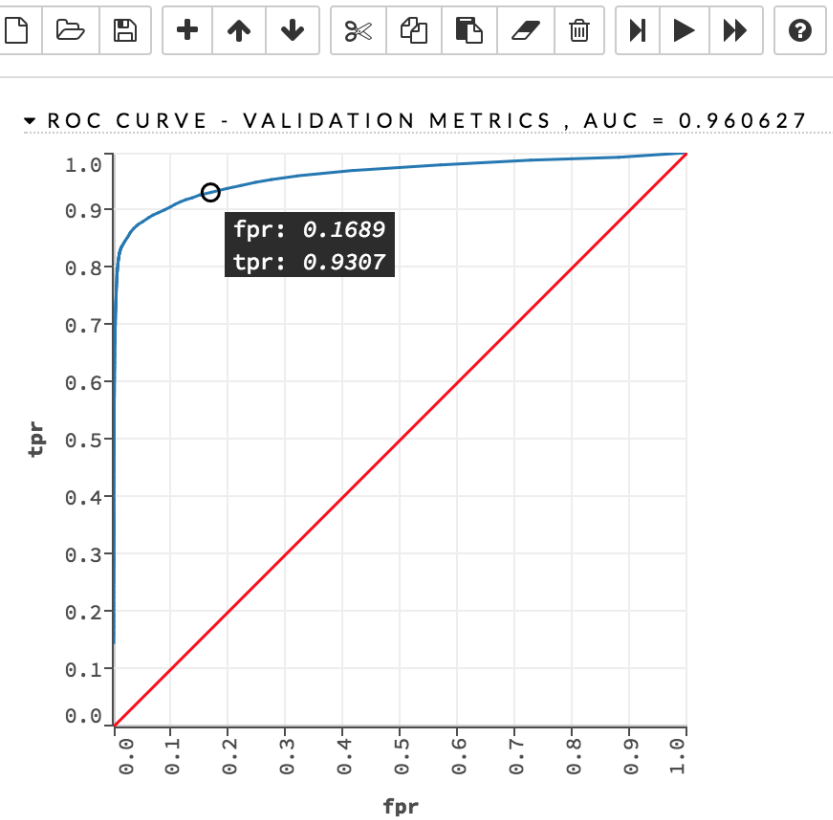
04.H2O Package

H2O 패키지 사용하기

h2o flow 에서 모델정보 확인
(ROC Curve, 변수중요도, 모델파라미터 등)

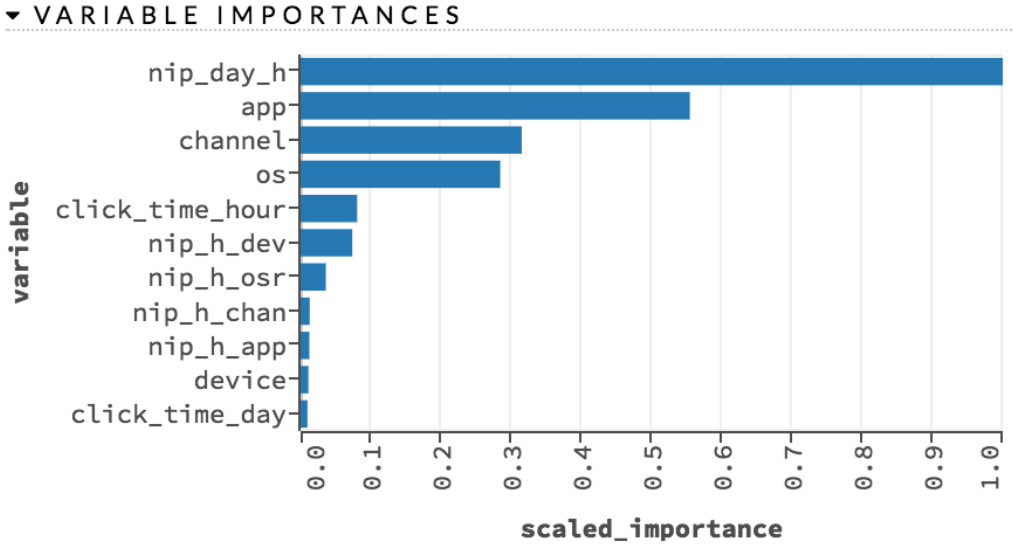
H2O FLOW Flow Cell Data Model Score Admin Help

Untitled Flow



Threshold: Choose...

Criterion: Choose...



MODEL PARAMETERS

Parameter	Value
model_id	depth_grid_model_2
training_frame	training
validation_frame	validation
score_tree_interval	10
response_column	is_attributed
ignored_columns	
ntrees	300
max_depth	8
stopping_rounds	5
stopping_metric	AUC
stopping_tolerance	0.0001
max_runtime_secs	1.7976931348623157e+308
seed	1234
learn_rate	0.05
learn_rate_annealing	0.99
distribution	bernoulli
sample_rate	0.8

Q & A

Thank you