고객의 전체 모습을 파악하는 테크닉

파이썬을 이용한 실무 데이터 분석

데이터 집계

분석의 전제조건

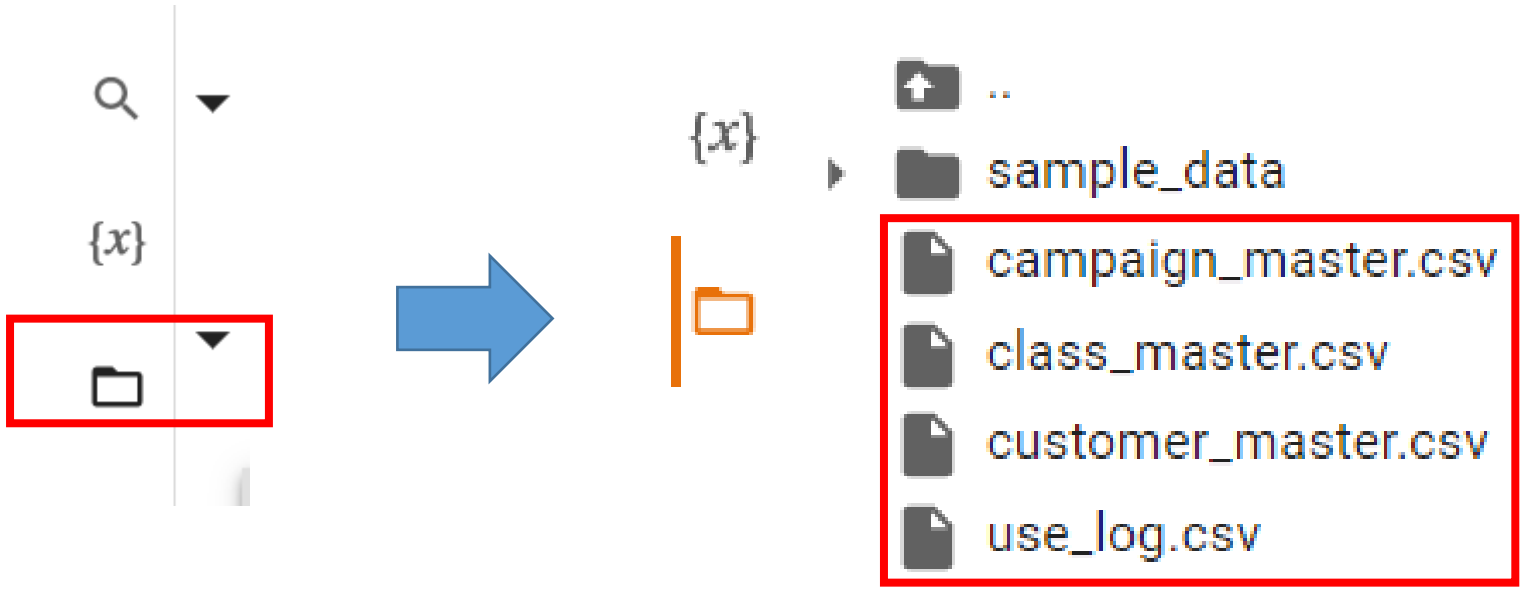
- 고객데이터 및 고객의 피트니스센터 이용 기록
- 센터를 언제든 사용할 수 있는 종일 회원, 낮에만 사용할 수 있는 주간회원, 밤에만 사용할 수 있는 야간회원, 이렇게 3가지 종류의 회원 구분이 있다.
- 일반적으로 입회비가 들지만, 비정기적으로 입회비 반액 할인이나 무료 행사를 한다.
- 월말까지 탈퇴 신청을 하면 그 다음 달 말에 탈퇴가 된다.

	파일 이름	개요
1	use_log.csv	회원의 센터 이용 이력 데이터
2	customer_master.csv	2019년 3월 말 시점의 회원 데이터
3	class_master.csv	회원 구분 데이터(종일, 주간, 야간)
4	campaign_master.csv	행사 구분 데이터(입회비 유무 등)

데이터 불러오기

데이터 업로드

- 데이터 파일을 구글 드라이브 임시 공간에 업로드



데이터 불러오기

데이터 불러오기

- pandas를 이용하여 use_log.csv 파일을 데이터 프레임으로 불러온다.
- 어떤 고객이 언제 센터를 이용했는지 알 수 있는 데이터이다. (총 197,428개)

```
import pandas as pd
uselog = pd.read_csv('use_log.csv')
print(len(uselog))
uselog.head()
```

197428

	log_id	customer_id	usedate
0	L00000049012330	AS009373	2018-04-01
1	L00000049012331	AS015315	2018-04-01
2	L00000049012332	AS040841	2018-04-01
3	L00000049012333	AS046594	2018-04-01
4	L00000049012334	AS073285	2018-04-01

데이터 불러오기

데이터 불러오기

- pandas를 이용하여 customer_master.csv 파일을 데이터 프레임으로 불러온다.
- 이름은 비식별화 처리, is_delete는 2019년 3월 시점에 탈퇴한 유저 식별자 (0이면 유지)

```
customer = pd.read_csv('customer_master.csv')
print(len(customer))
customer.head()
```

4192

	customer_id	name	class	gender	start_date	end_date	campaign_id	is_deleted
0	OA832399	XXXX	C01	F	2015-05-01 00:00:00	NaN	CA1	0
1	PL270116	XXXXX	C01	M	2015-05-01 00:00:00	NaN	CA1	0
2	OA974876	XXXXX	C01	M	2015-05-01 00:00:00	NaN	CA1	0
3	HD024127	XXXXX	C01	F	2015-05-01 00:00:00	NaN	CA1	0
4	HD661448	XXXXX	C03	F	2015-05-01 00:00:00	NaN	CA1	0

데이터 불러오기

데이터 불러오기

- pandas를 이용하여 `class_master.csv` 파일을 데이터 프레임으로 불러온다.
- 회원 구분 데이터로 class를 이용해서 회원 데이터와 결합할 수 있다.

```
class_master = pd.read_csv('class_master.csv')
print(len(class_master))
class_master.head()
```

```
3
```

	class	class_name	price
0	C01	0_종일	10500
1	C02	1_주간	7500
2	C03	2_야간	6000

데이터 불러오기

데이터 불러오기

- pandas를 이용하여 campaign_master.csv 파일을 데이터 프레임으로 불러온다.
- 캠페인 구분 데이터로 campaign_id를 이용해 회원 데이터와 결합할 수 있다.

```
campaign_master = pd.read_csv('campaign_master.csv')
print(len(campaign_master))
campaign_master.head()
```

```
3
```

	campaign_id	campaign_name
0	CA1	2_일반
1	CA2	0_입회비반액할인
2	CA3	1_입회비무료

데이터 가공

고객 데이터 가공

- customer에 class_master(회원 구분)와 campaign_master(캠페인 구분)를 결합해서 customer_join 생성

```
customer_join = pd.merge(customer, class_master, on="class", how="left")
customer_join = pd.merge(customer_join, campaign_master,
                         on="campaign_id", how="left")
customer_join.head()
```

	customer_id	name	class	gender	start_date	end_date	campaign_id	is_deleted	class_name	price	campaign_name
0	OA832399	XXXX	C01	F	2015-05-01 00:00:00	NaN	CA1	0	0_종일	10500	2_일반
1	PL270116	XXXXX	C01	M	2015-05-01 00:00:00	NaN	CA1	0	0_종일	10500	2_일반
2	OA974876	XXXXX	C01	M	2015-05-01 00:00:00	NaN	CA1	0	0_종일	10500	2_일반
3	HD024127	XXXXX	C01	F	2015-05-01 00:00:00	NaN	CA1	0	0_종일	10500	2_일반
4	HD661448	XXXXX	C03	F	2015-05-01 00:00:00	NaN	CA1	0	2_야간	6000	2_일반

데이터 가공

고객 데이터 가공

- 데이터 개수 확인

```
print(len(customer))
print(len(customer_join))
```

```
4192
4192
```

데이터 가공

고객 데이터 가공

- 조인할 때 키가 없거나 조인이 잘못되면 결측치가 들어가므로 결측치가 있는지 확인한다.
- end_date에 결측치가 있는 이유는 탈퇴하지 않은 회원의 탈퇴일이 공백이기 때문이다.

```
customer_join.isnull().sum()
```

```
customer_id         0
name                0
class               0
gender              0
start_date          0
end_date         2842
campaign_id         0
is_deleted          0
class_name          0
price               0
campaign_name       0
dtype: int64
```

데이터 집계

클래스 별 고객 수

```
customer_join.groupby("class_name").count()["customer_id"]
```

```
class_name
0_종일     2045
1_주간     1019
2_야간     1128
Name: customer_id, dtype: int64
```

데이터 집계

캠페인 별 고객 수

```
customer_join.groupby("campaign_name").count()["customer_id"]
```

```
campaign_name
0_입회비반액할인      650
1_입회비무료         492
2_일반            3050
Name: customer_id, dtype: int64
```

데이터 집계

성별 고객 수

```
customer_join.groupby("gender").count()["customer_id"]
```

```
gender
F    1983
M    2209
Name: customer_id, dtype: int64
```

데이터 집계

회원 자격을 유지하고 있는 회원과 탈퇴 회원 수

```
customer_join.groupby("is_deleted").count()["customer_id"]
```

```
is_deleted
0    2842
1    1350
Name: customer_id, dtype: int64
```

데이터 집계

특정 기간 동안 가입한 인원

- 2018년 4월 1일 이후부터 2019년 3월 31일까지 가입한 인원을 집계한다.

```
customer_join["start_date"] = pd.to_datetime(customer_join["start_date"])
customer_start = customer_join.loc[customer_join["start_date"]>=pd.to_datetime("20180401")]
print(len(customer_start))
```

```
1361
```

최근 고객 데이터 집계

가장 최근 월인 2019년 3월의 고객 데이터 파악

- is_deleted 열을 사용해 추출하면 2019년 3월 31일에 탈퇴한 고객은 카운트 되지 않으므로 주의
- 2019년 3월 31일에 탈퇴한 고객과 재적 중인 고객 추출

```
customer_join["end_date"] = pd.to_datetime(customer_join["end_date"])
customer_newer = customer_join.loc[(customer_join["end_date"]>=pd.to_datetime("20190331"))|(customer_join["end_date"].isna())]
print(len(customer_newer))
customer_newer["end_date"].unique()
```

```
2953
array([                          'NaT', '2019-03-31T00:00:00.000000000'],
      dtype='datetime64[ns]')
```

최근 고객 데이터 집계

회원 구분에 따른 최근 고객 수

- 전체를 집계했을 때와 비율이 크게 다르지 않다.
- 회원 비율 변화에 회원 구분이 중요한 영향을 미치지는 않는다는 것을 알 수 있다.

```
customer_newer.groupby("class_name").count()["customer_id"]
```

```
class_name
0_종일     1444
1_주간      696
2_야간      813
Name: customer_id, dtype: int64
```

최근 고객 데이터 집계

캠페인 구분에 따른 최근 고객 수

- 전체를 집계했을 때와 비율이 다르다. 일반 유저를 보면 72%:81%
- 입회 캠페인은 회원 비율 변화에 영향을 미친다고 추측할 수 있다.

```
customer_newer.groupby("campaign_name").count()["customer_id"]
```

```
campaign_name
0_입회비반액할인      311
1_입회비무료        242
2_일반           2400
Name: customer_id, dtype: int64
```

최근 고객 데이터 집계

성별에 따른 최근 고객 수

- 전체를 집계했을 때와 비율이 크게 다르지 않다.
- 회원 비율 변화에 성별이 중요한 영향을 미치지는 않는다는 것을 알 수 있다.

```
customer_newer.groupby("gender").count()["customer_id"]
```

```
gender
F    1400
M    1553
Name: customer_id, dtype: int64
```

이용 이력 데이터 집계

월/고객 이용 횟수 집계

- 고객 AS002855는 2018년 4월에 4번 이용했다는 것을 알 수 있다

```
uselog["usedate"] = pd.to_datetime(uselog["usedate"])
uselog["연월"] = uselog["usedate"].dt.strftime("%Y%m")
uselog_months = uselog.groupby(["연월","customer_id"],as_index=False).count()
uselog_months.rename(columns={"log_id":"count"}, inplace=True)
del uselog_months["usedate"]
uselog_months.head()
```

	연월	customer_id	count
0	201804	AS002855	4
1	201804	AS009013	2
2	201804	AS009373	3
3	201804	AS015315	6
4	201804	AS015739	7

이용 이력 데이터 집계

고객별 평균값, 중앙값, 최댓값, 최솟값 집계

- 고객 AS002855는 월평균 4.5회 이용했고, 가장 많이 이용한 달은 7번, 가장 적게 이용한 달은 2번 이용했다는 것을 알 수 있다. 중앙값은 5회이다.

```
uselog_customer = uselog_months.groupby("customer_id").agg(["mean", "median", "max", "min" ])["count"]

uselog_customer = uselog_customer.reset_index(drop=False)

uselog_customer.head()
```

	customer_id	mean	median	max	min
0	AS002855	4.500000	5.0	7	2
1	AS008805	4.000000	4.0	8	1
2	AS009013	2.000000	2.0	2	2
3	AS009373	5.083333	5.0	7	3
4	AS015233	7.545455	7.0	11	4

이용 이력 데이터로부터 정기 이용 여부 확인

정기적으로 이용한 고객 특정하기

- 매주 같은 요일에 왔는지 여부로 판단하기로 하자.
- 고객별로 월/요일별로 집계하고, 최댓값이 4 이상인 요일이 하나라도 있으면 정기이용자라고 보자.

```
uselog["weekday"] = uselog["usedate"].dt.weekday
uselog_weekday = uselog.groupby(["customer_id","연월","weekday"],
                 as_index=False).count()[["customer_id","연월", "weekday","log_id"]]
uselog_weekday.rename(columns={"log_id":"count"}, inplace=True)
uselog_weekday.head()
```

	customer_id	연월	weekday	count
0	AS002855	201804	5	4
1	AS002855	201805	2	1
2	AS002855	201805	5	4
3	AS002855	201806	5	5
4	AS002855	201807	1	1

이용 이력 데이터로부터 정기 이용 여부 확인

정기적으로 이용한 고객 특정하기

- 고객별로 최댓값을 계산하고, 그 최댓값이 4 이상인 경우에 routine_flg를 1로 지정하자.

```
uselog_weekday = uselog_weekday.groupby("customer_id",as_index=False).max
()[["customer_id", "count"]]

uselog_weekday["routine_flg"] = 0

uselog_weekday["routine_flg"]
      = uselog_weekday["routine_flg"].where(uselog_weekday["count"]<4, 1)

uselog_weekday.head()
```

	customer_id	count	routine_flg
0	AS002855	5	1
1	AS008805	4	1
2	AS009013	2	0
3	AS009373	5	1
4	AS015233	5	1

고객 데이터와 이용 이력 데이터 결합

데이터 결합

- 앞에서 작성한 uselog_customer와 uselog_weekday를 customer_join과 결합한다.
- 결합에 사용한 조인키는 customer_id이다.
- uselog_weekday에서는 결합 데이터를 customer_id와 routine_flg로만 한정했다.

```
customer_join = pd.merge(customer_join,
                         uselog_customer, on="customer_id", how="left")

customer_join = pd.merge(customer_join,
  uselog_weekday[["customer_id", "routine_flg"]], on="customer_id", how="left")

customer_join.head()
```

	customer_id	name	class	gender	start_date	end_date	campaign_id	is_deleted	class_name	price	campaign_name	mean	median	max	min	routine_flg
0	OA832399	XXXX	C01	F	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.833333	5.0	8	2	1
1	PL270116	XXXXX	C01	M	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	5.083333	5.0	7	3	1
2	OA974876	XXXXX	C01	M	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.583333	5.0	6	3	1
3	HD024127	XXXXX	C01	F	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.833333	4.5	7	2	1
4	HD661448	XXXXX	C03	F	2015-05-01	NaT	CA1	0	2_야간	6000	2_일반	3.916667	4.0	6	1	1

고객 데이터와 이용 이력 데이터 결합

결측치 확인

- 조인 결합 시 결측치가 생길 수 있으므로 확인을 한다.

```
customer_join.isnull().sum()
```

```
customer_id        0
name               0
class              0
gender             0
start_date         0
end_date        2842
campaign_id        0
is_deleted         0
class_name         0
price              0
campaign_name      0
mean               0
median             0
max                0
min                0
routine_flg        0
dtype: int64
```

회원 기간 계산

회원 기간 컬럼 추가

- 회원 기간은 start_date와 end_date의 차이로 한다.
- 아직 탈퇴하지 않아 end_date값이 없는 경우 2019년 4월 30일로 채워서 계산한다.

```
from dateutil.relativedelta import relativedelta
customer_join["calc_date"] = customer_join["end_date"]
customer_join["calc_date"] = customer_join["calc_date"].fillna(pd.to_datetime("20190430"))
customer_join["membership_period"] = 0
for i in range(len(customer_join)):
    delta = relativedelta(customer_join["calc_date"].iloc[i], customer_join["start_date"].iloc[i])
    customer_join["membership_period"].iloc[i] = delta.years*12 + delta.months
customer_join.head()
```

	customer_id	name	class	gender	start_date	end_date	campaign_id	is_deleted	class_name	price	campaign_name	mean	median	max	min	routine_flg	calc_date	membership_period
0	OA832399	XXXX	C01	F	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.833333	5.0	8	2	1	2019-04-30	47
1	PL270116	XXXXX	C01	M	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	5.083333	5.0	7	3	1	2019-04-30	47
2	OA974876	XXXXX	C01	M	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.583333	5.0	6	3	1	2019-04-30	47
3	HD024127	XXXXX	C01	F	2015-05-01	NaT	CA1	0	0_종일	10500	2_일반	4.833333	4.5	7	2	1	2019-04-30	47
4	HD661448	XXXXX	C03	F	2015-05-01	NaT	CA1	0	2_야간	6000	2_일반	3.916667	4.0	6	1	1	2019-04-30	47

고객의 각종 통계량 파악

고객 행동의 통계량 파악

- 컬럼의 mean은 한 고객 당 매월 평균 이용 횟수
- 행의 mean은 고객들의 매월 평균 이용 횟수의 평균

```
customer_join[["mean", "median", "max", "min"]].describe()
```

	mean	median	max	min
count	4192.000000	4192.000000	4192.000000	4192.000000
mean	5.333127	5.250596	7.823950	3.041269
std	1.777533	1.874874	2.168959	1.951565
min	1.000000	1.000000	1.000000	1.000000
25%	4.250000	4.000000	7.000000	2.000000
50%	5.000000	5.000000	8.000000	3.000000
75%	6.416667	6.500000	9.000000	4.000000
max	12.000000	12.000000	14.000000	12.000000

고객의 각종 통계량 파악

정기적으로 이용하는 회원 수 집계

- 정기적으로 이용하는 회원이 3,413명으로 그렇지 않은 회원(779명) 보다 많다는 것을 알 수 있다.

```
customer_join.groupby("routine_flg").count()["customer_id"]
```

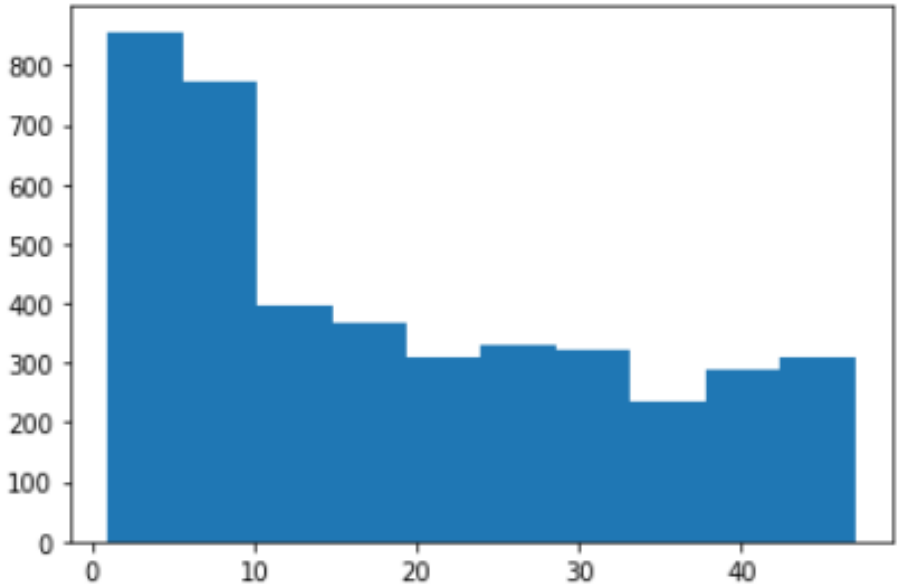
```
routine_flg
0     779
1    3413
Name: customer_id, dtype: int64
```

고객의 각종 통계량 파악

회원 기간의 분포

- 가로축은 회원 기간이므로 회원 기간이 10개월 이내인 고객이 많고 그 이상 되는 회원은 고르게 분포
- 회원 가입 후 짧은 기간 안에 고객이 빠져나간다는 것을 알 수 있다.

```
import matplotlib.pyplot as plt
%matplotlib inline
plt.hist(customer_join["membership_period"])
```



탈퇴 회원과 지속 회원의 차이 파악

지속 회원 통계

```
customer_stay = customer_join.loc[customer_join["is_deleted"]==0]
customer_stay.describe()
```

	is_deleted	price	mean	median	max	min	routine_flg	membership_period
count	2842.0	2842.000000	2842.000000	2842.000000	2842.000000	2842.000000	2842.000000	2842.000000
mean	0.0	8542.927516	6.030288	6.024279	8.471147	3.620690	0.984166	23.970443
std	0.0	1977.189779	1.553587	1.599765	1.571048	2.030488	0.124855	13.746761
min	0.0	6000.000000	3.166667	3.000000	5.000000	1.000000	0.000000	1.000000
25%	0.0	6000.000000	4.833333	5.000000	7.000000	2.000000	1.000000	12.000000
50%	0.0	7500.000000	5.583333	5.500000	8.000000	3.000000	1.000000	24.000000
75%	0.0	10500.000000	7.178030	7.000000	10.000000	5.000000	1.000000	35.000000
max	0.0	10500.000000	12.000000	12.000000	14.000000	12.000000	1.000000	47.000000

탈퇴 회원의 특징

탈퇴 회원 통계

- 탈퇴 회원의 매월 이용 횟수의 평균값, 중앙값, 최댓값, 최솟값이 모두 지속 회원 보다 작다.
- 특히 평균값과 중앙값은 2/3 정도이고 routine_flg를 보면 회원 절반 정도는 센터를 랜덤하게 이용한다.

```
customer_end = customer_join.loc[customer_join["is_deleted"]==1]
customer_end.describe()
```

	is_deleted	price	mean	median	max	min	routine_flg	membership_period
count	1350.0	1350.000000	1350.000000	1350.000000	1350.000000	1350.000000	1350.000000	1350.000000
mean	1.0	8595.555556	3.865474	3.621852	6.461481	1.821481	0.456296	8.026667
std	0.0	1949.163652	1.246385	1.270847	2.584021	0.976361	0.498271	5.033692
min	1.0	6000.000000	1.000000	1.000000	1.000000	1.000000	0.000000	1.000000
25%	1.0	6000.000000	3.000000	3.000000	4.000000	1.000000	0.000000	4.000000
50%	1.0	7500.000000	4.000000	4.000000	7.000000	2.000000	0.000000	7.000000
75%	1.0	10500.000000	4.666667	4.500000	8.000000	2.000000	1.000000	11.000000
max	1.0	10500.000000	9.000000	9.000000	13.000000	8.000000	1.000000	23.000000

데이터 덤프

데이터 덤프

- 데이터를 `customer_join.csv` 파일로 덤프합니다.

```
customer_join.to_csv("customer_join.csv", index=False)
```