인공지능공학

5조
아베다카노부
하라시마미쓰히로
사카모토타다아키
오리프조노프 울루그벡

목차

주제

회사의 고객 정보와 이용이력 데이터를 이용하여

고객의 계약 상태 예측 모델 제작하고 회사 이익 향상

데이터셋 구성 내용

미국의 휴대전화의 전기통신사에데이터

데이터셋은 2개

- 고객정를 포함한 50커럼 데이터(가족정보, 거주지역, 계약대수, 자동차의 우무, 등.
- 이용이력을 포함한 50커럼의 데이터(한달의 평균 이용시간, 한 달후의 계약상태, 등
  ...)

데이터를 고객번호를 기준으로 합쳐서 사용했음

10만명의 고객의 계약정보가 포함된 데이터셋

일본의 대학서 개최된 Competution에서 가지고 온 데이처셋

시장분석01

Source: IBISWorld

이미지1. 미국의 통신사 수익에 관한 그래프

왼쪽의 그래프는 미국의 무선통신업계의 수익의 합계와 수익성장률입니다. 이 그래프의 수익성장률에서는 아래와 같은 내용을 알 수 있다.

수익 성장률 (적선)

- 변화가 심하고, 급격한 감소, 성장의 폭이 있다.
- 2014년, 2016년, 2020년부터 2022년에 걸쳐 변화율이 크게 하락
- 2024년을 향해 급격한 상승이 보인다.

성장과 감소에 대한 자세한 내용은 다음과 같다.

COVID-19 판데믹에 의한 영향으로 불확실성이나 실업률 상승으로 소비자 지출이 감소한 것에 의한 **신규계약의 저하**가 원인으로 보인다.

2024년 이후의 급격한 상승은 기술혁신에 의한 새로운 가격대의 도입에 의한 상승이라고 생각되고 있다.

이것으로부터 외부적 요인에 크게 좌우되기 쉽다는 것을 알 수 있다.

그래프 인용원 및 참고

https://www.ibisworld.com/united-states/market-research-reports/wireless-telecommunications-carriers-industry/#IndustryStatisticsAndTrends

참考

https://www.analysysmason.com/research/content/short-reports/covid-19-operator-revenue-impact/

https://www2.deloitte.com/content/dam/Deloitte/us/Documents/technology-media-telecommunications/2024-telecom-industry-outlook.pdf#:~:text=URL%3A%20https%3A%2F%2Fwww2.deloitte.com%2Fcontent%2Fdam%2FDeloitte%2Fus%2FDocuments%2Ftechnology

시장분석 02

Source: IBISWorld

이미지1. 미국의 통신사 수익에 관한 그래프

이어서, 이 그래프의 수익의 합계(청색선)로부터는 이하의 내용을 판독할 수 있다.

수익 합계 (청색 선)
수익이 어느 정도 안정적이다.
오랜 기간에 걸쳐 성장해 왔기 때문에 어느 정도의 시장규모가 있다.

- 이미 대부분의 고객을 잡고 있기 때문에 신규 고객의 획득이나 시장 확대가 어려워 포화 상태에 있다.

정리하면, 앞으로 기술 핵신으로 인한 성장이 기대되는 반면 새로운 고객층을 개척하는데에 어려움이 있다.

グラフ引用元、および参考
https://www2.deloitte.com/content/dam/Deloitte/us/Documents/technology-media-telecommunications/2024-telecom-industry-outlook.pdf
https://www.ibisworld.com/united-states/market-research-reports/wireless-telecommunications-carriers-industry/#IndustryStatisticsAndTrends

고객 해약률 및 시장조사

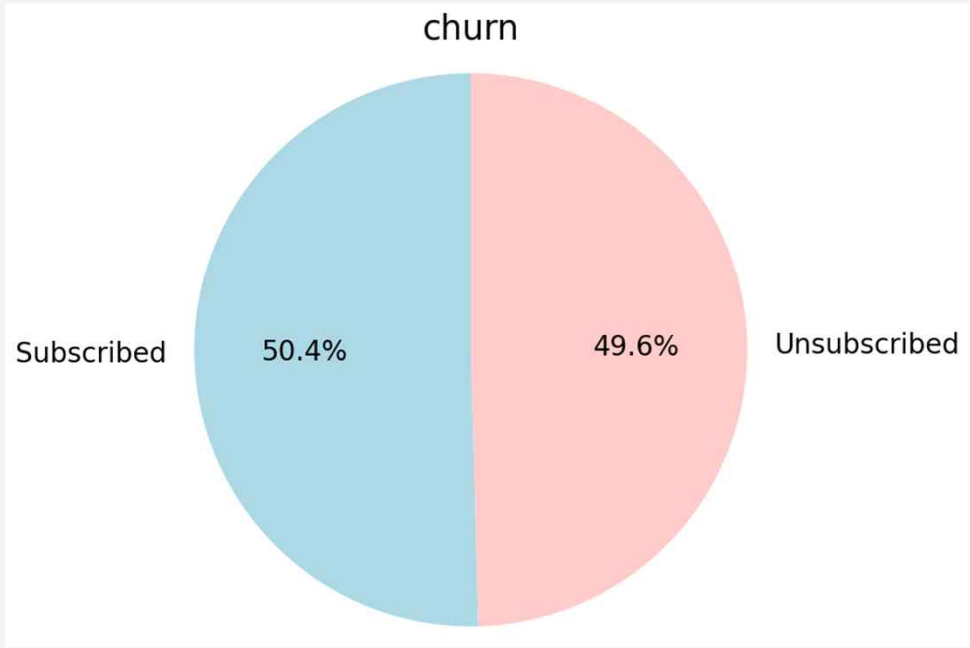


図2 해약 비율

왼쪽의 그림 2는 A사의 해지자의 비율에 관한 그래프이다.
이것을 보면 A사의 해약의 비율은 약 절반이다.
이것은 도 3의 타사의 해약률과 비교하면 높은 경향에 있다
.
또 해약하고 있는 사람들로부터의 월간 수익 비율을 보면
전체의 약 49%(약 2, 870, 795. 53$)를 차지하고 있다.
이상으로부터 이 고객층에 대한 비용효과는 충분히 높다는

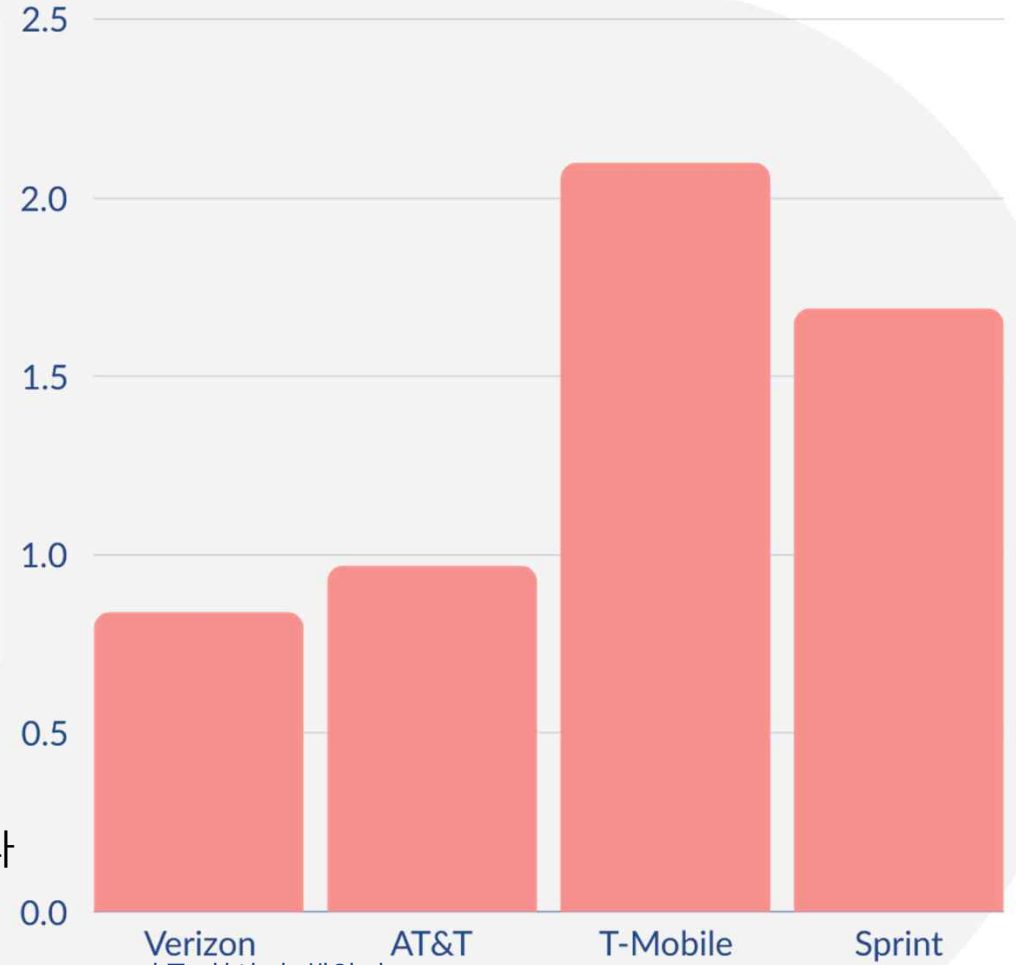


図3다른 회사의 해약비율

データ引用元
https://www.phonearena.com/news/Verizon-reports-lowest-churn-rate-in-the-industry-for-Q2_id33174

고객 해약률 및 시장조사

해약자를 1% 줄일 수 있으면

해약자의 한달의 편균금액 * 고객수 * 1%
= $58.2 * 33만명 * 1%
= $19,206

$19,206/월의 손실을 방지할 수가 있다

-> W26,853,829.20/월

EDA 및 전처리 (Outlier)

데이터의 분산, Outlier, 결손값, 상관성등의 시각화와 이해를 통해 데이터 전처리를 위한 기반을 조성

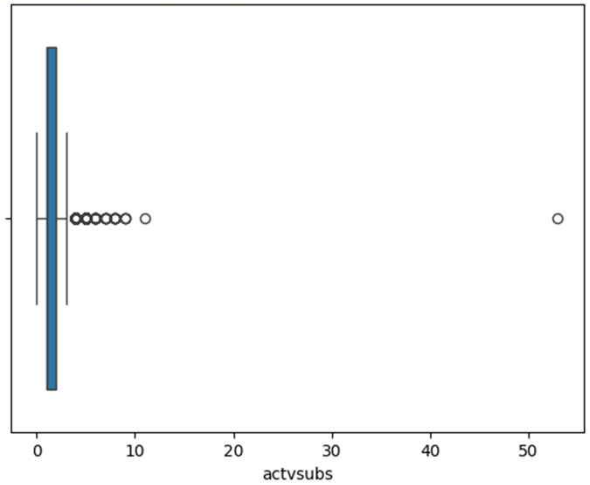
분산과 Outlier의 시각화 와 분석을 위해 Box plot을 이

Box plot의 선택 목적

1. 고객의 이용상황에 대해 극단적으로 많거나 적은 수치의 발견
2. 고객의 개인정보 일반화를 위해 수치적인 이상치 발견

이상치가 있는 컬럼에 대한 처리

가족수와 기 위해 삭제




```python
def tukey(data , c):
    q1 = np.percentile(data[c] , 25)
    q3 = np.percentile(data[c] , 75)

    IQR = q3 - q1

    upper_fence = q3 + 1.5 * IQR
    lower_fence = q1 - 1.5 * IQR

    return data[(data[c] < lower_fence) | (data[c] > upper_fence)]
```

```
totmou Outlier Percentage : 5.99%
totrev Outlier Percentage : 5.68%
avgqty Outlier Percentage : 5.25%
avg3rev Outlier Percentage : 6.16%
avgmou Outlier Percentage : 4.71%
uniqsubs Outlier Percentage : 3.9%
adjqty Outlier Percentage : 7.11%
adjrev Outlier Percentage : 5.65%
actvsubs Outlier Percentage : 1.2%
totcalls Outlier Percentage : 7.11%
avg3mou Outlier Percentage : 5.2%
adjmou Outlier Percentage : 5.98%
avg3qty Outlier Percentage : 5.38%
avgrev Outlier Percentage : 5.31%
```

EDA 및 전처리(결손값)

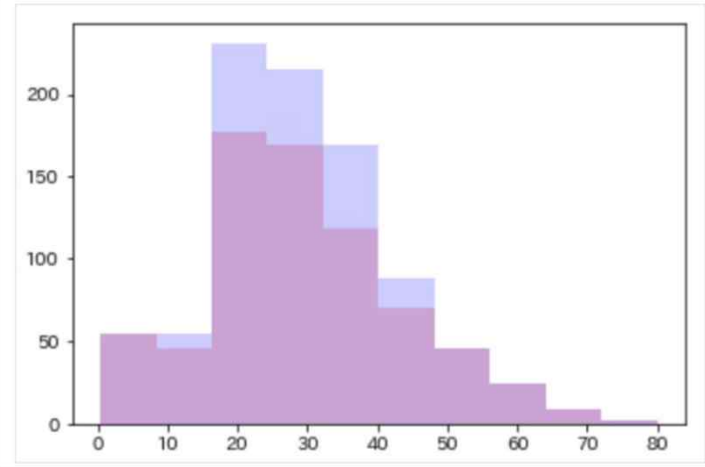
결손값이 1000개(전체의 0.01%)미만이라면 데이터를 삭제 처리

정기분포에 따라 란댐으로 값을 생성하여 결손값에 넣음

KNNImputer를 사용하여 비슷한 데이터를 찾고 그걸 기준으로 값을 넣어줌

전체 데이터 수는 10만개 -> 9만8천266
삭제데이터 수는 1734개 (0.017%)

	0
lor	29634
adults	22579
income	24948
numbcars	48424

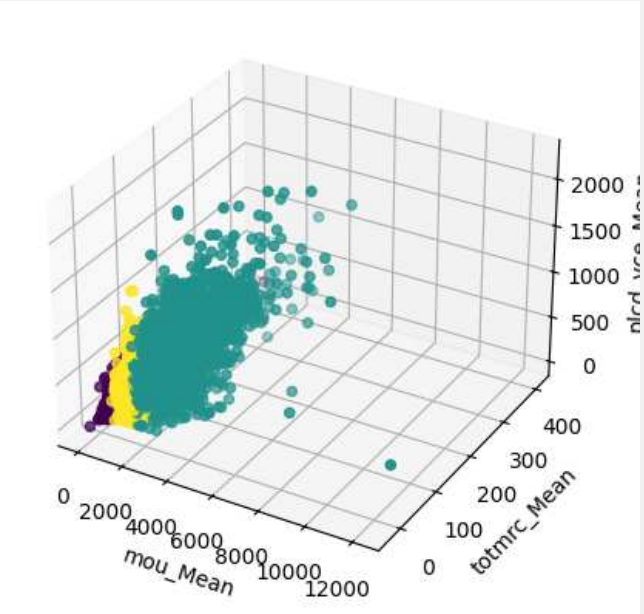


avg6mou	2839
avg6qty	2839
avg6rev	2839
hnd_price	847
phones	1
models	1
truck	1732
rv	1732
lor	30189
adults	23018
income	25435
numbcars	49364
forgntvl	1732
eqpdays	1
rev_Mean	357
mou_Mean	357
totmrc_Mean	357
da_Mean	357
ovrmou_Mean	357
ovrrev_Mean	357
vceovr_Mean	357
datovr_Mean	357
roam_Mean	357
change_mou	891
change_rev	891

EDA 및 전처리 (clustering)

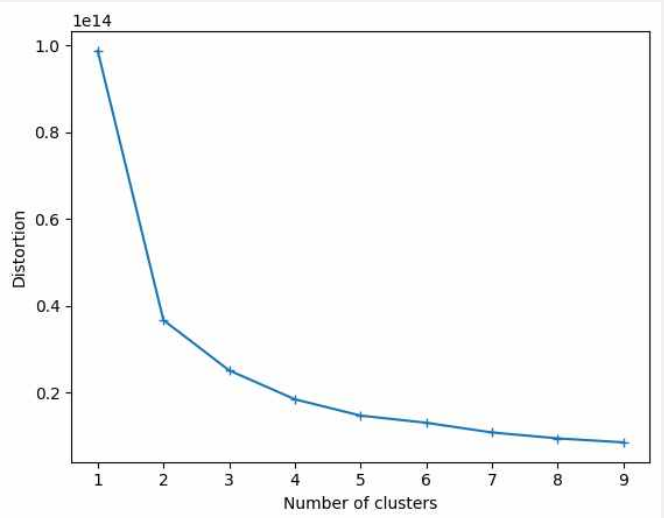
k-means법을 이용하여 사용도별로 3type로 분류
다음의 3속성을 사용하여 클러스터링

mou_Mean (달마다의 편군 사용 시간)
totmrc_Mean (달마다의 편균 요금)
plcd_vce_Mean(달마다의 편균 통화 횟수)



엘보법을 사용하여 클러스터스는 3이 적절하다고 생각했습니다.

엘보법이란 클러스터 수를 변화시키면서 잔차제곱합을 계산하고, 그 결과를 그림으로 나타내어 적절한 클러스터 수를 추정하는 기법이다.



EDA 및 전처리(clustering)

테이블을 보면 모든 속성이 1 > 2 > 0의 순으로 크게 되어 있는 걸 알 수 있다.

클래스터를 각각 light user , heavy user, middle user로 구분하겠다.

cluster별로 해약률을 계산해 보니 0 > 2 > 1 라는 사용도과 반대의 결과가 나왔다.

-> 사용율이 낮은 사람에게 주목하여 해약률을 저하시키면 해약율 1% 줄이기를 달성할 있다

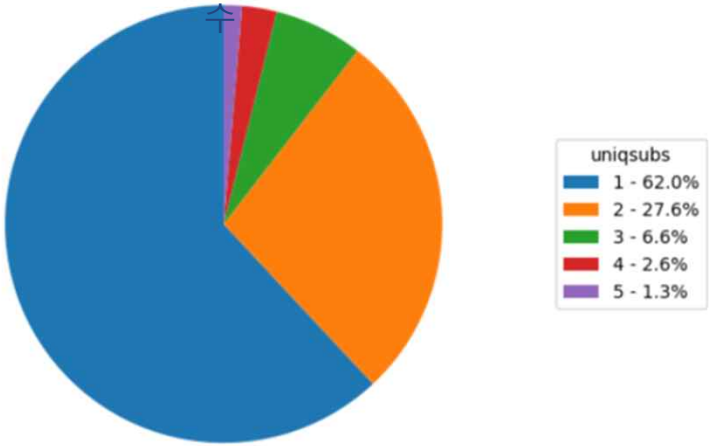
cluster	mou_Mean	hnd_price	avgqty
0	224	38	71
1	1941	80	504
2	851	56	233

cluster	분류
0	light user
1	heavy user
2	middle user

cluster	해약률
0	0.510887
1	0.430986
2	0.470531

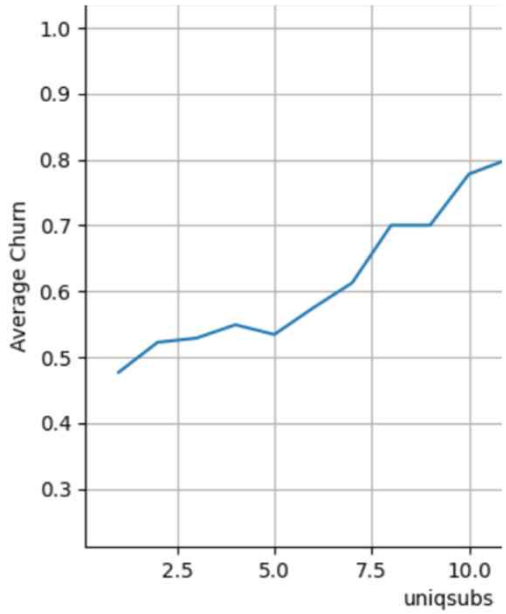
EDA 및 전처리(clustering)

한 가구의 계약자 수



2명 이상의 가족과 계약하고 있는 사람이 38% 있음

계약자 수에 따른 해약률



한 가구에 계약자가 많아질 수록 해약률도 높아짐

2명 이상 계약자가 있는 가구 중 클러스터 별로 해약자 비율

cluster	해약자 비율
light	0.40
middle	0.32
heavy	0.4

light user에 가족이 있는 비율이 많아

light user 중에 가족과 같이 계약하고 있는 사람들에게 주목하여서 가족계약자가 많아질 수록 할인해주는 대우가 필요하다고 생각했다.
또한 기계학습을 통해서 그 특징에 할당하는 분을 찾고 강하게 호소하면 효율적으로 효과를 나타낼 수 있다.

모델 작성(SVC)

서포트 벡터 머신 모델

데이터가 너무 커서 돌리기가 어려웠다
-> 주성분분석을 하여 차원삭감

gamma는 scale를 사용함으로 자동으로 적절한 값을 정되도록 했음

accuracy	0.595
recall	0.62
precision	0.59

```python
# 최대 차원수로 PLS 적용
pls = PLSRegression(n_components=min(X_train_scaled.shape[1], 130))  # 차원이 많을 경우 상한선 100으로 설정
pls.fit(X_train_scaled, y_train)

# 각 성분의 기여도 비율 계산
explained_variance_ratio = np.var(pls.x_scores_, axis=0) / np.var(X_train_scaled, axis=0).sum()
cumulative_variance_ratio = np.cumsum(explained_variance_ratio)
```

```python
# PLS로 차원 축소 실행
pls = PLSRegression(n_components=n_components)
X_train_reduced = pls.fit_transform(X_train_scaled, y_train)[0]  # [0]으로 변환된 X를 추출
X_test_reduced = pls.transform(X_test_scaled)
```

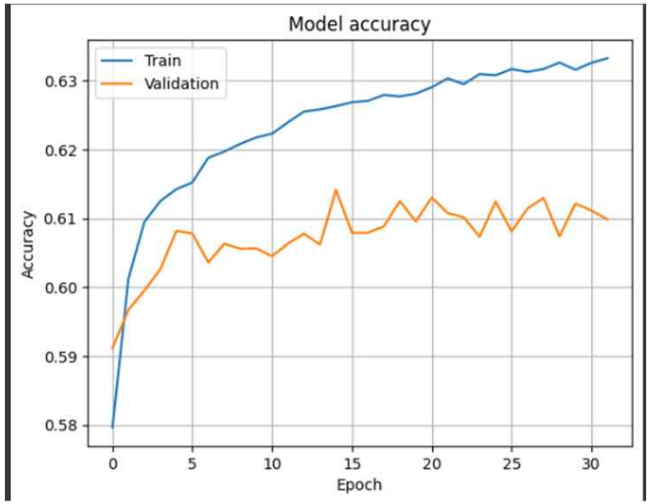
```python
# 서포트 벡터 머신 모델 생성
model = svm.SVC(gamma='scale')

# 교차 검증을 통한 점수 계산
cv_scores = cross_val_score(model, X_train_reduced, y_train, cv=5, scoring='accuracy')

# 교차 검증 결과 출력
print(f"Cross-validation scores (5-fold): {cv_scores}")
print(f"Average cross-validation score: {cv_scores.mean():.4f}")

# 모델을 훈련 데이터 전체로 학습
model.fit(X_train_reduced, y_train)

# 테스트 데이터를 사용하여 예측
y_pred = model.predict(X_test_reduced)

print("Accuracy on Test Set:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

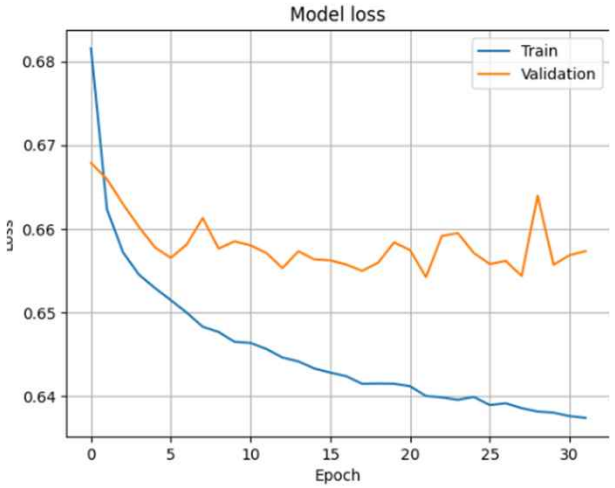
모델 작성(MLPC)

다층퍼셉트론 분류 모델

Sequential를 사용 했음

input층 hidden층 output층 3층으로 구성





accuracy	0.61
recall	0.61
precision	0.61

```
n_input = X_train.shape[1]  # 입력 특징의 수
n_hidden = 16  # 은닉층의 유닛 수
n_output = 1   # 출력층의 유닛 수

# MLP 모델 정의
mlp = Sequential()

# 은닉층 추가 (입력층은 생략하고, 첫 번째 은닉층에서 input_shape 지정)
mlp.add(Dense(units=100, input_shape=(n_input,), bias_initializer='zeros'))  # 첫 번째 은닉층
mlp.add(Dense(units=n_hidden, activation='relu', kernel_initializer=HeNormal(), bias_initializer='zeros'))  # 은닉층
mlp.add(Dense(units=n_output, activation='sigmoid', kernel_initializer=HeNormal(), bias_initializer='zeros'))  # 출력층

# 모델 컴파일
mlp.compile(optimizer=Adam(learning_rate=0.001), loss='binary_crossentropy', metrics=['accuracy'])

# 조기 종료 설정
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

# 모델 훈련
hist = mlp.fit(X_train_scaled, y_train, batch_size=32, epochs=100,
               validation_data=(X_test_scaled, y_test),
               verbose=2)
```

모델 작성(LightBGM)

```python
from sklearn.metrics import f1_score
# LGBMClassifier
lgb = LGBMClassifier()
lgb.fit(X_train, y_train)

y_pred1 = lgb.predict(X_test)

# 평가
print(classification_report(y_test, y_pred1))
```

	precision	recall	f1-score	support
0	0.64	0.63	0.64	10053
1	0.63	0.65	0.64	9947
accuracy			0.64	20000
macro avg	0.64	0.64	0.64	20000
weighted avg	0.64	0.64	0.64	20000

```python
# 랜덤 서치
params = {
    'objective': 'binary',
    'boosting_type': 'gbdt',
    'metric': 'auc',
    'num_leaves': 56,
    'learning_rate': 0.1,
    'feature_fraction': 0.7,
    #add697215.88
    'min_child_samples': 61,
    'max_depth': 7
}

# StratifiedKFold 설정
n_splits = 10
skf = StratifiedKFold(n_splits=n_splits, shuffle=True, random_state=42)

for train_index, valid_index in skf.split(X, y):
    X_train, X_valid = X.iloc[train_index], X.iloc[valid_index]
    y_train, y_valid = y.iloc[train_index], y.iloc[valid_index]
    model = LGBMClassifier(**params)
    model.fit(X_train, y_train)
    models.append(model)

# 예측
for model in models:
    y_pred = model.predict(X_test)

# 평가
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
0	0.70	0.69	0.69	10053
1	0.69	0.70	0.69	9947
accuracy			0.69	20000
macro avg	0.69	0.69	0.69	20000
weighted avg	0.69	0.69	0.69	20000

모델 향상을 위한 Outlier처리에 따른 분석

```python
df = pd.read_csv("clustering_num.csv")
import numpy as np

def replace_outliers_with_quartiles(data, c):
    q1 = np.percentile(data[c], 25)
    q3 = np.percentile(data[c], 75)

    IQR = q3 - q1

    upper_fence = q3 + 1.5 * IQR
    lower_fence = q1 - 1.5 * IQR

    data[c] = np.where(data[c] > upper_fence, q3,
                       np.where(data[c] < lower_fence, q1, data[c]))

    return data

for i in range(len(Outlier)):
    all_df = replace_outliers_with_quartiles(all_df, Outlier[i])
```

	precision	recall	f1-score	support
0.0	0.60	0.58	0.59	9920
1.0	0.58	0.61	0.60	9734
accuracy			0.59	19654
macro avg	0.59	0.59	0.59	19654
weighted avg	0.59	0.59	0.59	19654

	precision	recall	f1-score	support
0.0	0.50	1.00	0.67	9920
1.0	1.00	0.00	0.00	9734
accuracy			0.50	19654
macro avg	0.75	0.50	0.34	19654
weighted avg	0.75	0.50	0.34	19654

모델의 예상결과와 평가를 보니,
Outlier부분에 해약률이 큰사람의 특징이 포함되어 있었다
.
이에 따라 다른 사람에 비해 사용시간, 전화횟수, 전화시간이
특이하게 높거나 낮은 사람들이 해약하는 추세가 있다는 것을 알게 되었다

결론

이번 프로젝트의 데이터 분석과 모델 제작을 통해 해약하는 사람들의 특징으로 “ 사용도가 낮다, 가족 많다”는 것을 알게 되었다.
이런 특징에 주목하고 더 성능이 좋은 모델로 예측할 수 있게 되면 그 예측의 결과를 바탕으로 더욱 정확한 정보를 얻을 수 있으며,
앞으로 해약한다고 판단된 고개들에게 해약을 방지하기 위해 고객에 맞는 계약형태를 추천하거나 금전적인 우대와 같은 대처를 함으로 이익의 유지 또는 향상 시킬 수 있을 것이다.

잘 들어주셔서 감사합니다