회원 탈퇴를 예측하는 테크닉

파이썬을 이용한 실무 데이터 분석

데이터 가공

분석의 전제조건

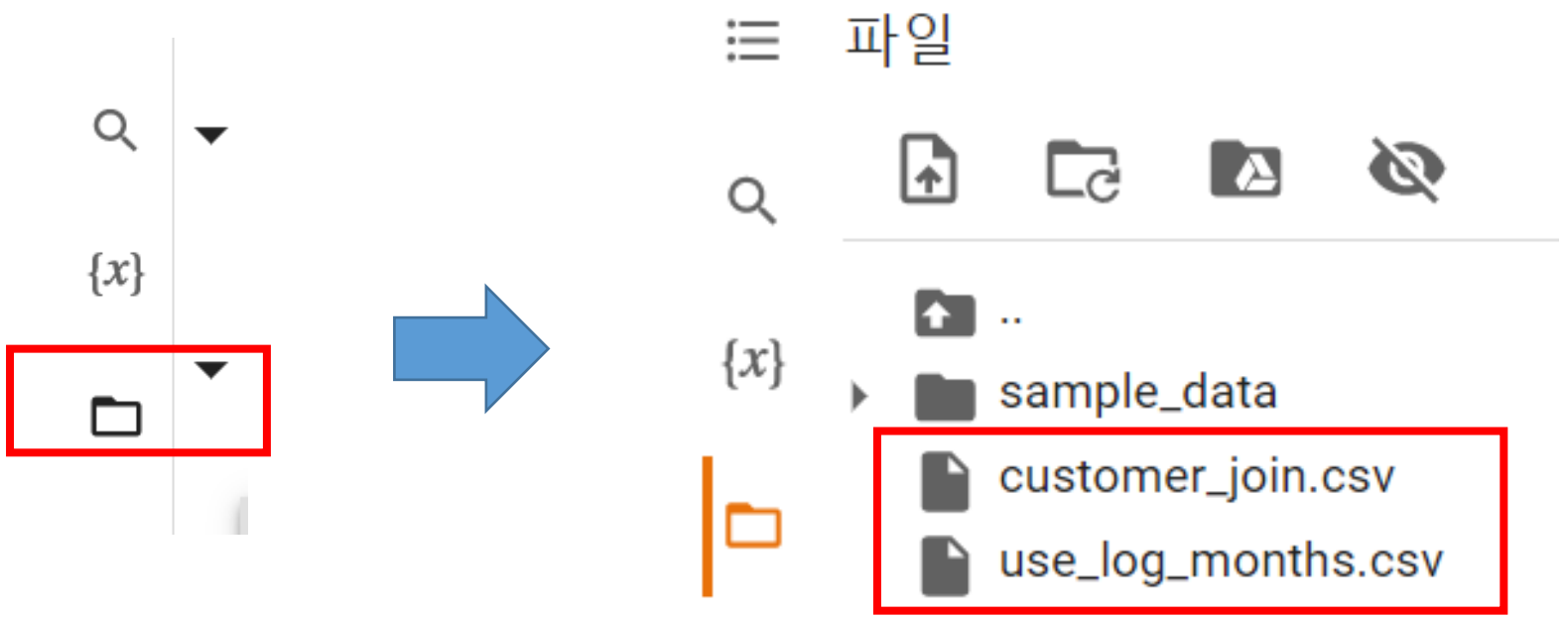
- 의사결정트리(Decision Tree) 알고리즘을 이용해 탈퇴할 만한 고객을 예측합니다.
- 이 결과를 이용해 탈퇴 방지를 위한 정책을 마련할 수 있습니다.

	파일 이름	개요
	use_log.csv	센터의 이용 이력 데이터, 기간은 2018년 4월~2019년 3월
	customer_master.csv	2019년 3월 말 시점의 회원 데이터
	class_master.csv	회원 구분 데이터(종일, 주간, 야간)
	campaign_master.csv	행사 구분 데이터(입회비 유무 등)
1	customer_join.csv	앞에서 작성한 이용 이력을 포함한 고객 데이터
2	use_log_months.csv	앞에서 작성한 이용 이력을 연월/고객별로 집계한 데이터

데이터 불러오기

데이터 업로드

- 데이터 파일을 구글 드라이브 임시 공간에 업로드



데이터 불러오기

데이터 불러오기

- pandas를 이용하여 customer_join.csv, use_log_months.csv 파일을 데이터 프레임으로 불러온다.

```
import pandas as pd

customer = pd.read_csv('customer_join.csv')

uselog_months = pd.read_csv('use_log_months.csv')
```

데이터 가공

미래를 예측하기 위한 데이터 가공

- 1개월 전의 이용 이력으로 다음달 탈퇴를 예측하려고 한다.
- 그 달과 1개월 전의 이용 이력만으로 데이터를 작성한다.

```
year_months = list(uselog_months["연월"].unique())
uselog = pd.DataFrame()
for i in range(1, len(year_months)):
    tmp = uselog_months.loc[uselog_months["연월"]==year_months[i]]
    tmp.rename(columns={"count":"count_0"}, inplace=True)
    tmp_before = uselog_months.loc[uselog_months["연월"]==year_months[i-1]]
    del tmp_before["연월"]
    tmp_before.rename(columns={"count":"count_1"}, inplace=True)
    tmp = pd.merge(tmp, tmp_before, on="customer_id", how="left")
    uselog = pd.concat([uselog, tmp], ignore_index=True)
uselog.head()
```

	연월	customer_id	count_0	count_1
0	201805	AS002855	5	4.0
1	201805	AS009373	4	3.0
2	201805	AS015233	7	NaN
3	201805	AS015315	3	6.0
4	201805	AS015739	5	7.0

탈퇴 전월의 탈퇴고객 데이터 작성

탈퇴 전월의 데이터를 작성하는 이유

- 이 스포츠 센터에서는 월말까지 탈퇴 신청을 해야 다음 달 말에 탈퇴할 수가 있다.
- 2018년 9월 30일에 탈퇴한 회원은 8월에 탈퇴 신청을 한 것이기 때문에 9월 데이터를 사용하면 탈퇴를 미연에 방지할 수가 없다.
- 그러므로 탈퇴 월을 2018년 8월로 하고, 그 1개월 전인 7월의 데이터로부터 8월에 탈퇴 신청을 할 확률을 예측해야 한다.

2018년 8월	2018년 9월	2018년 10월
탈퇴 신청 탈퇴 전월	탈퇴 신청 완료 탈퇴됨	탈퇴

탈퇴 전월의 탈퇴고객 데이터 작성

```
from dateutil.relativedelta import relativedelta
exit_customer = customer.loc[customer["is_deleted"]==1]
exit_customer["exit_date"] = None
exit_customer["end_date"] = pd.to_datetime(exit_customer["end_date"])
for i in range(len(exit_customer)):
    exit_customer["exit_date"].iloc[i] = exit_customer["end_date"].iloc[i] - relativedelta(months=1)
exit_customer["연월"] = pd.to_datetime(exit_customer["exit_date"]).dt.strftime("%Y%m")
uselog["연월"] = uselog["연월"].astype(str)
exit_uselog = pd.merge(uselog, exit_customer, on=["customer_id", "연월"], how="left")
print(len(uselog))
exit_uselog.head()
```

33851

	연월	customer_id	count_0	count_1	name	class	gender	start_date	end_date	campaign_id	...	price	campaign_name	mean	median	max	min	routine_flg	calc_date	membership_period	exit_date
0	201805	AS002855	5	4.0	NaN	NaN	NaN	NaN	NaT	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	201805	AS009373	4	3.0	NaN	NaN	NaN	NaN	NaT	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	201805	AS015233	7	NaN	NaN	NaN	NaN	NaN	NaT	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	201805	AS015315	3	6.0	NaN	NaN	NaN	NaN	NaT	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	201805	AS015739	5	7.0	NaN	NaN	NaN	NaN	NaT	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 22 columns

탈퇴 전월의 탈퇴고객 데이터 작성

결측치 처리

- 결합한 데이터는 탈퇴한 회원의 탈퇴 전월의 데이터 뿐이므로 결측치가 많다. 결측치를 제거한다.

```
exit_uselog = exit_uselog.dropna(subset=["name"])
print(len(exit_uselog))
print(len(exit_uselog["customer_id"].unique()))
exit_uselog.head()
```

```
1104
1104
```

	연월	customer_id	count_0	count_1	name	class	gender	start_date	end_date	campaign_id	...	price	campaign_name	mean	median	max	min	routine_flg	calc_date	membership_period	exit_date
19	201805	AS055680	3	3.0	XXXXX	C01	M	2018-03-01	2018-06-30	CA1	...	10500.0	일반	3.000000	3.0	3.0	3.0	0.0	2018-06-30	3.0	2018-05-30 00:00:00
57	201805	AS169823	2	3.0	XX	C01	M	2017-11-01	2018-06-30	CA1	...	10500.0	일반	3.000000	3.0	4.0	2.0	1.0	2018-06-30	7.0	2018-05-30 00:00:00
110	201805	AS305860	5	3.0	XXXX	C01	M	2017-06-01	2018-06-30	CA1	...	10500.0	일반	3.333333	3.0	5.0	2.0	0.0	2018-06-30	12.0	2018-05-30 00:00:00
128	201805	AS363699	5	3.0	XXXXX	C01	M	2018-02-01	2018-06-30	CA1	...	10500.0	일반	3.333333	3.0	5.0	2.0	0.0	2018-06-30	4.0	2018-05-30 00:00:00
147	201805	AS417696	1	4.0	XX	C03	F	2017-09-01	2018-06-30	CA1	...	6000.0	일반	2.000000	1.0	4.0	1.0	0.0	2018-06-30	9.0	2018-05-30 00:00:00

5 rows × 22 columns

지속 회원 데이터

지속 회원 데이터 작성

- 지속 회원은 탈퇴 월이 없기 때문에 어떤 연월의 데이터로 작성해도 된다.
- 지속 회원을 추출한 후 uselog 데이터에 customer_id로 결합한다.
- name 컬럼의 결측 데이터 제거하고 탈퇴 회원을 제거한다.

```
conti_customer = customer.loc[customer["is_deleted"]==0]

conti_uselog = pd.merge(uselog, conti_customer, on=["customer_id"], how="
left")

print(len(conti_uselog))

conti_uselog = conti_uselog.dropna(subset=["name"])

print(len(conti_uselog))
```

```
33851
27422
```

지속 회원 데이터

데이터 샘플링

- 탈퇴 데이터 1,104개, 지속 데이터 27,422개이므로 불균형한 데이터가 될 수 있어 샘플 수를 조정한다.
- 간단하게 지속 회원 데이터도 회원당 1개가 되게 언더샘플링 한다.

```
conti_uselog = conti_uselog.sample(frac=1).reset_index(drop=True)
conti_uselog = conti_uselog.drop_duplicates(subset="customer_id")
print(len(conti_uselog))
conti_uselog.head()
```

2842

	연월	customer_id	count_0	count_1	name	class	gender	start_date	end_date	campaign_id	...	class_name	price	campaign_name	mean	median	max	min	routine_flg	calc_date	membership_period
0	201805	PL617565	5	8.0	XXXX	C01	M	2016-11-01	NaN	CA1	...	종일	10500.0	일반	5.000000	5.0	8.0	3.0	1.0	2019-04-30	29.0
1	201812	TS357226	8	9.0	XXXXX	C01	M	2018-11-14	NaN	CA1	...	종일	10500.0	일반	8.200000	8.0	9.0	8.0	1.0	2019-04-30	5.0
2	201812	IK003033	3	6.0	XX	C01	F	2017-04-01	NaN	CA1	...	종일	10500.0	일반	5.333333	5.5	8.0	3.0	1.0	2019-04-30	24.0
3	201807	GD323923	5	5.0	XX	C01	M	2015-10-01	NaN	CA1	...	종일	10500.0	일반	4.583333	5.0	7.0	2.0	1.0	2019-04-30	42.0
4	201805	HD734433	3	6.0	XXXXX	C02	M	2016-01-01	NaN	CA1	...	주간	7500.0	일반	4.333333	4.0	8.0	2.0	1.0	2019-04-30	39.0

5 rows × 21 columns

지속 회원 데이터

지속 회원 데이터와 탈퇴 회원 데이터 결합

- 탈퇴 회원 데이터 1,104개와 지속 고객 데이터 2,842개가 결합되어 3,946개가 되었다.

```
predict_data = pd.concat([conti_uselog, exit_uselog],ignore_index=True)
print(len(predict_data))
predict_data.head()
```

3946

	연월	customer_id	count_0	count_1	name	class	gender	start_date	end_date	campaign_id	...	price	campaign_name	mean	median	max	min	routine_flg	calc_date	membership_period	exit_date
0	201805	PL617565	5	8.0	XXXX	C01	M	2016-11-01	NaN	CA1	...	10500.0	일반	5.000000	5.0	8.0	3.0	1.0	2019-04-30	29.0	NaN
1	201812	TS357226	8	9.0	XXXXX	C01	M	2018-11-14	NaN	CA1	...	10500.0	일반	8.200000	8.0	9.0	8.0	1.0	2019-04-30	5.0	NaN
2	201812	IK003033	3	6.0	XX	C01	F	2017-04-01	NaN	CA1	...	10500.0	일반	5.333333	5.5	8.0	3.0	1.0	2019-04-30	24.0	NaN
3	201807	GD323923	5	5.0	XX	C01	M	2015-10-01	NaN	CA1	...	10500.0	일반	4.583333	5.0	7.0	2.0	1.0	2019-04-30	42.0	NaN
4	201805	HD734433	3	6.0	XXXXX	C02	M	2016-01-01	NaN	CA1	...	7500.0	일반	4.333333	4.0	8.0	2.0	1.0	2019-04-30	39.0	NaN

5 rows × 22 columns

예측할 달의 재적 기간

재적 기간 컬럼 추가

- 시간적 요소가 들어간 데이터이므로 재적 기간을 이용하는 것은 좋은 접근이다.

```
predict_data["period"] = 0
predict_data["now_date"] = pd.to_datetime(predict_data["연월"], format="%Y%m")
predict_data["start_date"] = pd.to_datetime(predict_data["start_date"])
for i in range(len(predict_data)):
    delta = relativedelta(predict_data["now_date"][i], predict_data["start_date"][i])
    predict_data["period"][i] = int(delta.years*12 + delta.months)
predict_data.head()
```

	연월	customer_id	count_0	count_1	name	class	gender	start_date	end_date	campaign_id	...	mean	median	max	min	routine_flg	calc_date	membership_period	exit_date	period	now_date
0	201809	PL572723	4	6.0	XX	C03	M	2017-10-01	NaN	CA1	...	6.166667	5.5	10.0	3.0	1.0	2019-04-30	18.0	NaN	11	2018-09-01
1	201808	PL988813	6	6.0	XXXXXX	C01	F	2017-07-01	NaN	CA2	...	6.833333	6.5	9.0	4.0	1.0	2019-04-30	21.0	NaN	13	2018-08-01
2	201811	TS600370	7	3.0	XXX	C01	F	2016-09-01	NaN	CA1	...	5.416667	6.0	7.0	3.0	1.0	2019-04-30	31.0	NaN	26	2018-11-01
3	201809	AS777651	3	4.0	XXXXX	C01	M	2015-05-01	NaN	CA1	...	4.166667	3.5	7.0	2.0	1.0	2019-04-30	47.0	NaN	40	2018-09-01
4	201806	PL746909	8	3.0	XXXXXX	C03	M	2016-11-01	NaN	CA1	...	4.583333	4.5	8.0	2.0	1.0	2019-04-30	29.0	NaN	19	2018-06-01

5 rows × 24 columns

결측치 제거

결측치가 포함된 데이터 제거

- 결측치를 파악한다.
- count_1, end_date, exit_date에 결측치가 있다.

```
predict_data.isna().sum()
```

```
연월                0
customer_id         0
count_0             0
count_1           233
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
calc_date           0
membership_period   0
exit_date        2842
period              0
now_date            0
dtype: int64
```

결측치 제거

결측치가 포함된 데이터 제거

- end_date, exit_date는 탈퇴한 고객만 값을 갖고 있고 지속 회원은 결측치가 된다.
- count_1에 결측치가 있는 데이터만 제거한다.

```
predict_data = predict_data.dropna(subset=["count_1"])
predict_data.isna().sum()
```

```
연월                0
customer_id         0
count_0             0
count_1             0
name                0
class               0
gender              0
start_date          0
end_date         2661
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
calc_date           0
membership_period   0
exit_date        2661
period              0
now_date            0
dtype: int64
```

문자열 변수 처리

분석에 필요한 변수 추출

- 설명변수(Feature) : campaign_name, class_name, gender, count_1, routine_flg, period
- 목적변수(Target) : is_deleted

```
target_col = ["campaign_name", "class_name", "gender", "count_1", "routine_flg", "period", "is_deleted"]

predict_data = predict_data[target_col]

predict_data.head()
```

	campaign_name	class_name	gender	count_1	routine_flg	period	is_deleted
0	일반	종일	M	8.0	1.0	18	0.0
1	일반	종일	M	9.0	1.0	0	0.0
2	일반	종일	F	6.0	1.0	20	0.0
3	일반	종일	M	5.0	1.0	33	0.0
4	일반	주간	M	6.0	1.0	28	0.0

문자열 변수 처리

카테고리형 변수 처리

- 카테고리 변수로는 더미 변수를 만든다.

```
predict_data = pd.get_dummies(predict_data)
predict_data.head()
```

	count_1	routine_flg	period	is_deleted	campaign_name_일반	campaign_name_입회비무료	campaign_name_입회비반액할인	class_name_야간	class_name_종일	class_name_주간	gender_F	gender_M
0	8.0	1.0	18	0.0	1	0	0	0	1	0	0	1
1	9.0	1.0	0	0.0	1	0	0	0	1	0	0	1
2	6.0	1.0	20	0.0	1	0	0	0	1	0	1	0
3	5.0	1.0	33	0.0	1	0	0	0	1	0	0	1
4	6.0	1.0	28	0.0	1	0	0	0	0	1	0	1

문자열 변수 처리

카테고리형 변수 처리

- 생성된 더미 변수 중 필요 없는 항목은 삭제한다.
- c즉, gender_F가 0이면 gender_M이 1이,class_name_주간이 0이면 class_name_야간이 1이라는 의미

```
del predict_data["campaign_name_일반"]
del predict_data["class_name_야간"]
del predict_data["gender_M"]
predict_data.head()
```

	count_1	routine_flg	period	is_deleted	campaign_name_입회비무료	campaign_name_입회비반액할인	class_name_종일	class_name_주간	gender_F
0	8.0	1.0	18	0.0	0	0	1	0	0
1	9.0	1.0	0	0.0	0	0	1	0	0
2	6.0	1.0	20	0.0	0	0	1	0	1
3	5.0	1.0	33	0.0	0	0	1	0	0
4	6.0	1.0	28	0.0	0	0	0	1	0

탈퇴 예측 모델 구축

의사결정나무(Decision Tree) 알고리즘으로 모델 학습 및 예측

```
from sklearn.tree import DecisionTreeClassifier
import sklearn.model_selection

exit = predict_data.loc[predict_data["is_deleted"]==1]
conti = predict_data.loc[predict_data["is_deleted"]==0].sample(len(exit))

X = pd.concat([exit, conti], ignore_index=True)
y = X["is_deleted"]
del X["is_deleted"]
X_train, X_test, y_train, y_test = sklearn.model_selection.train_test_split(X,y)

model = DecisionTreeClassifier(random_state=0)
model.fit(X_train, y_train)
y_test_pred = model.predict(X_test)
print(y_test_pred)
```

탈퇴 예측 모델 구축

예측 결과

- 앞 코드 출력 결과 (y_test_pred)
- 1은 탈퇴, 0은 유지

```
[1. 0. 1. 1. 1. 0. 0. 0. 1. 1. 0. 0. 1. 1. 0. 0. 1. 0. 1. 0. 1. 0. 0. 0.
 1. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 1. 1. 0. 0. 0. 1. 1. 0. 0. 0. 0. 1. 1.
 1. 0. 0. 1. 0. 1. 0. 1. 0. 1. 1. 1. 1. 0. 0. 1. 0. 1. 1. 0. 1. 1. 1. 0.
 1. 1. 0. 1. 1. 1. 0. 1. 1. 0. 0. 1. 1. 1. 0. 1. 0. 1. 0. 1. 0. 1. 1. 0.
 0. 1. 1. 1. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 1. 0. 1. 0.
 1. 1. 1. 0. 1. 0. 1. 0. 1. 1. 0. 1. 1. 1. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0.
 0. 1. 1. 1. 0. 1. 0. 1. 0. 0. 1. 0. 0. 0. 1. 0. 0. 1. 1. 0. 0. 1. 0. 1.
 1. 1. 1. 0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 1.
 0. 0. 1. 1. 1. 1. 0. 0. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 0. 0. 0. 1. 1.
 1. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 1. 1. 0. 0. 0. 1. 0. 1. 0. 1. 0. 1. 1.
 1. 0. 1. 0. 0. 0. 1. 1. 0. 1. 0. 1. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 1. 0.
 1. 1. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 1. 1. 0. 1. 1. 0. 1. 0. 0.
 0. 1. 0. 1. 1. 1. 0. 0. 1. 0. 1. 1. 0. 1. 1. 1. 1. 0. 1. 0. 1. 1. 0. 1.
 1. 1. 1. 0. 0. 0. 1. 1. 0. 0. 1. 0. 1. 0. 1. 0. 0. 1. 0. 1. 0. 0. 1. 1.
 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 1. 1. 0. 0. 0. 1. 1. 0. 0.
 1. 0. 1. 0. 1. 1. 1. 0. 0. 0. 1. 0. 1. 0. 1. 0. 0. 0. 1. 0. 1. 1. 1. 0.
 1. 1. 1. 0. 1. 0. 0. 0. 0. 0. 1. 0. 1. 0. 1. 1. 0. 1. 0. 0. 1. 1. 1. 0.
 0. 1. 1. 1. 0. 1. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 1.
 1. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 1. 0. 0. 0. 1. 1. 0. 1.
 1. 1. 1. 0. 1. 0. 0. 1. 0. 0. 0. 1. 0. 1. 1. 1. 1. 1. 0. 0. 1. 0. 0. 1.
 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 1. 1. 0.
 0. 0. 1. 0. 1. 0. 0. 0. 1. 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 0. 1. 0.]
```

탈퇴 예측 모델 구축

예측 결과를 데이터프레임에 저장

- y_test 값에 대한 예측 결과 y_test_pred를 results_test 데이터프레임에 저장한다.

```
results_test = pd.DataFrame({"y_test":y_test ,"y_pred":y_test_pred })
results_test.head()
```

	y_test	y_pred
197	1.0	1.0
1329	0.0	0.0
695	1.0	1.0
2	1.0	1.0
647	1.0	1.0

예측 모델 평가

모델 평가

- 정확도 계산. (y_test와 y_pred가 일치하는 개수) / (전체 데이터 개수)

```
correct = len(results_test.loc[results_test["y_test"]==results_test["y_pred"]])
data_count = len(results_test)
score_test = correct / data_count
print(score_test)
```

```
0.8821292775665399
```

- 아래 결과를 보았을 때 과적합(overfitting)의 경향이 있다.

```
print(model.score(X_test, y_test))
print(model.score(X_train, y_train))
```

```
0.8821292775665399
0.9784537389100126
```

모델 튜닝

과적합 문제 해결

- 데이터 늘리기, 변수 재검토, 모델 파라미터 변경과 같은 방법을 적용해 볼 수 있다.
- 트리의 깊이를 얕게 하는 모델 파라미터를 변경해 보자 (max_depth=5)

```
X = pd.concat([exit, conti], ignore_index=True)
y = X["is_deleted"]
del X["is_deleted"]
X_train, X_test, y_train, y_test = sklearn.model_selection.train_test_split(X,y)

model = DecisionTreeClassifier(random_state=0, max_depth=5)
model.fit(X_train, y_train)
print(model.score(X_test, y_test))
print(model.score(X_train, y_train))
```

```
0.9144486692015209
0.9233206590621039
```

모델에 기여하는 특성 확인

변수의 중요도 파악

- 변수와 중요도를 데이터프레임에 저장한다.
- 변수의 기여율을 출력하는 함수는 앞과는 다르다. (model.feature_importances)
- 1개월 전의 이용 횟수, 정기 이용 여부, 재적 기간이 기여도가 높다.

```
importance = pd.DataFrame({"feature_names":X.columns,
                           "coefficient":model.feature_importances_})

importance
```

	feature_names	coefficient
0	count_1	0.372231
1	routine_flg	0.128296
2	period	0.497529
3	campaign_name_입회비무료	0.000000
4	campaign_name_입회비반액할인	0.001945
5	class_name_종일	0.000000
6	class_name_주간	0.000000
7	gender_F	0.000000

회원 탈퇴 예측

예측을 위한 데이터 작성(1)

- 임의의 데이터를 만들고 데이터를 가공한다. 카테고리 변수 때문에 조금 복잡할 수 있다.

```
count_1 = 3
routing_flg = 1
period = 10
campaign_name = "입회비무료"
class_name = "종일"
gender = "M"
```

회원 탈퇴 예측

예측을 위한 데이터 작성(2)

- 임의의 데이터를 만들고 데이터를 가공한다. 카테고리 변수 때문에 조금 복잡할 수 있다.

```
if campaign_name == "입회비반값할인":
    campaign_name_list = [1, 0]
elif campaign_name == "입회비무료":
    campaign_name_list = [0, 1]
elif campaign_name == "일반":
    campaign_name_list = [0, 0]
if class_name == "종일":
    class_name_list = [1, 0]
elif class_name == "주간":
    class_name_list = [0, 1]
elif class_name == "야간":
    class_name_list = [0, 0]
if gender == "F":
    gender_list = [1]
elif gender == "M":
    gender_list = [0]
input_data = [count_1, routing_flg, peri
od]
input_data.extend(campaign_name_list)
input_data.extend(class_name_list)
input_data.extend(gender_list)
```

회원 탈퇴 예측

임의의 데이터에 대해 예측 수행

- 0.93의 확률로 탈퇴가 예상된다는 결과가 나왔다.
- 변수 값을 바꿔가며 예측해 보자.

```
print(model.predict([input_data]))
print(model.predict_proba([input_data]))
```

```
[1.]
[[0.06185567 0.93814433]]
```