

# Music Recommendation

KKBox's Music Recommendation Challenge Data in Kaggle



- KKBox's Music Recommendation Challenge Data in Kaggle

Variables	Rows / Cols	Description	Remark
Train	7,377,418 / 6	유저들의 노래 청취에 관한 데이터	
Test	2,556,790 / 6	상동	
Songs	2,296,320 / 7	노래에 대한 데이터	
Song_extra_info	2,296,869 / 3	노래에 대한 메타데이터	
Members	34,403 / 7	유저 데이터	

# 사용자의 과거 청취 데이터를 기반으로 해당 노래 재청취 여부를 예측



Target

(반응변수)

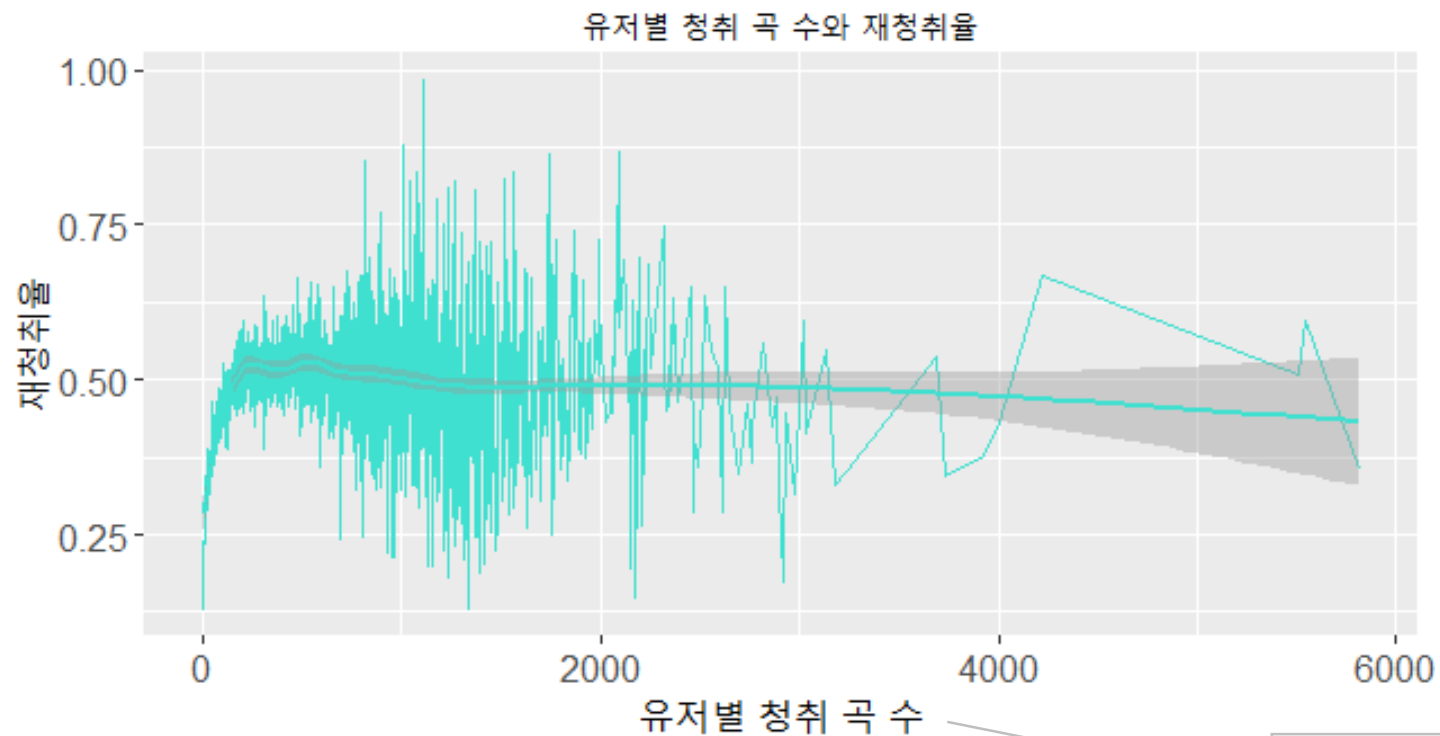
- 1, 사용자가 해당 노래를 처음으로 듣고 한달 안에 재청취한 경우
- 0, 해당 노래를 들긴 했지만, 한달 안에 재청취하지 않은 경우

# Exploratory Data Analysis

File_name (Rows / Cols)	Variable	Format	Description	Unique rows	NaN (NaN / Total)	Examples
Train (7,377,418 / 6)	msno	chr	사용자 ID	30,755	0	"mxGXmmQza/nMXJ/Z9140yG06nxd+Nx4iz2oB/zV6quE=", ...
	song_id	chr	곡 id	359,966	0	"CXoTN1eb7AI+DntdU1vbcwGRV4SCIDxZu+YD8JP8r4E="...
	source_system_tab	chr	노래가 재생된 탭	8	24,849 (0.24%)	"my library", "explore", ...
	source_screen_name	chr	사용자 화면에 나타난 레이아웃	20	414804 (5.6%)	"Explore", "Local playlist more", "Search more"
	source_type	chr	첫 음악 재생 시작점	12	21,539 (0.29%)	"online-playlist", "local-playlist", "album", ...
	target	int	한 달 내 반복 청취 여부	2	0	1, 1, 1, 1, 0, 1, 1, 1, 0, ...

- Train의 결측치들은 모두 범주형 변수이기 때문에 예측해서 결측치를 채우기가 어려움.
- 따라서 해당 행은 모두 “Not Value”로 변환하였음, 총 461,192행(6%)이 변환됨.

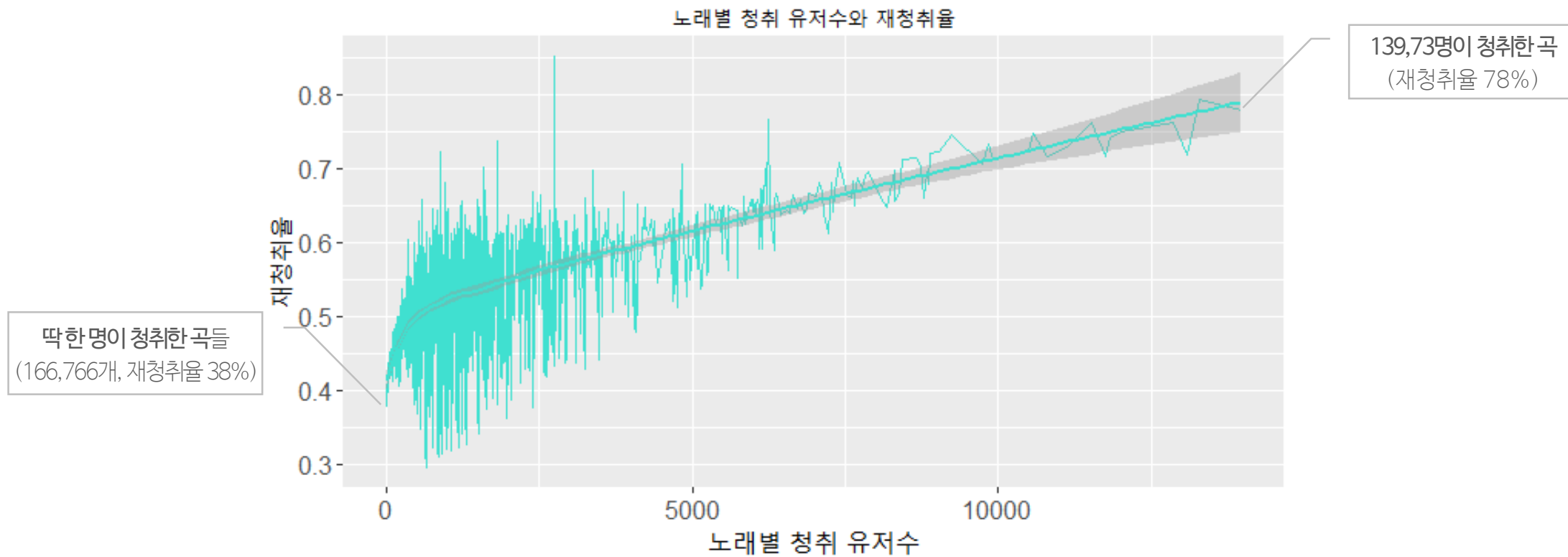
**Q.** 유저들의 청취 곡 수와 재청취율과의 관계는 어떨까?



- 이 그래프를 통해 유추할 수 있는 것 세 가지
  - 노래를 자주 듣는 유저라고 해서 평균 재청취율이 높진 않다.
  - 노래를 적게 듣는 유저들이 있는 구간에서는 노래를 많이 들을 수록 재청취율도 높아진다.
  - 노래를 정말 많이 듣는 유저는 재청취를 하기보다 다양한 노래를 듣는 경우가 조금 더 많은 것 같다.

왜냐하면, 유저 등장빈도가 높은 쪽으로 갈 수록 재청취율은 오히려 조금씩 하락했기 때문.

## Q. 노래별 청취 유저수와 재청취율과의 관계는 어떨까?



- 청취 유저수가 많은 노래일 수록 재청취율도 높았다.
- 즉, 노래별 청취 유저수와 재청취율은 **양의 상관관계**(0.66)가 있다.
- 또한, 이는 **노래의 인기**도를 의미한다고 볼 수 있다.



Account >

Seung Hwan Oh  
rakkoon23@gmail.com

Offline



My Library



Discover



Radio



People



Notification



Setting

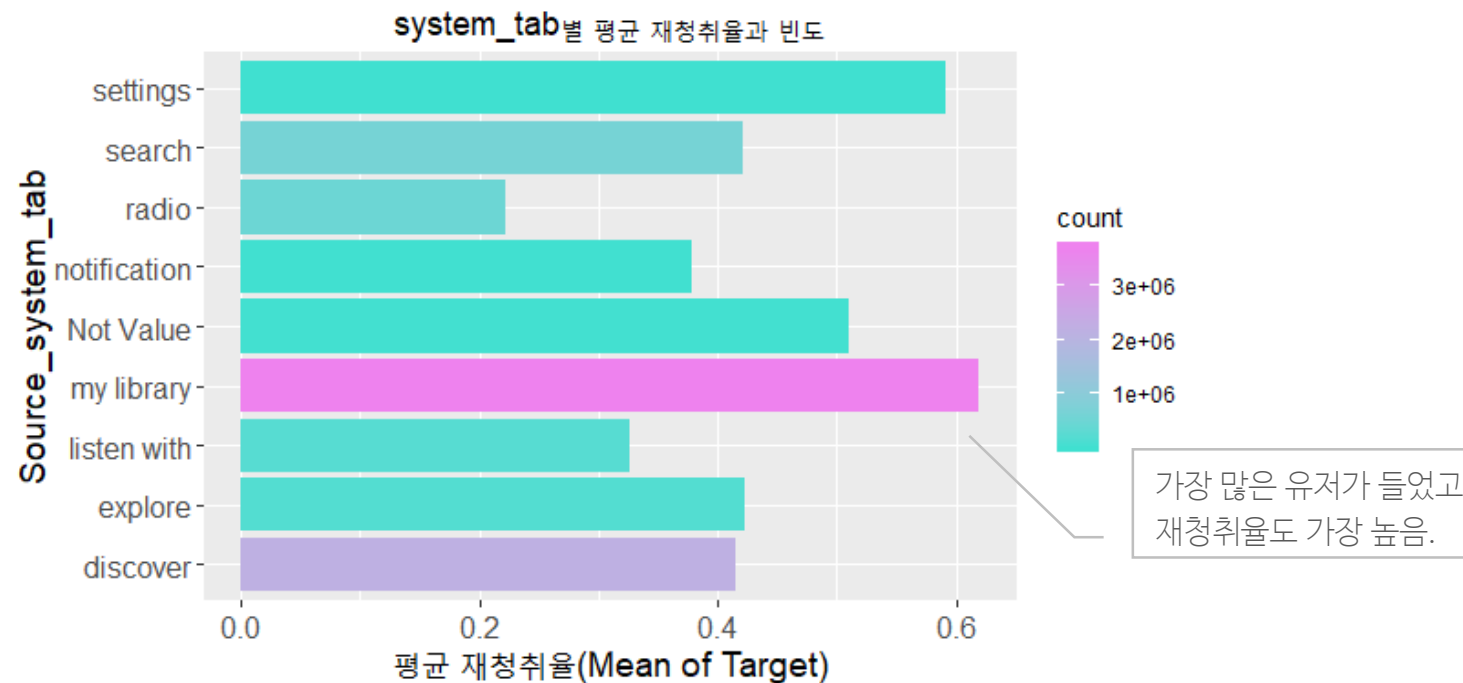


More

Go Premium

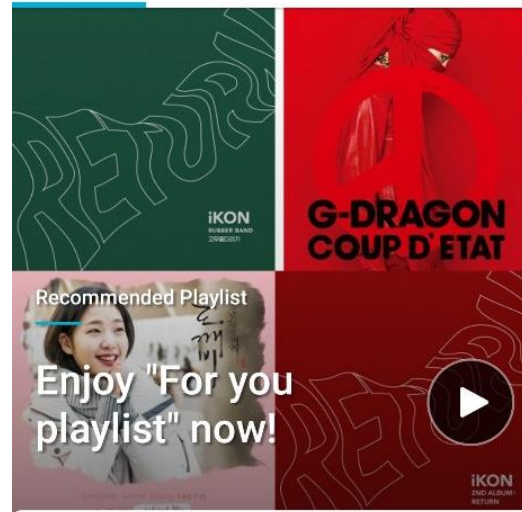
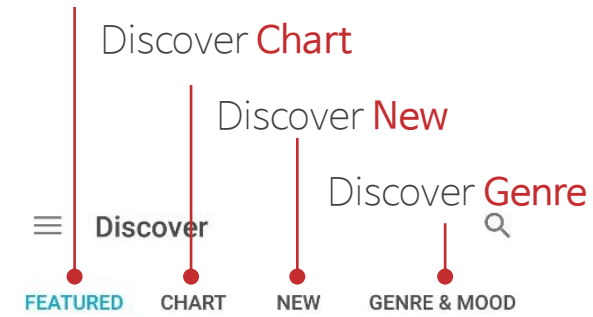


Q. Source\_system\_tab에 따라 재청취율이 어떻게 달라질까?

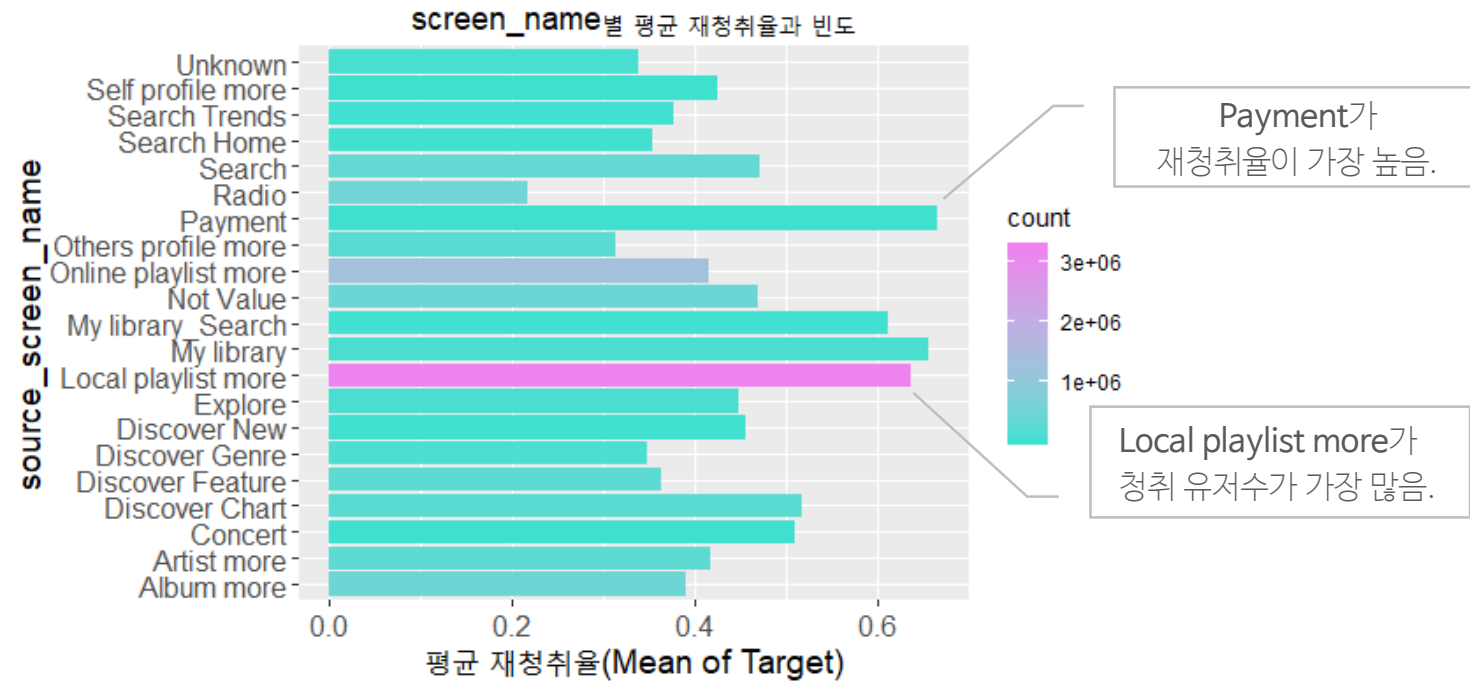


- 막대의 높이가 높을수록 재청취율이 높음.
- 막대의 색이 분홍색에 가까울수록 빈도가 높음.
- 유저들은 system\_tab의 **My\_library**를 통해 노래를 가장 많이 들었고, 재청취율도 가장 높았음.

Discover **Feature**

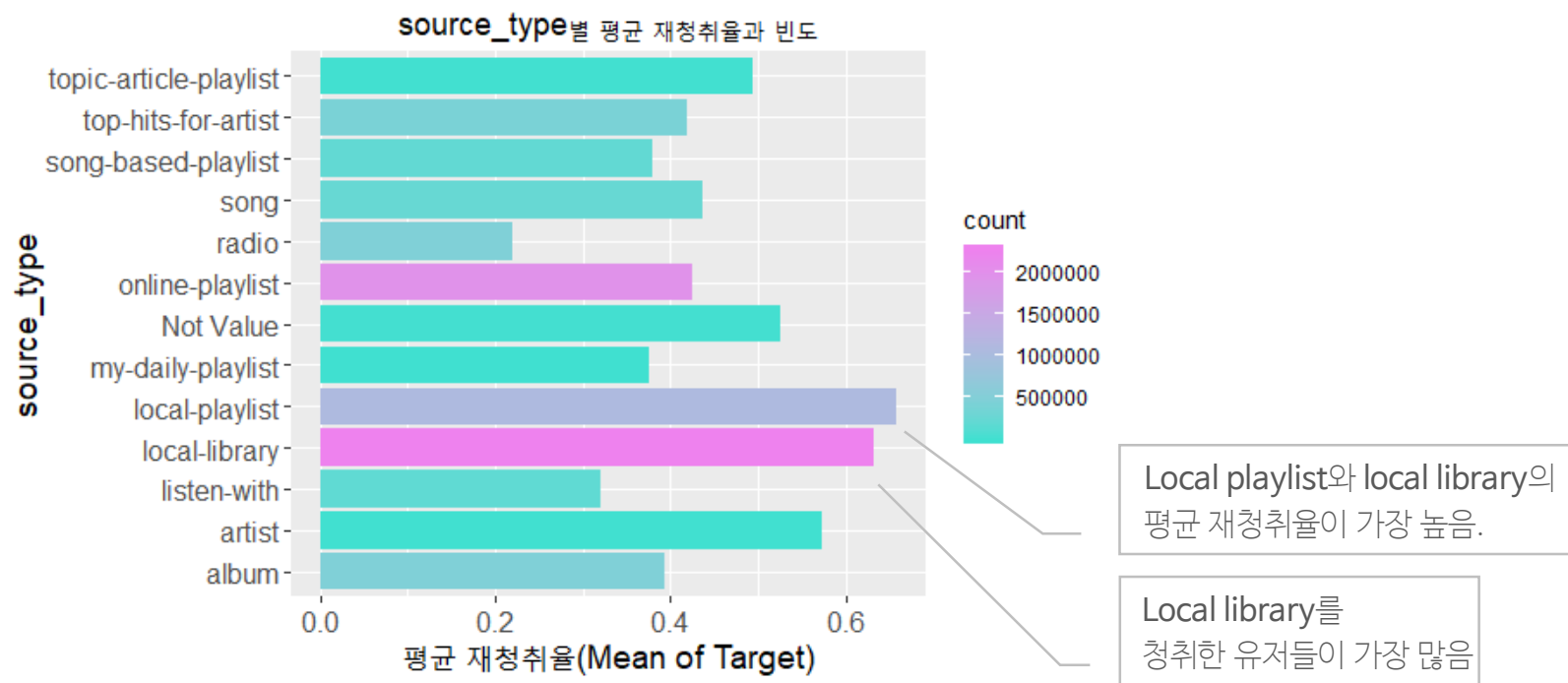


Q. Source\_screen\_name에 따라 재청취율이 어떻게 달라질까?



- **Payment**는 개별적으로 구매한 노래로 추정됨. 따라서 해당 노래에 대한 높은 재청취율은 상식적이라 생각할 수 있음. 하지만, Payment 데이터는 12개밖에 없으므로 평균 재청취율이 큰 의미를 갖기 힘들다.
- 따라서 실제로는 **My library**가 **평균 재청취율이 가장 높다고** 볼 수 있다.
- 대체로 my library와 playlist와 관련된 곳이 재청취율이 높았다. 유저들이 좋아하는 노래를 따로 저장해두고 다시 듣는 습관이 있다는 것을 생각했을 때 납득할 수 있는 결과라 볼 수 있다.

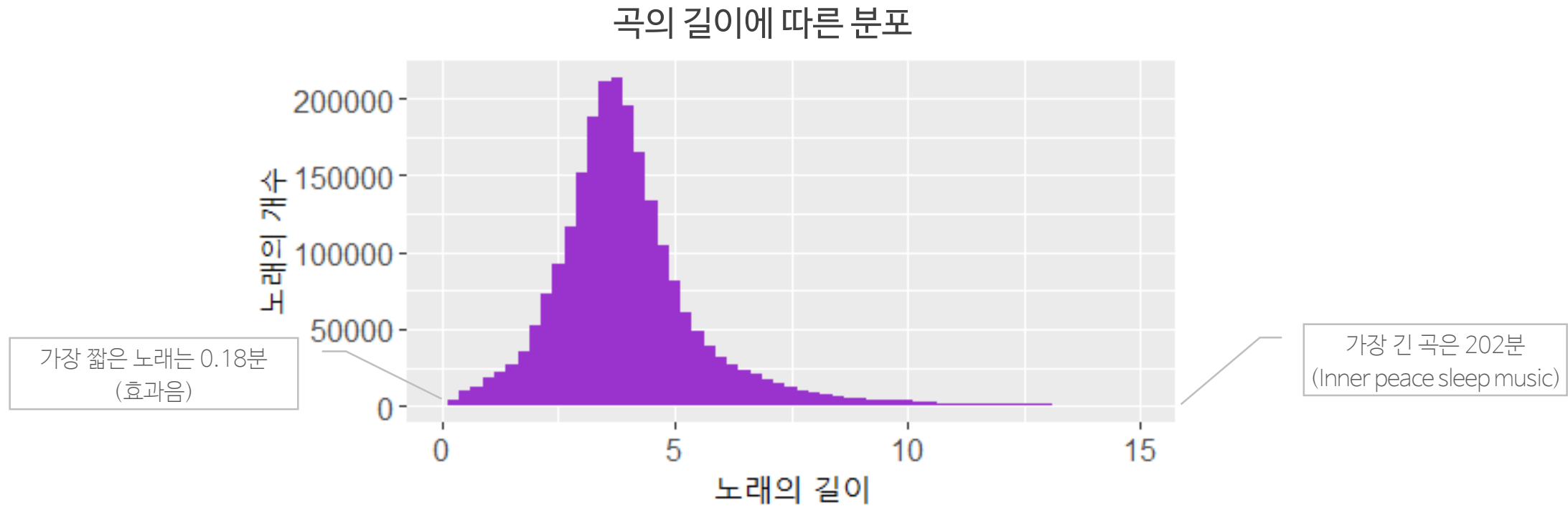
Q. Source\_type에 따라 재청취율이 어떻게 달라질까?



- Local library보다 Local playlist의 재청취율은 비슷했다.
- 하지만, 청취한 유저는 Local library가 훨씬 많았다. 즉, Local playlist보다 Local library를 사람들이 훨씬 더 많이 이용한다고 추측할 수 있다.

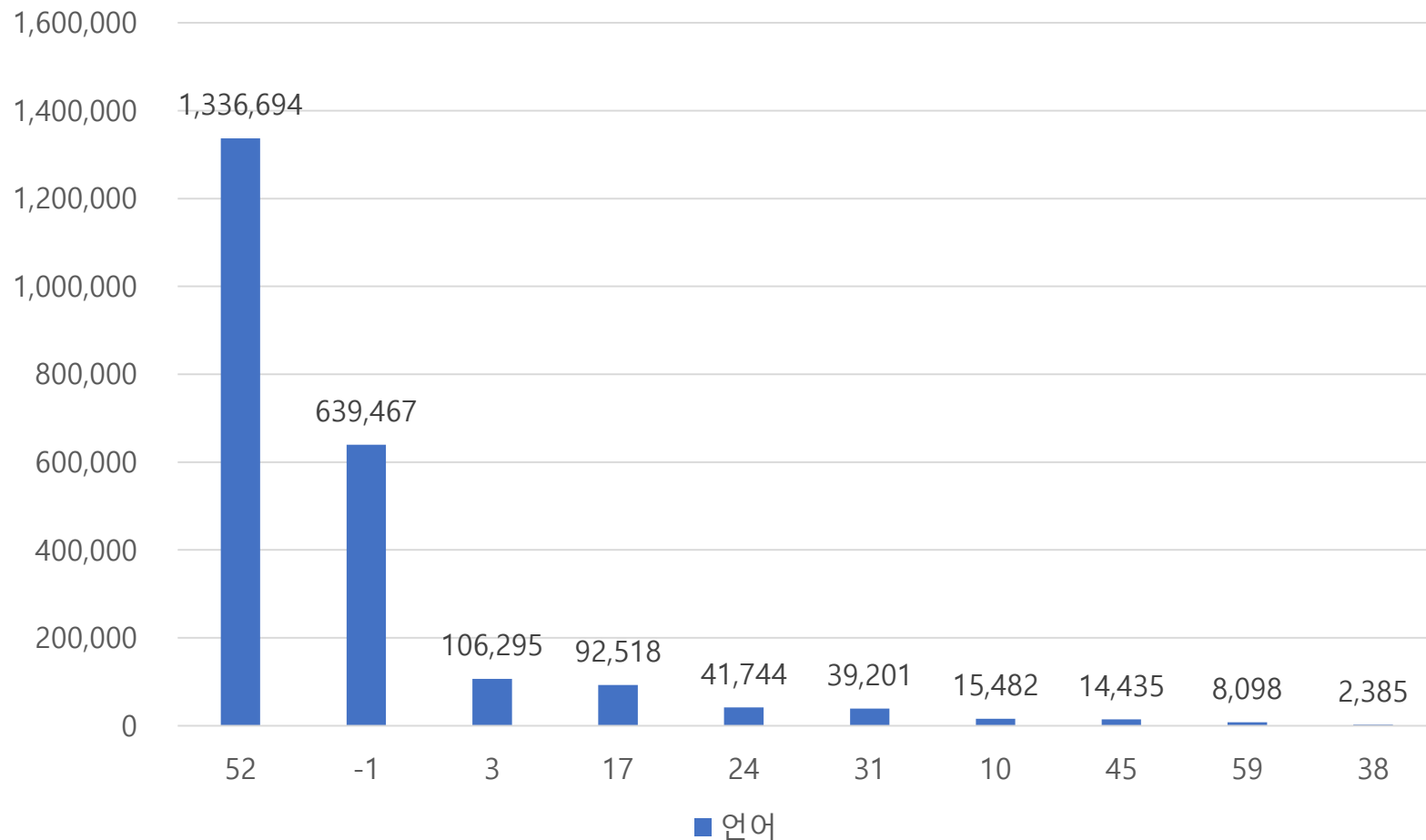
File_name (Rows / Cols)	Variable	Format	Description	Unique rows	NaN (NaN / Total)	Examples
Songs (2,296,320 / 7)	song_id	chr	곡 id	2,296,320	0	“CXoTN1eb7AI+Dntd U1vbcwGRV4SCIDxZ u+YD8JP8r4E= “...
	songs_length	int	곡의 길이 (밀리초 단위*)	146,534	0	247640, 197328, ...
	genre_ids	chr	곡의 장르	1,045	94,116 (4%)	“465”, “352 1995”, “2157”, “465”, “359” ...
	artist_name	chr	가수명	222,363	0	“張信哲 (Jeff Chang)”, “BLACKPINK”, “SUPE R JUNIOR”, ...
	composer	chr	작곡가명	329,823	1,071,354 (46%)	“董貞”, “Maggie Roger s  Nicholas Das”, “JJ L in” ...
	lyricist	chr	작사가명	110,925	1,945,268 (84%)	“何<U+555F>弘”, “”, “Wu Qing Feng”
	language	num	언어	10	1	3, 31, 31, 3, -1, 52, ...

- 장르(genre\_ids)의 결측치(4%)는 “Not Value”로 변환.
- 작곡가의 결측치(46%)와 작사가의 결측치(84%)는 지나치게 많음.
- 유저들이 노래를 재청취할 때 작곡가와 작사가의 영향은 적을 것이라 생각하여 해당 변수들은 모델에 포함시키지 않기로 함.



- 대부분의 노래들의 길이는 2.5분 ~ 5분이다.
- 노래 길이의 평균은 4.1분

언어별 곡의 개수

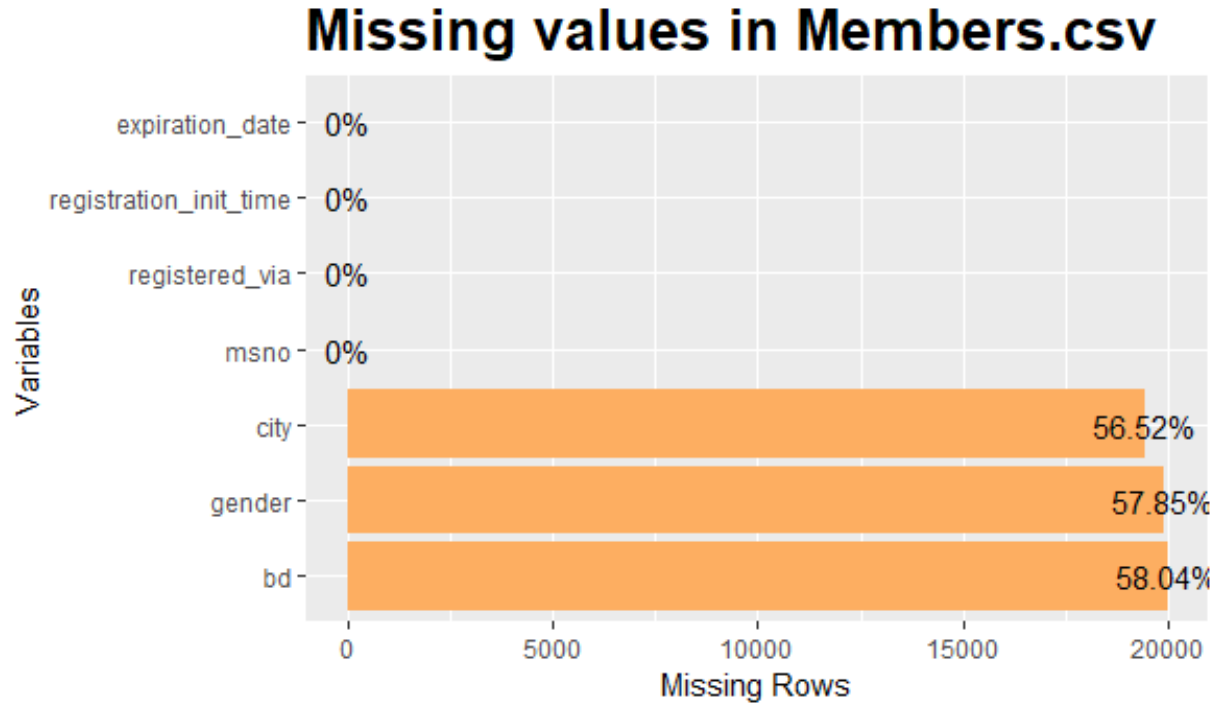


- 가수 이름으로 확인 결과, 3번은 대만어, 52번은 영어, 31번은 한국어였음.
- Language가 -1인 것들은 대부분 작사가도 없고, classic이나 피아노곡들이 많았음.  
가사가 있는 것도 있었지만, MRO이 아닐까 추측했음.

File_name (Rows / Cols)	Variable	Format	Description	Unique rows	NaN (NaN / Total)	Examples
Members (34,403 / 7)	msno	chr	사용자 ID	34,403	0	“mxGXmmQza/nMXJ /Z9140yG06nxd+Nx4 iz2oB/zV6quE=“, ...
	city	int	도시 코드	21	19,445 (56.52%)	1, 13, 5, 1, 1, 1, ...
	bd	int	나이	95	19,969 (58.04%)	0, 0, 0, 33, 20, 0, ...
	gender	chr	성별	2	19,902 (57.85%)	“”, “”, “male”, “”, “”, “f emale”, ...
	registered_via	int	등록 방법	6	0	7, 7, 4, 9, 4, 9, 4, ...
	registration_init_time	int	등록 시작일 (YYYYMMDD)	3,862	0	20110820, 20150628, ...
	expiration_date	int	만료일 (YYYYMMDD)	1,484	0	20170920, 20201017, ...

- 도시(city)와 성별(gender), 나이(bd) 변수는 모두 50% 이상의 결측치가 있음.
- 나이의 경우 0세 이하, 75세 초과는 모두 결측 처리하였음.
- Cart(Classification And Regression Tree) 알고리즘으로 예측하여 채워넣음.






- City가 -1, gender가 공란, bd(나이)가 0인 값들의 개수는 아래와 같이 비슷함.
  - City & Gender & Age: 18,356
  - City & Gender: 18,441
  - City & Age: 18,516
  - Gender & Age: 19,481


즉, 세 개의 변수에서 결측치가 동시에 발생하는 경우가 많다.

### Customize settings

#### Fill out personal info

We use the data below to create personalized playlists and events for you. Your information is safe with us.

 89/9/17

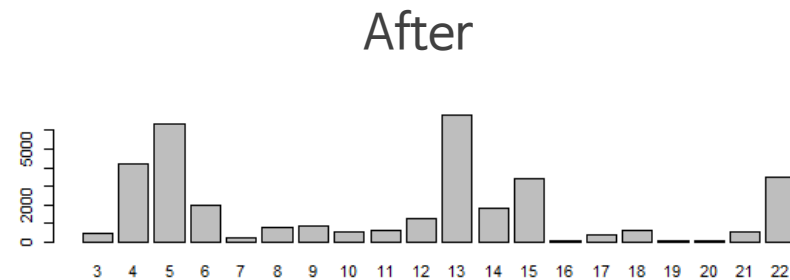
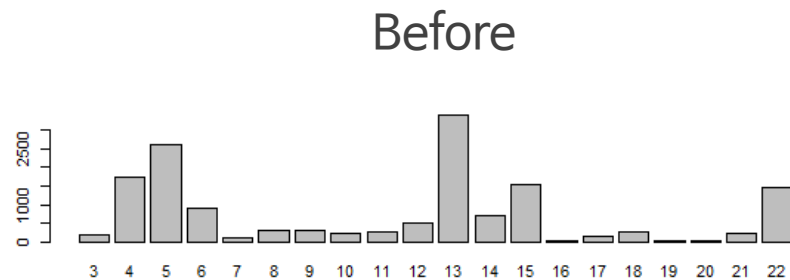
 MALE

Next

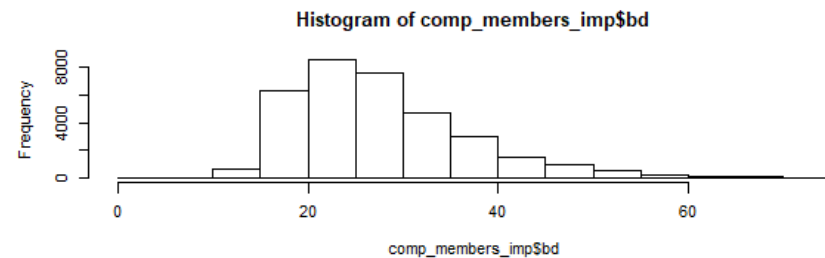
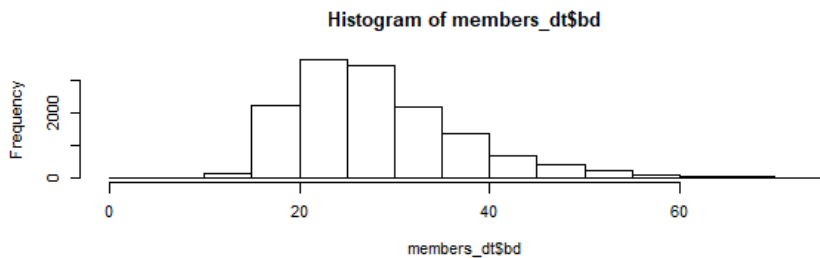
Skip

- 어플 가입시 입력하지 않고 Skip하는 것이 가능.
- 만약 Skip하면, 결측치로 기록되는 것으로 추정.

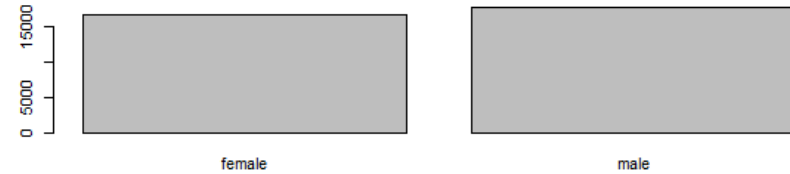
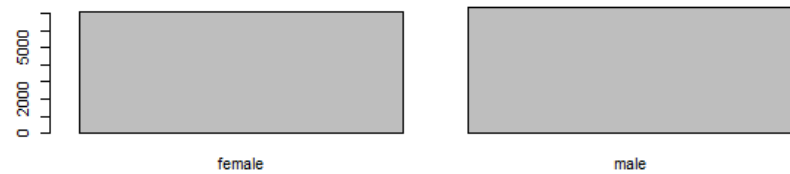
도시별 유저의 분포



나이별 유저의 분포

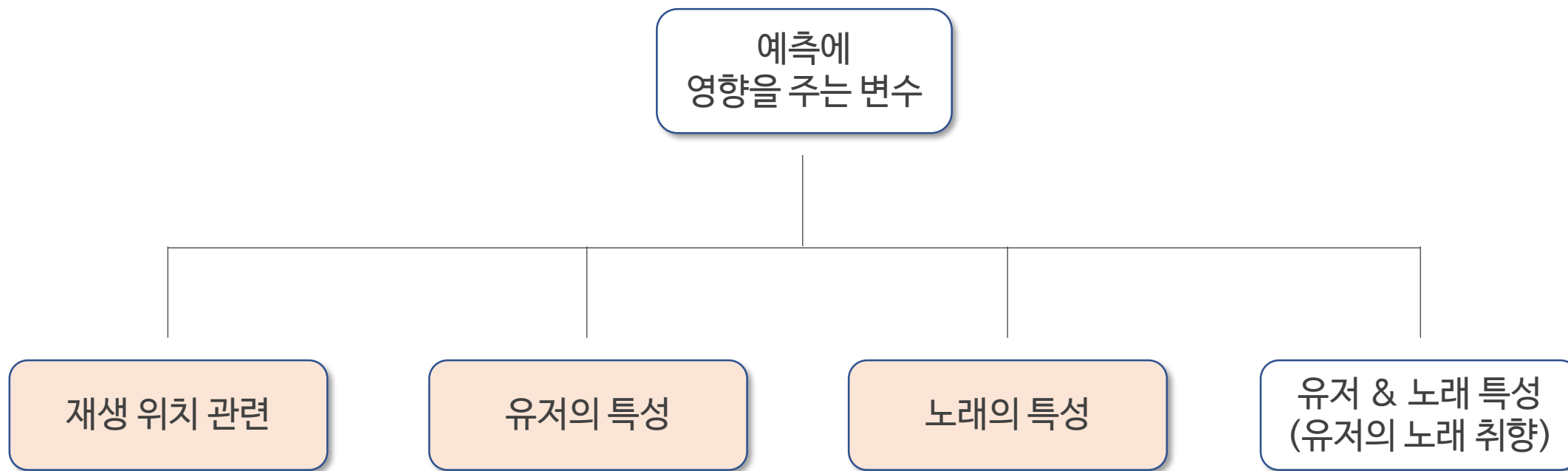


성별 유저의 분포

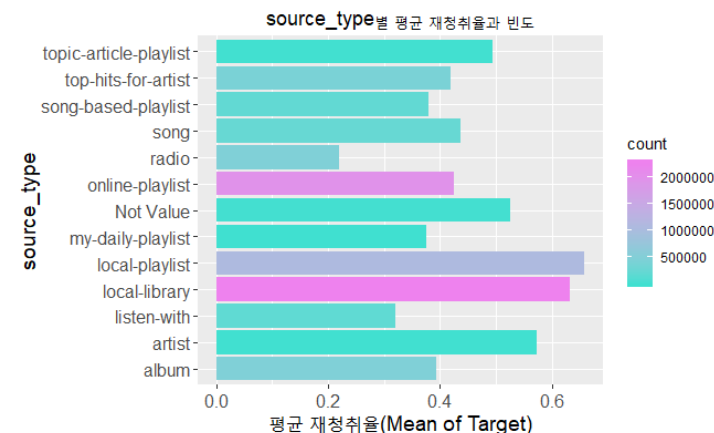
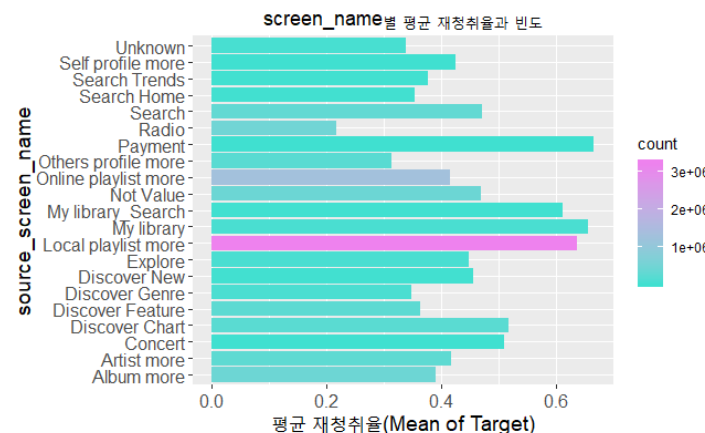
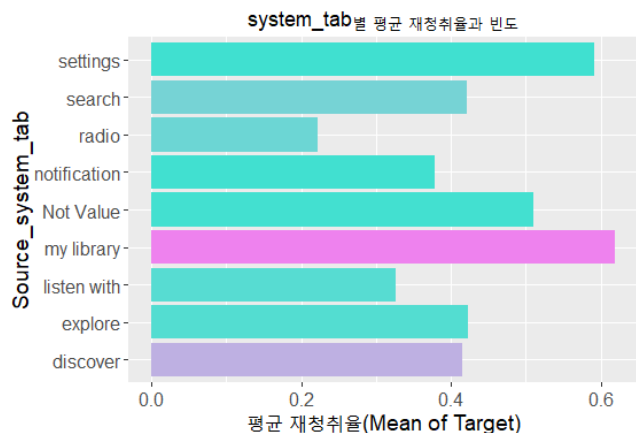


- Mice 패키지의 cart 알고리즘으로 도시, 나이, 성별의 결측치를 예측하여 채워넣음.
- 전과 후의 분포가 크게 다르지 않음.

# Feature Engineering



# Feature Engineering | 재생 위치 관련



- 재생 위치에 따라 재청취율이 다르게 나타나는 것을 확인할 수 있었음.
  - 유저의 재생 리스트에 저장된 곡들의 재청취율이 높게 나타남(my library, local playlist, local library etc...).
- 실제로 어플을 설치하여 살펴본 결과, 한 변수 안에 유사한 기능을 제공하는 다수의 항목들이 있었음  
(ex. 탭 변수의 Discover & Explore → 사실상 동일한 추천 기능)

유사한 기능을 제공하는 항목들끼리 그룹화하여, 좀 더 단순한 범주로 나누었음.

(System\_tab 예시)

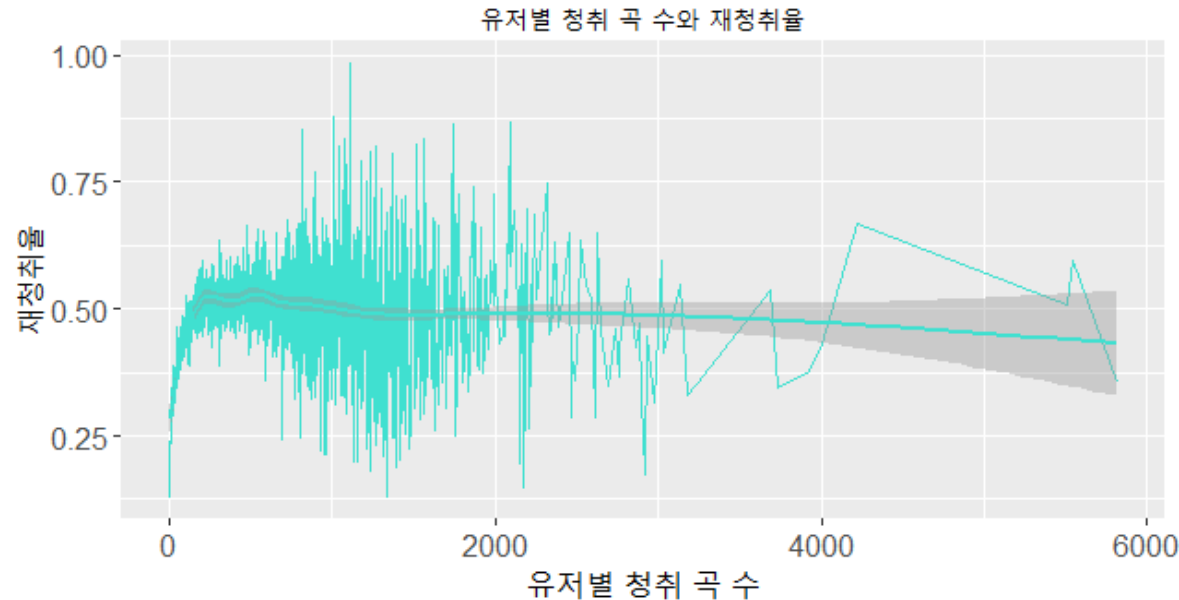
- My library

- Search
- Discover
- Explore
- Radio

- Notification
- Settings

- Not Value

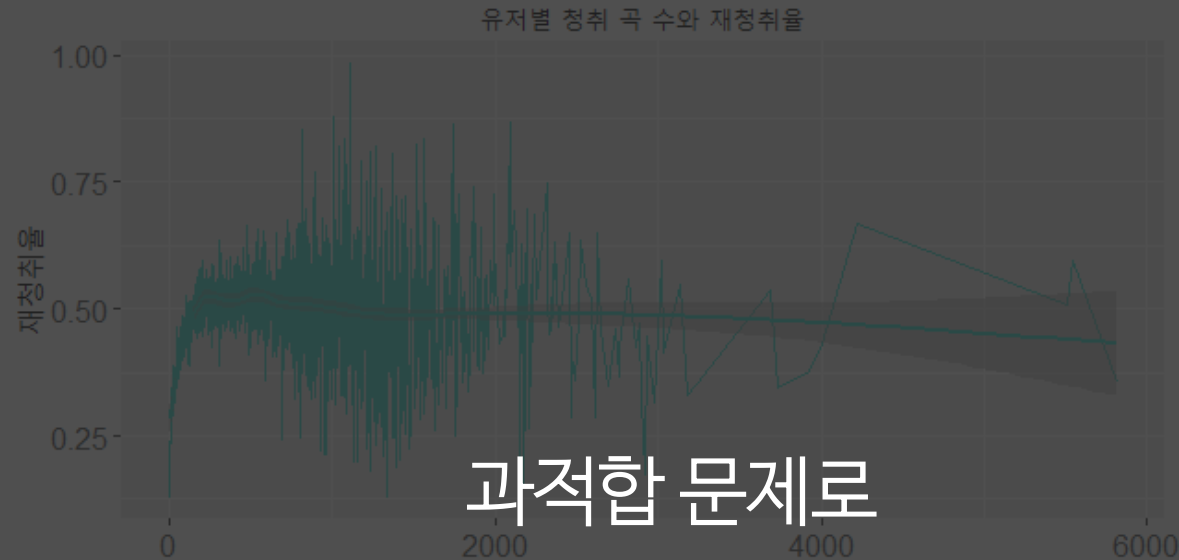
## Feature Engineering | 유저 특성 관련 - 평균 재청취율



- 국소적으로 경향성을 보이는 부분이 보이긴 하지만, 전체적으로 재청취율의 편차가 크다.
- 따라서 단순히 유저 별 청취곡 수를 변수로 사용하기보다,
- 해당 유저가 새로운 노래를 듣기를 좋아하는지, 좋아하는 노래를 또 듣는 것을 좋아하는지를 고려해주는 것이 더욱 바람직할 것으로 예상됨.

$$\text{유저의 평균 재청취율} = \frac{\text{재청취한 노래 수}}{\text{청취한 전체 노래 수}}$$

\* Test 데이터에는 target변수가 없으므로, train에서 구해진 값을 유저ID 기준으로 붙여서 사용함



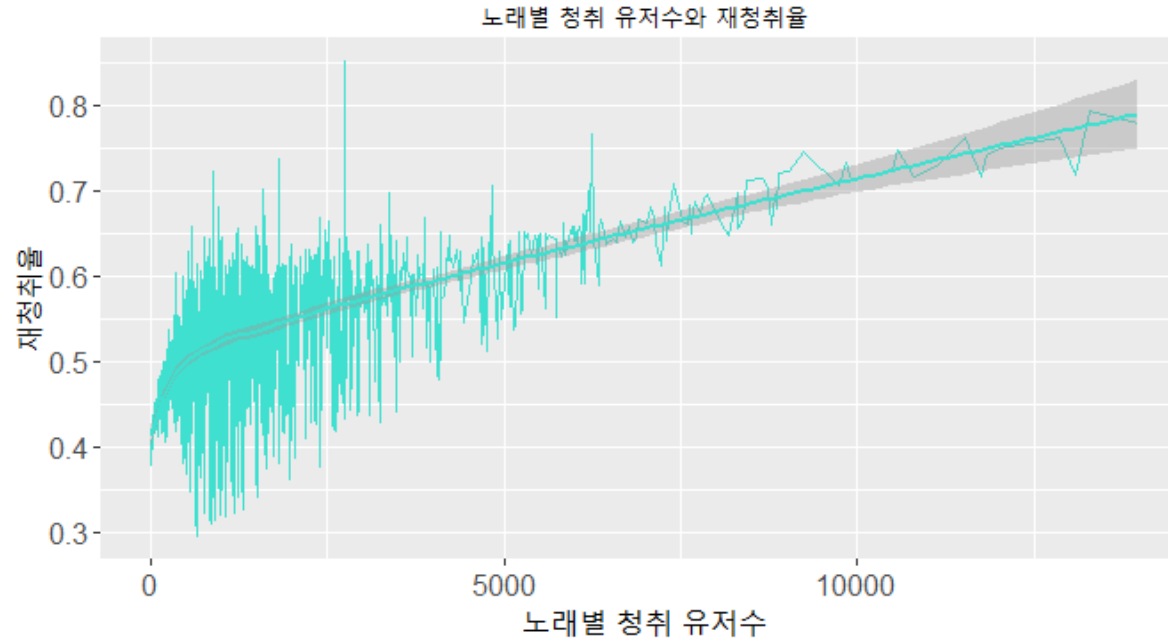
- 국소적으로 경향성을 보이는 부분이 보이긴 하지만, 전체적으로 재청취율의 편차가 크다.
- 따라서 단순히 유저 별 청취곡 수를 변수로 사용하기보다,
- 해당 유저가 새로운 노래를 듣기를 좋아하는지, 좋아하는 노래를 또 듣는 것을 좋아하는지를 고려해주는 것이 더욱 바람직할 것으로 예상됨.

$$\text{유저의 평균 재청취율} = \frac{\text{재청취한 노래 수}}{\text{청취한 전체 노래 수}}$$

\* Test 데이터에는 target변수가 없으므로, train에서 구해진 값을 유저ID 기준으로 붙여서 사용함



## Feature Engineering | 노래 특성 - 1) 노래의 인기도(대중성)



- 앞서 EDA에서 살펴본 바와 같이, 해당 노래를 들은 유저가 많을 수록 재청취율이 비교적 높게 나타나고 있었음.
- 즉, 대중에게 인기가 많은 곡일수록, 재청취율이 높아지는 것으로 추정함.

노래의 인기도 = 데이터에서 노래가 등장한 빈도

## Feature Engineering | 유저 & 노래를 동시에 고려한 특성



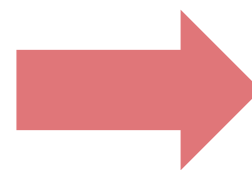
BTS 팬



태진아 팬



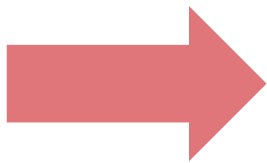
BTS 노래 재청취율은?



같은 BTS의 노래지만,  
각 유저에게 다른 의미를 갖는다  
= 재청취 확률이 다르다!

$$P(\text{BTS 노래 재청취} \mid \text{BTS 팬})$$

$$P(\text{BTS 노래 재청취} \mid \text{태진아 팬})$$



유저가 주어졌을 때 특정 가수의 노래를 재청취할 확률,  
(유저, 가수) 쌍에 대한 **조건부확률 활용**

## Feature Engineering | 유저 & 노래를 동시에 고려한 특성

(기본가정) 아티스트에 대한 선호도가 높으면, 재청취할 확률이 더 높을 것이다



$$\propto \begin{aligned} & \text{(유저 } i \text{의 아티스트A에 대한 선호도)} \\ & = \text{(유저 } i \text{가 청취한 아티스트 A의 곡 수)} / \text{(유저 } i \text{가 청취한 전체 곡 수)} \end{aligned}$$

## Feature Engineering | 유저 & 노래를 동시에 고려한 특성

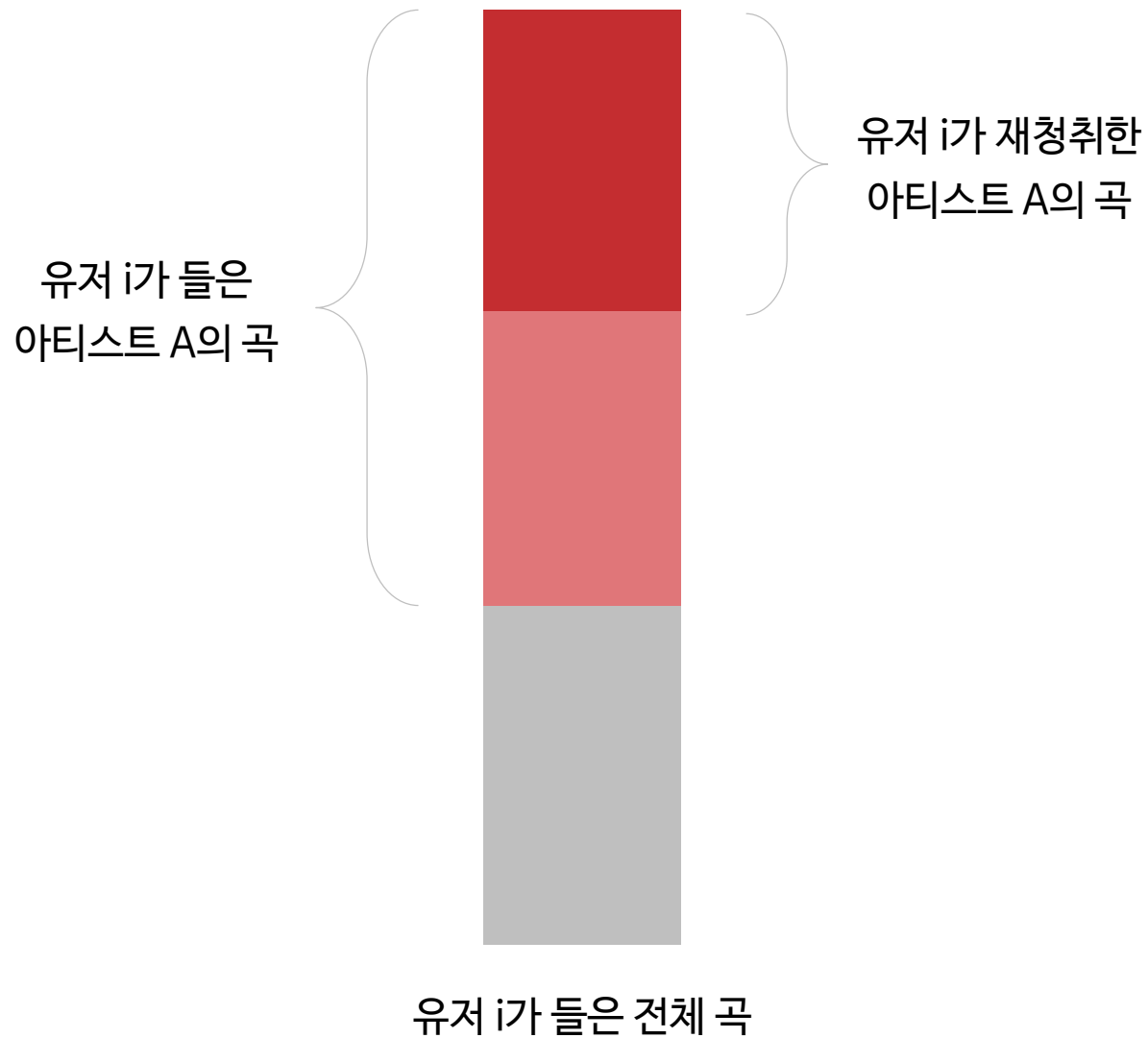
(기본가정) 아티스트에 대한 선호도가 높으면, 재청취할 확률이 더 높을 것이다

$$P(\text{아티스트A의 노래 재청취} \mid \text{유저 } i)$$

$$\propto P(\text{아티스트A의 노래 재청취} \mid \text{아티스트A}, \text{유저 } i)$$

$$\propto (\text{유저 } i \text{의 아티스트A에 대한 재청취율}) \\ = (\text{유저 } i \text{가 재청취한 아티스트 A의 곡 수}) / (\text{유저 } i \text{가 청취한 아티스트A의 곡수})$$

# Feature Engineering | 유저 & 노래를 동시에 고려한 특성



## Feature 1

(유저 i의 아티스트A에 대한 선호도)

$$= \frac{\text{[Medium Red Box]} \text{ (유저 i가 청취한 아티스트A의 곡 수)}}{\text{[Gray Box]} \text{ (유저 i가 청취한 전체 곡 수)}}$$

## Feature 2

(유저 i의 아티스트A에 대한 선호도)

$$= \frac{\text{[Dark Red Box]} \text{ (유저 i가 재청취한 아티스트A의 곡 수)}}{\text{[Medium Red Box]} \text{ (유저 i가 청취한 아티스트A의 곡 수)}}$$

동일한 방식을 아티스트 뿐만 아니라,  
노래의 장르, 언어, 국가에도 적용하여  
각각에 대한 조건부 확률을 구하여 변수로 사용했다.

$$P(y=1 | \text{장르, 유저}) = \frac{\text{유저 } i \text{가 청취한 아티스트A의 곡 수}}{\text{유저 } i \text{가 청취한 전체 곡 수}}$$
$$P(y=1 | \text{언어, 유저}) = \frac{\text{유저 } i \text{의 아티스트A에 대한 선호도}}{\text{유저 } i \text{가 청취한 아티스트A의 곡 수}}$$
$$P(y=1 | \text{국가, 유저}) = \frac{\text{유저 } i \text{의 아티스트A에 대한 선호도}}{\text{유저 } i \text{가 청취한 아티스트A의 곡 수}}$$

유저 i가 들은  
아티스트 A의 곡



$P(y=1 | \text{장르, 유저})$   
유저 i가 들은 전체 곡



$P(y=1 | \text{언어, 유저})$



$P(y=1 | \text{국가, 유저})$

유저 i가 들은  
아티스트 A의 곡

유저 i가 재청취한  
아티스트 A의 곡  
But, 해결하지 못한 문제점 2가지

① Test 데이터에서 새로 등장하는 유저들로 인해 발생하는 문제

② 특정 아티스트(장르, 언어 etc)의 노래를 한 곡만 들었고,  
해당 노래를 재청취한 경우에도 관심도가 높게 나타나는 문제

유저 i가 들은 전체 곡

Feature 1

(유저 i의 아티스트A에 대한 선호도)

(유저 i가 청취한 아티스트A의 곡 수)

=

(유저 i가 청취한 전체 곡 수)

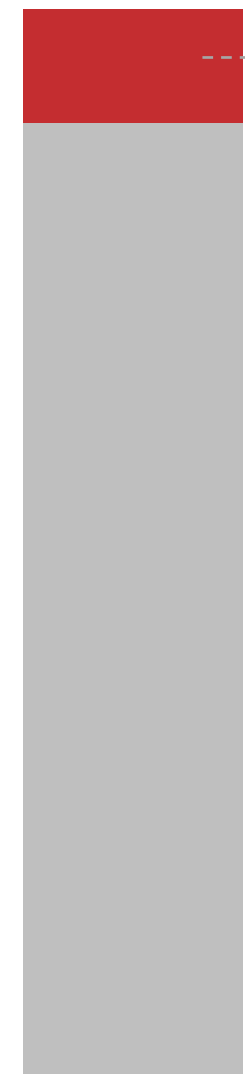
=

(유저 i가 재청취한 아티스트A의 곡 수)

(유저 i가 청취한 아티스트A의 곡 수)



# Feature Engineering | (참고)복수의 장르를 처리한 방식



전체 곡 수  
2,296,320개

복수의 장르가 기입된 곡  
**12%** (266,992곡)



최초 기입된 **1개의 장르**를  
해당 곡의 **대표 장르**로 사용

가정1

?



일반적으로 노래의 장르를 떠올릴 때,  
대표적인 하나의 장르로 기억

가정2

465 | 921

≠

921 | 465

4cA1bEi+ZYpee0yh693pq6n8VRaohvwy3TGz/E1PvBc=	465   921
4cNd187p2ehUxC4jMg14pGr4rCo9uCUJLEtx3lCftSs=	465   921
4w3+iWBbB04FZK+Qu8vf76h/ZyLLuc+4Sov7ch0STDY=	465   921

lS53287sFNLLvH8iCefpYTwtQ+vGkbGJRzq1ZzrZfY=	921   465
r6GeBYvk+23DbUZLndM1iVTng19i6gGqBNH8ChYowo=	921   465
/GgBz281lYZcwQ3M0kS2dTZHGR861evmCjrAg8wQjM=	921   465

동일한 장르 번호를 가진 노래들의  
장르 기입 순서가 다름

Modeling



# Modeling

```
> glimpse(both3)
Observations: 9,934,208
Variables: 11
$ msno          <chr> "++5wYjoMgQHoRuD3GbbvmphZbBBwymzv5Q4l8sywtuU=", "++5wYjoMgQHoRuD3GbbvmphZbBBwymzv...
$ song_id       <chr> "+/lcxtBy9FuH00bLsK9wRf3z19zSyvDNMpTWSGCAXxc=", "+kkZYh2T9gs8DuRGpLK407VJOlt9eUjL...
$ id            <int> 5944833, 6125574, 4810302, 6753212, 5980353, 5975160, 5977522, 4704944, 5593686, ...
$ play_count    <int> 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 757, 7...
$ artist_msno_prob <dbl> 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.0488771...
$ artist_mean_target <dbl> 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.35...
$ genre_msno_prob <dbl> 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.04887715, 0.0488771...
$ genre_mean_target <dbl> 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.3513514, 0.35...
$ country_msno_prob <dbl> 0.1109643, 0.1109643, 0.1109643, 0.1109643, 0.1109643, 0.1109643, 0.1109643, 0.11...
$ language_msno_prob <dbl> 0.08322325, 0.08322325, 0.08322325, 0.08322325, 0.08322325, 0.08322325, 0.08322325, 0.0832232...
$ user_mean_target <dbl> 0.4974533, 0.4974533, 0.4974533, 0.4974533, 0.4974533, 0.4974533, 0.4974533, 0.49...
```

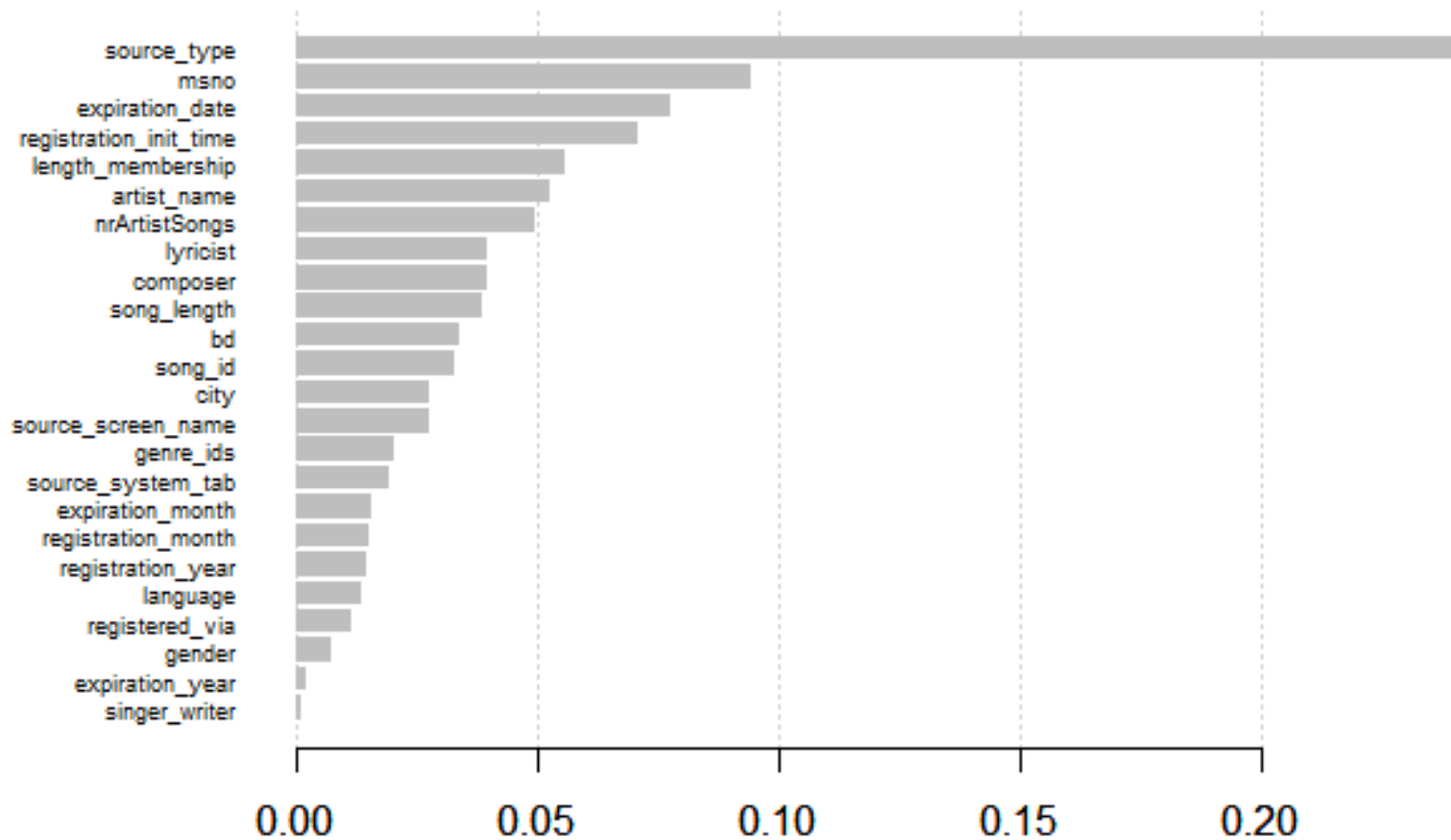
# Modeling

```
> glimpse(both4)
Observations: 9,934,210
Variables: 25
$ msno      <chr> "//4hBneqk/4/TtgL1XXQ+eKx7fJTeSvSNt0ktxjSIYE=", "//4hBneqk/4/TtgL1XXQ+eKx7fJTeSvS...
$ song_id   <chr> "/1DkMXTGMVH60jiFoia7+DZMucB8l1A4w7I9O6Ntti8=", "/424yvSx/6ozuKUGm6Y3kRXbJwiktyKJ...
$ id        <int> 6176262, 6114664, 7269212, 368891, 5621965, 6475944, 230898, 363286, 6938082, 687...
$ target    <int> 0, 0, 0, -1, 0, 1, -1, -1, 1, 1, 1, -1, 1, 1, -1, 1, -1, -1, 1, -1, -1, 1, 0, 0, ...
$ city      <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 1...
$ bd        <int> 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 1...
$ gender    <chr> "female", "female", "female", "female", "female", "female", "female", "female", "...
$ registered_via <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, ...
$ length_membership <int> 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 1015, 101...
$ song_length <int> 248737, 199157, 305760, 262687, 203592, 221239, 328176, 304692, 149211, 258638, 2...
$ language  <int> 52, 52, 52, 3, 52, 52, 52, 3, 52, 52, 3, 3, 52, 52, 52, 52, 17, 52, 52, 52, 52, 5...
$ genre_1   <int> 465, 465, 465, 458, 465, 921, 1609, 465, 465, 2022, 465, 465, 1259, 1259, 2022, 3...
$ cc        <chr> "GB", "US", "FR", "TW", "US", "GB", "US", "TW", "GB", "QM", "TW", "TW", "US", "US...
$ singer_writer <int> 1, 1, 2, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
$ nrArtistSongs <int> 158, 53, 252, 25, 269, 193, 539, 201, 117, 28, 52, 152, 18, 1, 125, 22, 387, 25, ...
$ sst       <int> 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, ...
$ ssn       <int> 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 3, 0, 3, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, ...
$ st        <int> 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, ...
$ artist_msno_prob <dbl> 0.0028462998, 0.0180265655, 0.0009487666, 0.0018975332, 0.0246679317, 0.002846299...
$ artist_mean_target <dbl> 0.33333333, 0.36842105, 0.00000000, NA, 0.65217391, 0.66666667, 0.07317073, NA, 0...
$ genre_msno_prob <dbl> 0.50853890, 0.50853890, 0.50853890, 0.07590133, 0.50853890, 0.05882353, 0.0588235...
$ genre_mean_target <dbl> 0.4768392, 0.4768392, 0.4768392, 0.4400000, 0.4768392, 0.3793103, 0.4000000, 0.47...
$ country_msno_prob <dbl> 0.186907021, 0.360531309, 0.004743833, 0.211574953, 0.360531309, 0.186907021, 0.3...
$ language_msno_prob <dbl> 0.6347249, 0.6347249, 0.6347249, 0.2352941, 0.6347249, 0.6347249, 0.6347249, 0.23...
$ user_mean_target <dbl> 0.4031117, 0.4031117, 0.4031117, 0.4031117, 0.4031117, 0.4031117, 0.4031117, 0.40...
> |
```

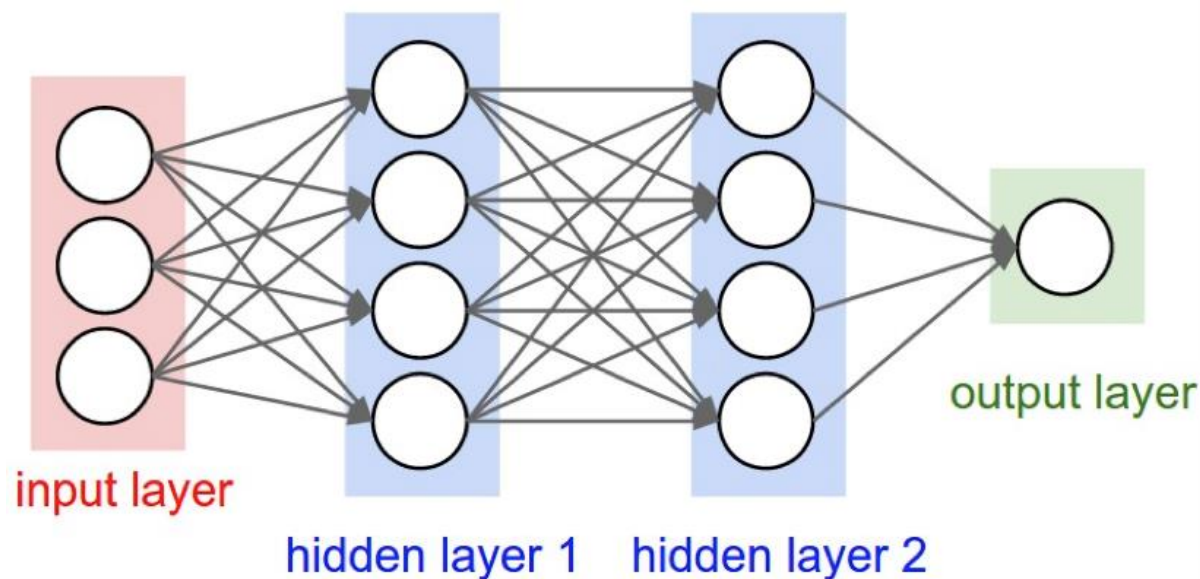
# Modeling

```
> glimpse(both)
Observations: 9,934,208
Variables: 25
$ song_id      <int> 6759, 6756, 6756, 6756, 6757, 6757, 6757,...
$ msno         <int> 28415, 588, 9068, 16242, 18674, 21331, 64...
$ source_system_tab <int> 2, 2, 2, 2, 5, 5, 2, 5, 5, 9, 2, 9, 2, 9,...
$ source_screen_name <int> 1, 13, 13, 13, 10, 10, 2, 2, 10, 2, 6, 19...
$ source_type   <int> 8, 8, 8, 11, 6, 6, 2, 2, 5, 2, 11, 10, 12...
$ target        <dbl> 0, 0, -1, 0, 0, 0, 0, -1, -1, 0, 0, -1, 0...
$ id            <int> 4045319, 7231155, 405746, 2425582, 326533...
$ city          <int> 1, 1, 22, 5, 13, 1, 1, 1, 1, 1, 9, 6, 11,...
$ bd            <int> 0, 0, 28, 0, 39, 30, 0, 0, 0, 0, 31, 26, ...
$ gender        <int> 1, 1, 2, 1, 2, 2, 1, 1, 1, 1, 3, 2, 3, 3,...
$ registered_via <int> 7, 7, 7, 3, 9, 4, 7, 7, 7, 7, 3, 9, 4, 7,...
$ registration_init_time <dbl> 17054, 17188, 15893, 15702, 13086, 17127,...
$ expiration_date <dbl> 17392, 17430, 17721, 17427, 17439, 17220,...
$ registration_year <int> 2016, 2017, 2013, 2012, 2005, 2016, 2011,...
$ registration_month <int> 9, 1, 7, 12, 10, 11, 3, 8, 1, 12, 2, 3, 1...
$ expiration_year <int> 2017, 2017, 2018, 2017, 2017, 2017, 2017,...
$ expiration_month <int> 8, 9, 7, 9, 9, 2, 10, 9, 9, 10, 9, 9, 3, ...
$ song_length   <int> 223921, 271302, 271302, 271302, 221413, 2...
$ genre_ids     <int> 550, 373, 373, 373, 478, 478, 478, 478, 3...
$ artist_name   <int> 3663, 1698, 1698, 1698, 13916, 13916, 139...
$ composer      <int> 256052, 222366, 222366, 222366, 222366, 2...
$ lyricist      <int> 549531, 222366, 222366, 222366, 222366, 2...
$ language      <int> 45, 52, 52, 52, -1, -1, -1, -1, 52, 52, -...
$ singer_writer <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
$ length_membership <dbl> 338, 242, 1828, 1725, 4353, 93, 2396, 396...
```

XGBOOST로 본 각 변수의 중요도



# Modeling



```
model <- keras_model_sequential() # 순차적 모델 방법
model %>%
  layer_dense(
    units= 128,
    input_shape = c(ncol(x_train)),
    kernel_initializer='he_normal'
  ) %>%
  layer_activation("relu") %>%
  layer_batch_normalization() %>%
  layer_dropout(rate= 0.1) %>%

  layer_dense(
    units= 512,
    kernel_initializer='he_normal'
  ) %>%
  layer_activation("relu") %>%
  layer_batch_normalization() %>%
  layer_dropout(rate= 0.3) %>%

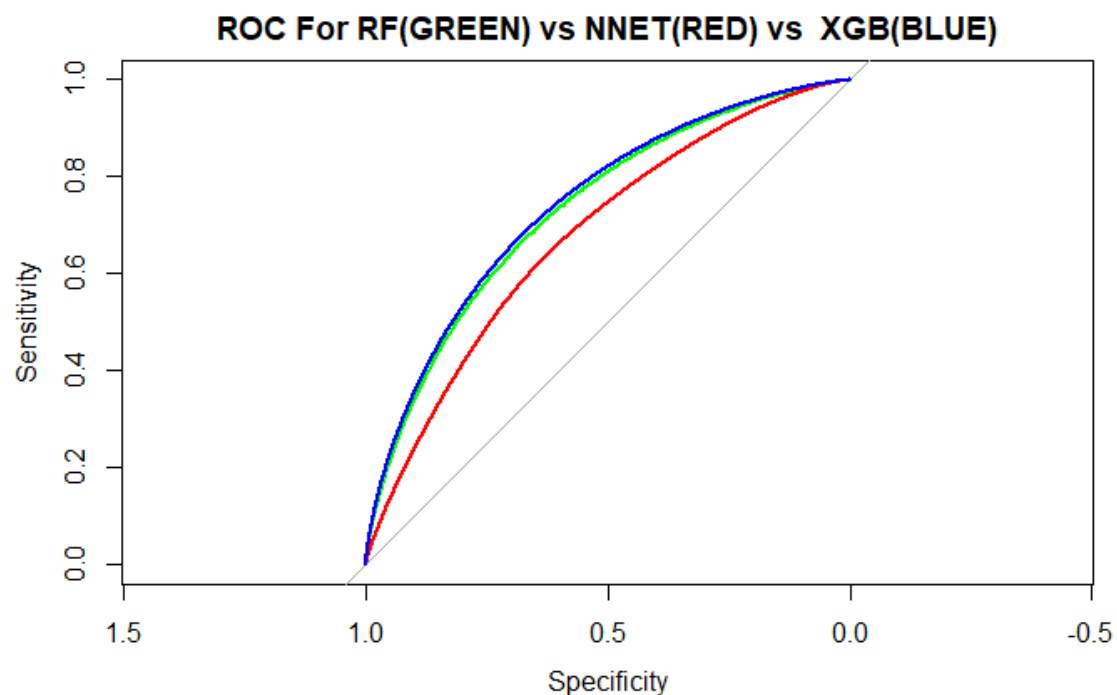
  layer_dense(
    units= 64,
    kernel_initializer='he_normal'
  ) %>%
  layer_activation("relu") %>%
  layer_batch_normalization() %>%
  layer_dropout(rate= 0.1) %>%

  layer_dense(2) %>%
  layer_activation("softmax")

model %>% compile(
  loss = 'categorical_crossentropy',
  optimizer = optimizer_rmsprop(),
  metrics = c("accuracy")
)

history <- model %>% fit(
  x_train, y_train, verbose=2,
  view_metrics = FALSE,
  epochs = 7, batch_size=B_Size,
  validation_split = 0.1
)
```





```
> diagnosis(y_ens_val, pred_1_e, title="ranger")
```

Summary results for ranger

```
auc: 0.7309967
acc: 0.6709923
bce: 0.3140447
mll: Inf
rmse: 0.4592146
```

```
> diagnosis(y_ens_val, pred_3_e, title="xgb")
```

Summary results for xgb

```
auc: 0.7402587
acc: 0.6779543
bce: 0.3042186
mll: 0.6077121
rmse: 0.4563571
```

Summary results for keras

```
auc: 0.6753066
acc: 0.632089
bce: 0.3133689
mll: 0.645039
rmse: 0.4760453
```

XGBOOST - 랜덤포레스트 모델 - 신경망 모델 순으로 AUC가 높게 나타났다.

## Stacking 알고리즘

```

```{r using optim}
cat("\n=== Optimizing for ensemble start:", format(Sys.time()), "===\n")

e_m<- cbind(v1= to_p(pred_1_e),
            v2= to_p(pred_2_e),
            v3= to_p(pred_3_e))

t_m<- cbind(v1= to_p(pred_1_t),
            v2= to_p(pred_2_t),
            v3= to_p(pred_3_t))

m_fun <- function(par){
  return(1-auc(y_ens_val, (e_m %*% par)/sum(par)))
}
start_par=rep(1, ncol(e_m))/ ncol(e_m)
a <- optim(start_par, m_fun, method= "L-BFGS-B",
           lower = 0, upper = 1)

#
submit_prediction <- (t_m %*% a$par)/sum(a$par) # blended model
submit_prediction_rf <- pred_1_e # ranger
submit_prediction_keras <- pred_2_e # keras
submit_prediction_xgb <- pred_3_e # xgb

```

```

> diagnosis(y_ens_val, (e_m %*% a$par)/sum(a$par), title="optimizer"); cat(

Summary results for optimizer
auc: 0.7605575
acc: 0.6930984
bce: 0.2904171
mll: 0.583196
rmse: 0.4463667

```

# Can you build the best music recommendation system?

Prize Money



KKBOX · 1,081 teams · 10 months ago

[Overview](#)

[Data](#)

[Kernels](#)

[Discussion](#)

[Leaderboard](#)

[Rules](#)

[Team](#)

[My Submissions](#)

[Late Submission](#)

## Your most recent submission

Name	Submitted	Wait time	Execution time	Score
Ex1.csv	a few seconds ago	5 seconds	31 seconds	0.64792
Complete				

[Jump to your position on the leaderboard ▼](#)

[Public Leaderboard](#)

[Private Leaderboard](#)

The private leaderboard is calculated with approximately 50% of the test data.

This competition has completed. This leaderboard reflects the final standings.

[Refresh](#)

■ In the money

감사합니다