



통신사 이탈 고객 예측

노상희

목차

1. 문제 정의
2. 데이터 분석(전처리, 시각화)
3. 데이터셋 생성
4. 모델 분석
5. 결론(시사점, 한계점, 발전방향, 느낀점)

문제 정의 ..

- 통신사 고객 유치 경쟁 과열
- 통신사 이탈 고객의 통신 사용 현황 분석 필요

통신 3사 "5G 프리미엄폰 고객 잡아라"

노정연 기자 dana_fm@kyunghyang.com



- 삼성 폴더블폰 사전판매 진행
- SKT, 최대 153만원 할인 제공
- KT는 5G-콘텐츠 무제한으로
- LGU+, 유선 전용 제휴팩 지원

알뜰폰 요금경쟁 치열, 데이터·제휴 혜택에 소비자 방긋

2021.07.04 11:27:47 / 최민지 cmj@ddaily.co.kr

관련기사

- ↳ 쪼그라든 6월 번호이동시장, 알뜰폰만 살았다
- ↳ 고꾸라지던 LTE 가입자 수, 17개월만에 첫 반등 '알뜰폰 덕'
- ↳ 알뜰폰에 힘 주는 LGU+, 최대 150GB 데이터 쏜다



문제 정의 - 데이터 출처(kaggle)


🔄 [kaggle.com/barun2104/telecom-churn](https://www.kaggle.com/barun2104/telecom-churn) ☆

🔍 Search

Dataset




Customer Churn

Predict Customer Churn in a Telecom Industry

 Barun Kumar • updated a year ago (Version 2)

[Data](#) [Tasks](#) [Code \(10\)](#) [Discussion \(2\)](#) [Activity](#) [Metadata](#)

[Download \(126 KB\)](#) [New Notebook](#) ⋮

 Usability 9.4  License Data files © Original Authors  Tags business, tabular data, binary classification

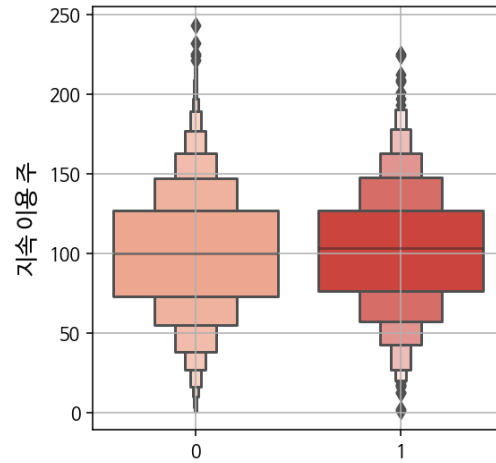
Description

Context

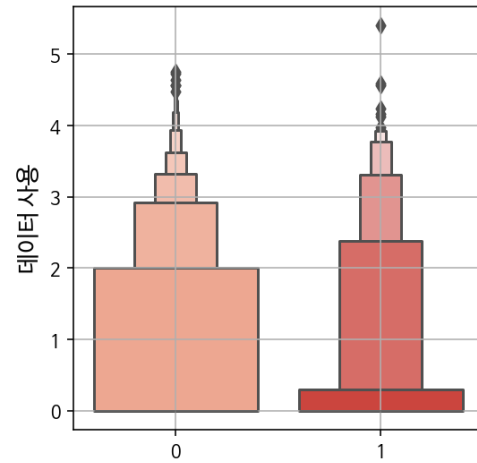
With the rapid development of telecommunication industry, the service providers are inclined more towards expansion of the subscriber base. To meet the need of surviving in the competitive environment, the retention of existing customers has become a huge challenge. It is stated that the cost of acquiring a new customer is far more than that for retaining the existing one. Therefore, it is imperative for the telecom industries to use advanced analytics to understand consumer behavior and in-turn predict the association of the customers as whether or not they will leave the company.

<https://www.kaggle.com/barun2104/telecom-churn>

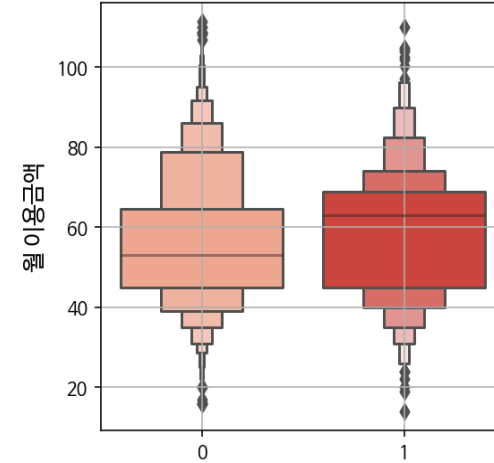
데이터 전처리 - 이상치 유무 체크



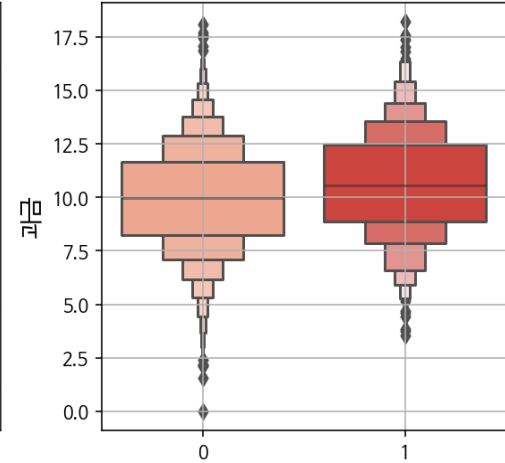
이탈 고객



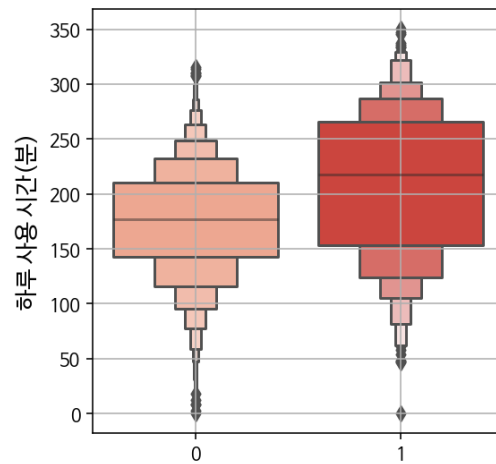
이탈 고객



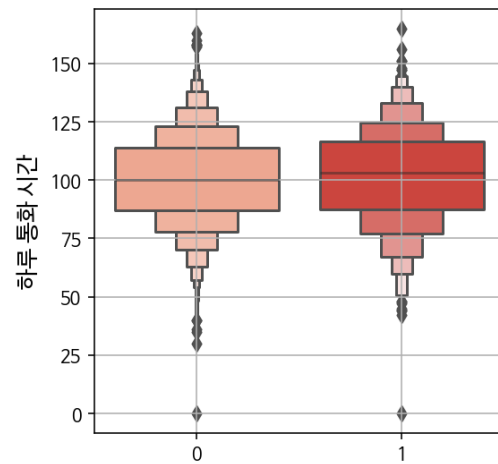
이탈 고객



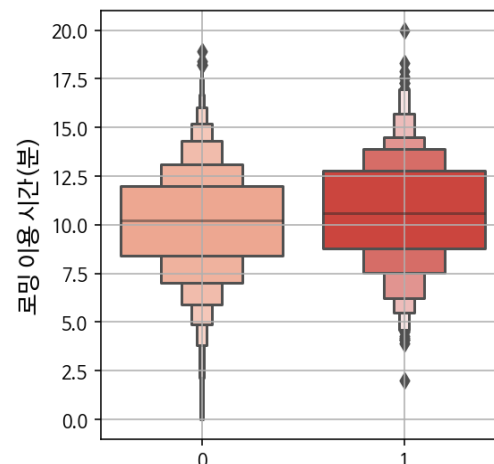
이탈 고객



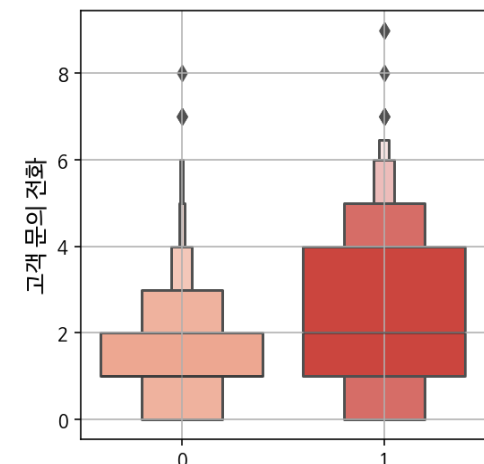
이탈 고객



이탈 고객



이탈 고객



이탈 고객

데이터 전처리 ..

1. feature 모두 사용

```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 3333 entries, 0 to 3332
```

```
Data columns (total 11 columns):
```

#	Column	Non-Null Count	Dtype
0	Churn	3333 non-null	int64
1	AccountWeeks	3333 non-null	int64
2	ContractRenewal	3333 non-null	category
3	DataPlan	3333 non-null	category
4	DataUsage	3333 non-null	float64
5	CustServCalls	3333 non-null	int64
6	DayMins	3333 non-null	float64
7	DayCalls	3333 non-null	int64
8	MonthlyCharge	3333 non-null	float64
9	OverageFee	3333 non-null	float64
10	RoamMins	3333 non-null	float64

```
dtypes: category(2), float64(5), int64(4)
```

```
memory usage: 241.2 KB
```

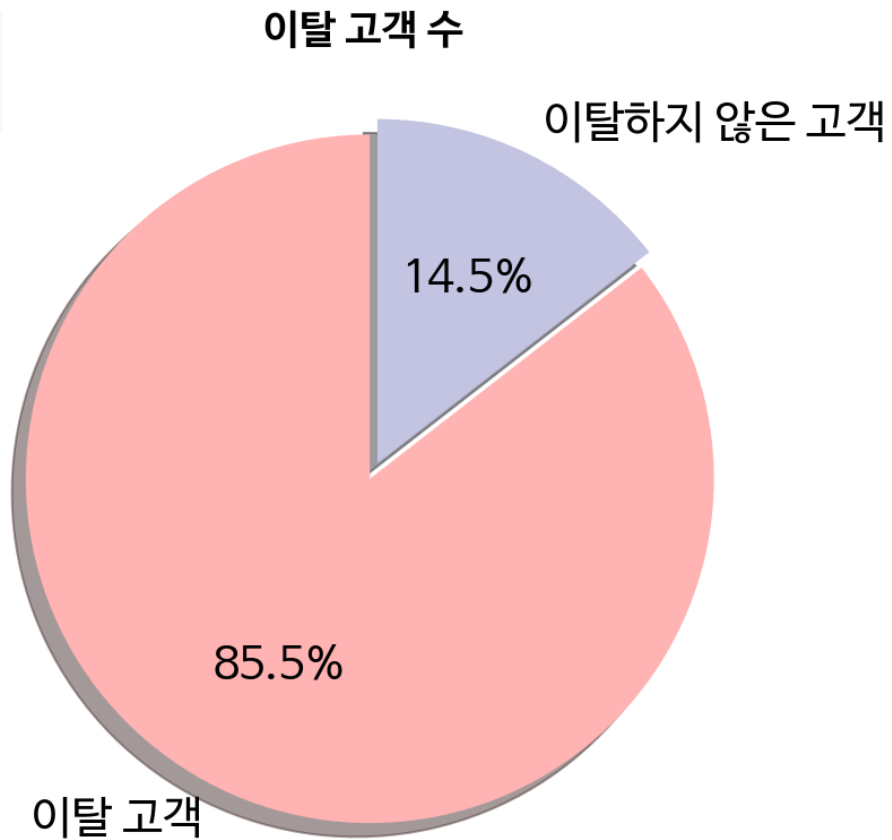
2. Feature 2, 3번

int64 → category로 변경

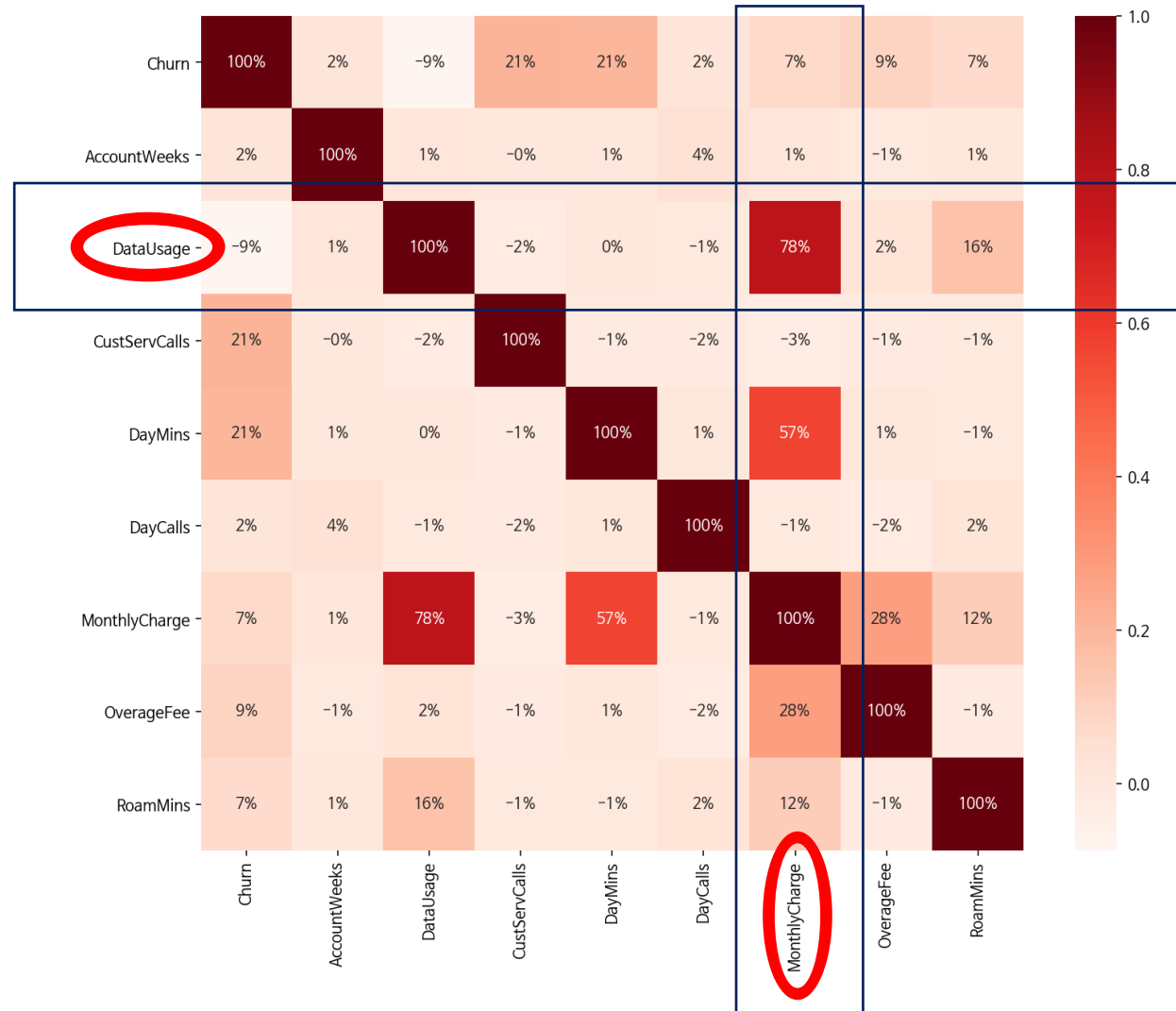
데이터 전처리 - 이상치 유무 체크

```
1 df['Churn'].value_counts()
```

```
0    2850  
1     483  
Name: Churn, dtype: int64
```



데이터 전처리 - 이상치 유무 체크



데이터셋 생성 - 학습데이터, 테스트데이터, 검증데이터 구분

```
1 # 학습 데이터 2333개 변수 가져오기
2 X_train = df.iloc[:2333, 1:11]
3 y_train = df.iloc[:2333, 0]
4
5 print(X_train.shape)
6 print(y_train.shape)
7
8 # 테스트 데이터 1000개 변수 가져오기
9 X_test = df.iloc[2333:, 1:11]
10 y_test = df.iloc[2333:, 0]
11
12 print(X_test.shape)
13 print(y_test.shape)
```

1. 총 3,333개 데이터
→ 학습 데이터 : 테스트 데이터 (7:3) split

2. 학습 데이터 2,333개
→ 훈련데이터 : 검증데이터 (7:3) split

```
1 X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
2 stratify=y_train, # 0과 1 비율을 맞춰서 트레이닝
3 test_size=0.3,
4 random_state=777)
```

```
1 print(f'훈련 데이터 {X_train.shape} 레이블 {y_train.shape}')
2 print(f'검증 데이터 {X_val.shape} 레이블 {y_val.shape}')
```

훈련 데이터 (1633, 10) 레이블 (1633,)
검증 데이터 (700, 10) 레이블 (700,)

예측 모델 - 구성

```
1 model = Sequential()  
2 model.add(Dense(12, activation='relu', input_shape=(10, )))  
3 model.add(Dense(8, activation='relu'))  
4 model.add(Dense(1, activation='sigmoid'))
```

```
1 model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 12)	132
dense_1 (Dense)	(None, 8)	104
dense_2 (Dense)	(None, 1)	9

Total params: 245
Trainable params: 245
Non-trainable params: 0

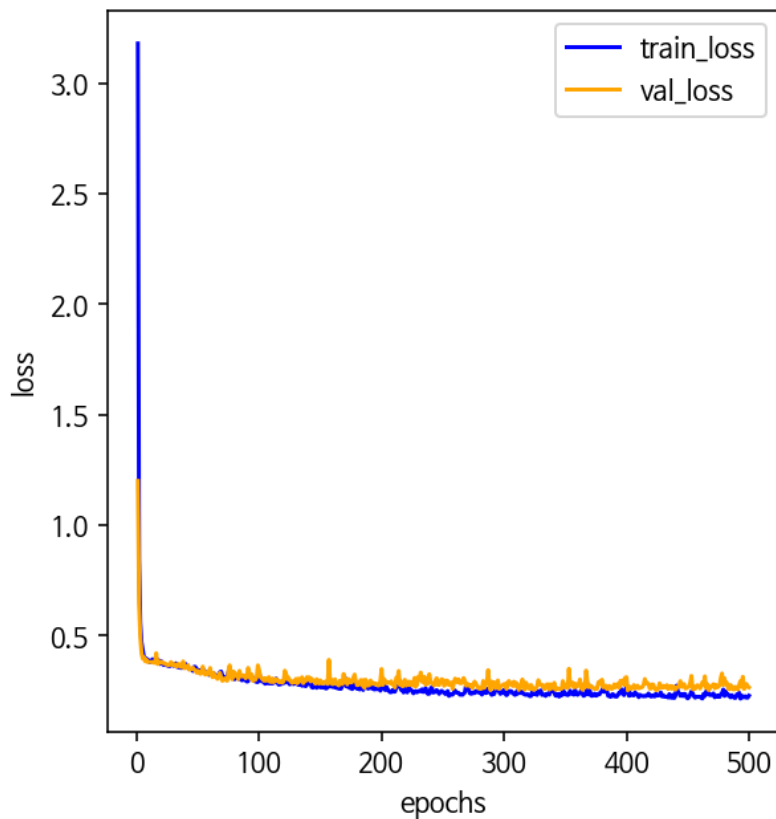
예측 모델 - 학습

```
1 from keras.optimizers import Adam
```

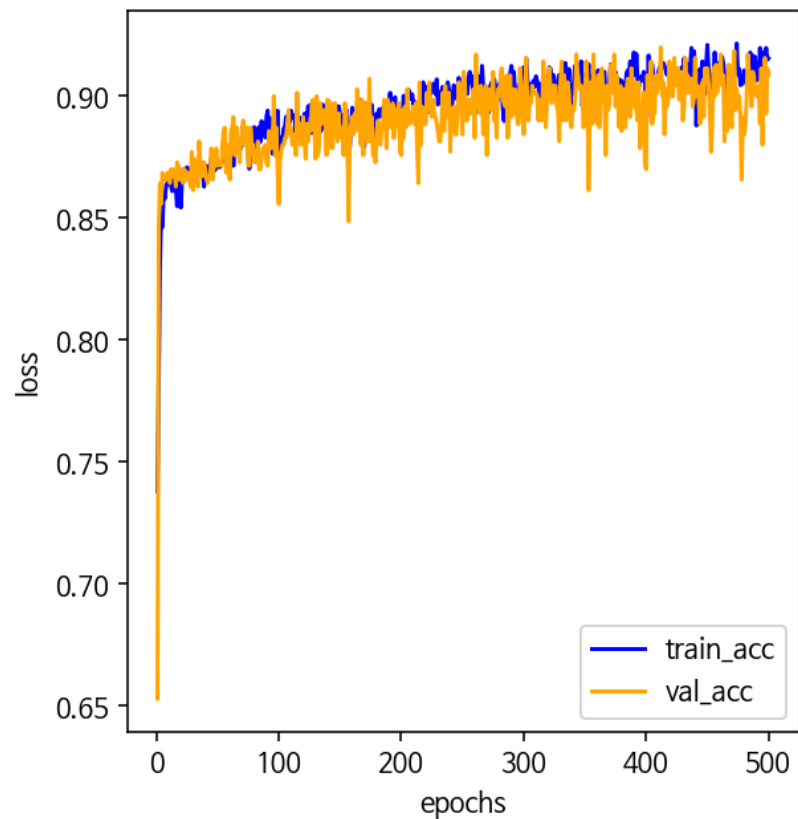
```
1 model.compile(loss='binary_crossentropy',  
2               optimizer=Adam(learning_rate=0.01),  
3               metrics=['acc'])
```

```
history = model.fit(X_train, y_train,  
                    epochs=500,  
                    batch_size=128,  
                    validation_data = (X_val, y_val))
```

train and val loss



train and val acc



예측 모델 - 성능 평가 지표

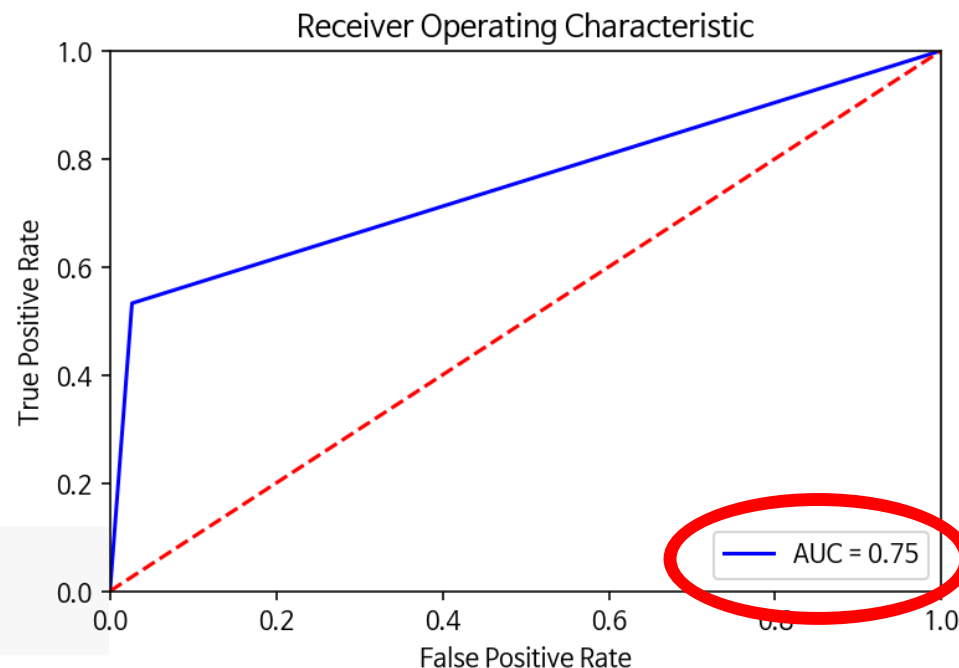
- Precision (False Positive 오류가 적은 지표)

	precision	recall	f1-score	support
0	0.92	0.94	0.93	831
1	0.67	0.59	0.62	169
accuracy			0.88	1000
macro avg	0.79	0.76	0.78	1000
weighted avg	0.88	0.88	0.88	1000

```
1 scores = model.evaluate(X_test, y_test)
2 print("%s: %.2f%%" %(model.metrics_names[1], scores[1]*100))
```

32/32 [=====] - 0s 1ms/step - loss: 0.2760 - acc: 0.8990

acc: 89.90%



결론 - 시사점

- Precision 지표

정확도 89.9%,

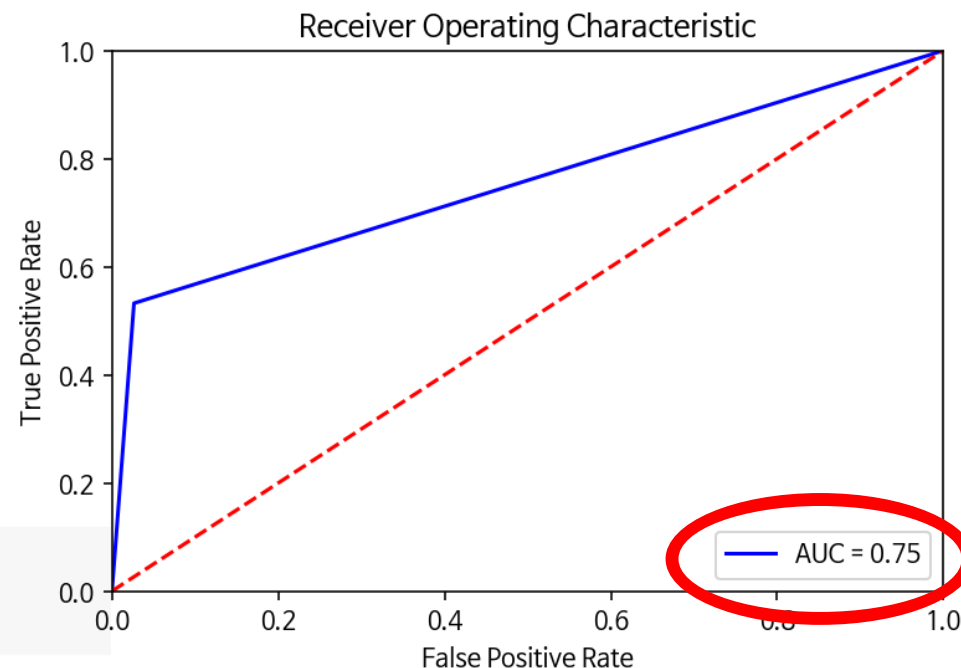
ROC 커브 0.75

	precision	recall	f1-score	support
0	0.92	0.94	0.93	831
1	0.67	0.59	0.62	169
accuracy			0.88	1000
macro avg	0.79	0.76	0.78	1000
weighted avg	0.88	0.88	0.88	1000

```
1 scores = model.evaluate(X_test, y_test)
2 print("%s: %.2f%%" %(model.metrics_names[1], scores[1]*100))
```

32/32 [=====] - 0s 1ms/step - loss: 0.2760 - acc: 0.8990

acc: 89.90%



결론 - 한계점 & 향후 발전 방향

1. False Positive(실제 False를 True로 잘못 예측) 오류가 작다..

	precision	recall	f1-score	support
0	0.92	0.94	0.93	831
1	0.67	0.59	0.62	169
accuracy			0.88	1000
macro avg	0.79	0.76	0.78	1000
weighted avg	0.88	0.88	0.88	1000

```
1 scores = model.evaluate(X_test, y_test)
2 print("%s: %.2f%%" %(model.metrics_names[1], scores[1]*100))
```

```
32/32 [=====] - 0s 1ms/step - loss: 0.2760 - acc: 0.8990
acc: 89.90%
```

참조

- 데이터셋

<https://www.kaggle.com/barun2104/telecom-churn>

- 문제정의(통신사 경쟁 과열)

http://biz.khan.co.kr/khan_art_view.html?artid=202009102148015&code=930201

- 문제정의(통신사 이탈 고객 분석 필요)

<http://www.ddaily.co.kr/news/article/?no=217381>

감사합니다.
