# zzsza / Kaggle_Expedia-hotel-recommendations

github.com/zzsza/Kaggle_Expedia-hotel-recommendations

zzsza

## Kaggle_Expedia-hotel-recommendations

https://www.kaggle.com/c/expedia-hotel-recommendations

## Abstract

Planning your dream vacation, or even a weekend escape, can be an overwhelming affair. With hundreds, even thousands, of hotels to choose from at every destination, it's difficult to know which will suit your personal preferences. Should you go with an old standby with those pillow mints you like, or risk a new hotel with a trendy pool bar?

Expedia wants to take the proverbial rabbit hole out of hotel search by providing personalized hotel recommendations to their users. This is no small task for a site with hundreds of millions of visitors every month!

Currently, Expedia uses search parameters to adjust their hotel recommendations, but there aren't enough customer specific data to personalize them for each user. In this competition, Expedia is challenging Kagglers to contextualize customer data and predict the likelihood a user will stay at 100 different hotel groups.

The data in this competition is a random selection from Expedia and is not representative of the overall statistics.

## Evaluation

Submissions are evaluated according to the Mean Average Precision @ 5 (MAP@5):

where |U| is the number of user events, P(k) is the precision at cutoff k, n is the number of predicted hotel clusters.

## Daily Diary (John Park님 피드백 포함)

### 2017.2.3

- Expedia Hotel Recommendation 시작

- 데이터의 용량 약 3.79g -> pandas에서 읽을시 약 5g
- row : 37,670,293 / Feature : 24
- 2013, 2014년 데이터를 기반으로 2015년의 호텔 추천( y : hotel_cluster / MAP@5 )
- 2013 : 2014 = 1:2의 비율
- 데이터를 볼 때, 시각적으로 그래프를 그리는 것도 좋지만 그냥 데이터를 직접 들여다보는 방법을 추천

```
pd.set_option("max_columns", 40)
```

- 처음엔 데이터를 3~5메가정도로 줄여보고 모델링을 시작하는 것이 좋다
- 위에서 5000~~10000~~100000명 정도로 잘라보기 => 단, 위에서 자르는 방법도 있지만 아래와 같이 랜덤으로 추출을 여러번 해서 검증

```
df1 = df.ix[np.random.choice(df.index, 10000)]
```

- 샘플링을 통해 랜덤포레스트의 important_feature가 변하는지 파악 => 만약 변한다면 데이터가 흔들린다는 증거 => 이런 경우 해결책은?
- is_mobile같이 0, 1인 데이터는 재미가 없다. 복잡하게 나와있는 데이터가 더 의미가 있다는 것
- check_in, out 데이터는 가로로 볼 것 => 요일같은 경우는 평일 / 주말 / 시즌별로 볼 것
- y 값은 보통 제일 좌측 혹은 우측에 두는데, 좌측에 두고 데이터를 움직여보면 더 편하다

## 할 일

- 데이터 랜덤 추출 후, 랜덤포레스트 결과값 달라지는지 파악 (row, column 둘 다 실행)
- h2o 설치 및 공부 ( 머신러닝/딥러닝을 쉽게 돌릴 수 있음 => 한글 자료가 적으니 추후 번역해보면 좋을 것 같다 )
- Feature Engineering 하기 전 Train set 더 완성하기
- 데이터 바라보기

## Question

데이터 분석의 흐름 ( 캐글에 결과 제출하고 모델을 수정하는지 그냥 로그로스같은 지표를 보는지 )

> => Competition이 진행하고 있는 경우엔 1일 제출 횟수가 제한되어 있기 때문에(보통 5회) CV까지 튜닝하고 넣어봄, 그러나 Competition이 끝난 경우엔 횟수 제한이 없기에 그냥 많이 넣어봐도 좋을 것 같음

랜덤포레스트로 나온 값들을 몇개만 신뢰? 혹은 트리를 몇개 할 지

> => 보통 중요도를 지켜보면, 갑자기 스므스해지는 구간이 있음. 그 구간 위에부터 설정..! (보통 5~10개) 그 외의 데이터들은 noise

> 트리는 보통 100개 이상을 사용해야지 15개 30개정도는 너무 약함

h2o.ai와 파이썬으로 분석하는 경우의 차이

> => 모델 튜닝의 차이고 h2o.ai가 최적화가 잘 되어있는 느낌(추후 번역을 해봐도 좋을 컨텐츠)

데이터 불균형이 되었으면 어떻게 해야되는지?

> => 많은 데이터를 넣어보는 것

- is_booking이란 변수를 중요하게 생각했는데 낮게 나왔음. 이런 경우 진짜 중요하지 않은 것일수도 있고, 아닐수도 있지만 전자의 확률이 큼.
- 이 경우 is_booking 변수와 randomforest의 중요도가 높게 나온 변수들만 넣고 모델을 돌려보고 갑자기 중요도가 높아진다면 그 변수는 중요한 것, 아니라면 중요하지 않은 것

## 2017.2.6

- Random Forest에서 뽑은 Feature로 모델 돌려봄 -> 0.14201
- 다른 사람이 뽑은 Feature로 모델 돌려봄 -> 0.15316
- Feature Engineering 살짝 하고난 후 => 0.16293
- 현재 1900명중 하위 30%, 1위의 점수는 0.6정도 => 올라갈 일만 남았다
- 모델을 생성하고 Test set에 넣을 때 보통 13분 걸림 => 트리 수가 10일 경우
- 트리를 증가 시키면 Memory Error.. rate 비율을 조정해도 결국 에러가 나서 딜레마 => 다른 방법을 찾아보기
- Kaggle에서 반드시 모델을 사용해서 fit! 이러진 않음. Cluster 나 다양한 방법으로 실시함 ( 꼭 모델로 해야하는 건 아니다. 편견에 빠지지 말자 )
- Feature 생각 더 해보기

## 2017.2.7

- Data Leak이 존재 => train 데이터에서 test의 답이 있는 경우 => 모델 성능이 조금..
- 이 Competition은 대부분 most_popular를 사용해서 풀었음 ( 많은 사람들이 모델에 비해 안정하다고 판단했나봄 ) => 08. other solution
- most_popular가 baseline => 사람들의 특징을 파악해보기 ( Long tail 부분의 20% 사람들이 왜 특정 호텔을 예약했는지? 등 숫자로 나와 어렵겠지만 분석해보기 )

- 의외로 cnt의 비중이 낮았음. 많은 사람들이 결제하기 전에 많이 유사한 호텔을 보고 결제한다고 생각했지만 데이터는 대부분 1번에, 아니면 많아봤자 2번에 결제를 함
- srch_destination_id / hotel_cluster의 관계 (ex. "강남"을 검색한 후 나오는 호텔 목록 중 클릭한 것이 나옴. srch_destination_id로 묶으면 해당 지역의 호텔들이 나옴 )
- Data Leak을 공략해보기
- XGB 활용해보기 ( Docker 내에 xgboost 설치 완료, ~~Windows 로컬은 무언가 에러가..ㅅ_ㅠ~~ Windows 설치 완료!! Docker는 메모리 문제로 loading이 힘든 상황)

## 2017.2.8

- Leakege data를 활용 -> 0.49658
- Dict 타입에 대한 심층적 이해 ( link ) => json처럼 많은 데이터를 쉽게 처리해 메모리 효율성 증대
- XGBoost 사용 -> 0.30335 => 음.... 랜덤포레스트보단 성능이 좋음
- 많은 변수를 그냥 사용하는 경우 vs 차원축소(PCA)를 하는 경우 => 사용하는 경우도 있음! pca extreme value가 기존 데이터에서 누구를 따라오는지 보고 찾아볼 수 있음!
- 영화 컨택트를 보며 느낀점 : 언어학자/물리학자가 외계인의 언어를 이해하기 위한 과정이 나오는데 데이터 분석 과정이랑 너무 동일..! 영화를 보면서 NLP에 대한 관심이 증가했음. 섹시한 언어학자님 => NLP 의 앞부분(vectorize)은 약간 코딩스러운..! 개발개발스러운 요소고 그 뒷부분이 머신러닝 부분이랑 동일함. 캐글을 통해 문제해결력을 풀어보기..!

## 2017.2.9

- 1위 script는 유저와 검색어 자료를 기반으로 hotel city를 찾아내는 방법
- 나와있는 script가 아닌 나의 모델을 만들어보기
- srch_destination, is_booking, user_location_city, orig_destination_distance를 사용
- srch_destination, is_booking 을 기반으로 1차적인 추천 => 추천 값이 nan가 나온 친구들은 srch_destination, user_location_city, orig_destination_distance 변수로 랜덤포레스트 후 예측 => 0.45197 / 0.46883
- 하 마지막에 확인해보니 내가 새로 문제를 만들고 그 문제에 맞게 풀었던거임..ㅎㅎ 나니..? 후후..ㅎㅎㅎ 내 꾀에 내가 넘어간 꼴.. 이 값이 왜 높은진 이해할 수 없지만..