DATE
11/22/2023

인공지능 발표

은행 고객 이탈 예측
컴퓨터공학전공 김나연

INDEX

01 개요 및 필요성

은행에서 고객 유지 및 고객 유치는 은행 운영에 있어서 중요한 부분이다.
어떤 변수들이 이탈 여부에 영향을 미치는지 살펴보고 이탈 예측 모델을 구현해본다.
은행에서는 이탈 여부에 미치는 변수들을 파악해 이와 관련하여 고객 유지 전략을 세울 수 있다.

02 데이터 확인 및 전처리

column	explain	column	explain	column	explain
customer_id	고객 고유 식별자	customer_nw_category	고객 순 자산 카테고리	previous_month_credit	전월 크레딧 금액
vintage	은행 계약 기간	branch_code	지역 식별 코드	current_month_debit	이번달 차변 금액
age	나이	current_balance	고객 계정 현재 잔액	previous_month_debit	전월 차변 금액
gender	성별	previous_month_end_balance	전월 말 계정 잔액	current_month_balance	이번달 계좌 잔액
dependents	부양 가족 수	average_monthly_balance_prevQ	전 분기 평균 월별 잔액	previous_month_balance	전월 계정 잔액
occupation	직업	average_monthly_balance_prevQ2	직전 두번째 분기의 평균 월별 잔액	churn	이탈 여부
city	거주 도시	current_month_credit	이번 달 크레딧 금액	last_transaction	고객 마지막 거래 시간

02 데이터 확인 및 전처리

```python
## dataset에 각 열에 대한 null값 비율 확인

for col in df.columns:
    percentage_null = np.round((df[col].isnull().sum()*100) / len(df[col]),2)
    print('Null Values for column {} is {}%'.format(col, percentage_null))
```

```
Null Values for column customer_id is 0.0%
Null Values for column vintage is 0.0%
Null Values for column age is 0.0%
Null Values for column gender is 1.85%
Null Values for column dependents is 8.68%
Null Values for column occupation is 0.28%
Null Values for column city is 2.83%
Null Values for column customer_nw_category is 0.0%
Null Values for column branch_code is 0.0%
Null Values for column current_balance is 0.0%
Null Values for column previous_month_end_balance is 0.0%
Null Values for column average_monthly_balance_prevQ is 0.0%
Null Values for column average_monthly_balance_prevQ2 is 0.0%
Null Values for column current_month_credit is 0.0%
Null Values for column previous_month_credit is 0.0%
Null Values for column current_month_debit is 0.0%
Null Values for column previous_month_debit is 0.0%
Null Values for column current_month_balance is 0.0%
Null Values for column previous_month_balance is 0.0%
Null Values for column churn is 0.0%
Null Values for column last_transaction is 0.0%
```

- dependents, city에 잘못된 데이터 유형 확인
- gender,dependents, occupation, city에 null값 존재

02 데이터 확인 및 전처리

```
## 'dependents' and 'city'의 데이터 타입 변환
df1[['dependents','city']] = df1[['dependents','city']].astype('int64')

## 'last_transaction'을 datetime으로 변환
df1['last_transaction'] = pd.to_datetime(df1['last_transaction'], errors='coerce').replace('Nat','0000-00')
df1['last_transaction_period'] = df1['last_transaction'].dt.to_period('M').replace('NaT', '0000-00')

df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 24832 entries, 0 to 28381
Data columns (total 22 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   customer_id                     24832 non-null  int64
 1   vintage                         24832 non-null  int64
 2   age                             24832 non-null  int64
 3   gender                          24832 non-null  object
 4   dependents                      24832 non-null  int64
 5   occupation                      24832 non-null  object
 6   city                            24832 non-null  int64
 7   customer_nw_category            24832 non-null  int64
 8   branch_code                     24832 non-null  int64
 9   current_balance                 24832 non-null  float64
 10  previous_month_end_balance      24832 non-null  float64
 11  average_monthly_balance_prevQ   24832 non-null  float64
 12  average_monthly_balance_prevQ2  24832 non-null  float64
 13  current_month_credit            24832 non-null  float64
 14  previous_month_credit           24832 non-null  float64
 15  current_month_debit             24832 non-null  float64
 16  previous_month_debit            24832 non-null  float64
 17  current_month_balance           24832 non-null  float64
 18  previous_month_balance          24832 non-null  float64
 19  churn                           24832 non-null  int64
 20  last_transaction                22067 non-null  datetime64[ns]
 21  last_transaction_period         22067 non-null  period[M]
dtypes: datetime64[ns](1), float64(10), int64(8), object(2), period[M](1)
memory usage: 4.4+ MB
```

- 잘못 들어간 데이터 타입 변환
- null값 제거

02 데이터 확인 및 전처리

```
df1.describe()
```

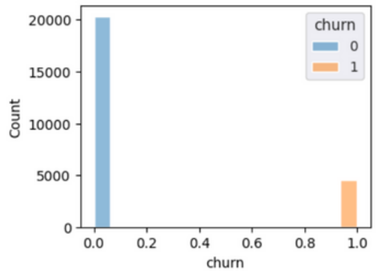
	customer_id	vintage	age	dependents	city	customer_nw_category	branch_code	current_balance	previous_month_end_balance	average_monthly_balance_prevQ	average_monthly_balance_prevQ2
count	24832.000000	24832.000000	24832.000000	24832.000000	24832.000000	24832.000000	24832.000000	2.483200e+04	2.483200e+04	2.483200e+04	2.483200e+04
mean	15120.719555	2090.259907	47.818903	0.352368	798.836783	2.215689	864.746013	7.005608e+03	7.102969e+03	7.068944e+03	6.641527e+03
min	1.000000	73.000000	1.000000	0.000000	0.000000	1.000000	1.000000	-5.503960e+03	-3.145380e+03	1.428690e+03	-1.650610e+04
25%	7525.500000	1957.000000	36.000000	0.000000	409.000000	2.000000	159.000000	1.799033e+03	1.916702e+03	2.193320e+03	1.847767e+03
50%	15117.500000	2153.000000	46.000000	0.000000	837.000000	2.000000	531.000000	3.294560e+03	3.387745e+03	3.539225e+03	3.371800e+03
75%	22680.250000	2292.000000	60.000000	0.000000	1096.000000	3.000000	1364.000000	6.629138e+03	6.655170e+03	6.662980e+03	6.517560e+03
max	30301.000000	2476.000000	90.000000	52.000000	1649.000000	3.000000	4782.000000	1.398486e+06	1.398486e+06	1.398486e+06	1.389627e+06
std	8736.591147	273.916048	16.864493	1.007858	430.826252	0.663905	890.994940	2.130752e+04	2.200392e+04	2.047696e+04	1.866737e+04

- 순 자산 범주는 1-3인데 세부 정보가 표시되어 있지 않아 조사 필요

02 데이터 분석 및 시각화

전체 데이터에서 이탈 고객 확인

```
fig, ax = plt.subplots(1,1, figsize=(4,3))
sns.set(font_scale=0.9)

## Plot the number of of customers who churned versus who did not
sns.histplot(data=df1, x='churn', hue='churn', palette=sns.color_palette('tab10', 2))

plt.tight_layout()
plt.show()

## Print the number of customers churned and not churned
no_churned = (df['churn'] == 0).sum()
no_churned1 = (df1['churn'] == 1).sum()
total_customers = (len(df['customer_id']))
percent_churned = (round((no_churned / total_customers) * 100, 2))
percent_churned1 = (round((no_churned1 / total_customers) * 100, 2))

print('전체 고객 중 이탈하지 않은 고객은 {},이며 전체고객의 {}%이다. '.format(no_churned, percent_churned))
print('전체 고객 중 이탈한 고객은 {}이며, null값을 포함하여 전체 고객의 {}%이다.'.format(no_churned1, percent_churned1))
```



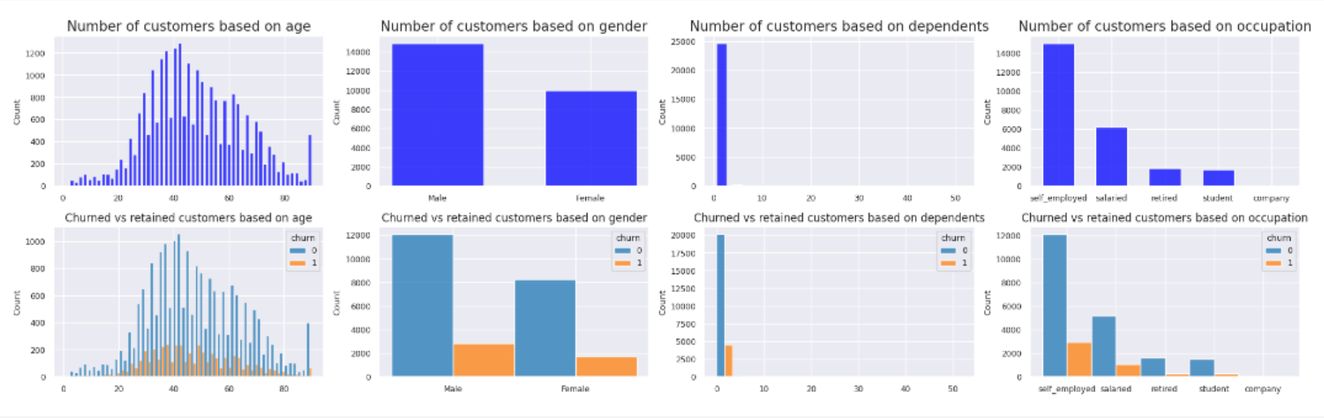

```
전체 고객 중 이탈하지 않은 고객은 23122,이며 전체고객의 81.47%이다.
전체 고객 중 이탈한 고객은 4518이며, null값을 포함하여 전체 고객의 15.92%이다.
```

- 이탈하지 않은 고객(0) : 23,122명(81.47%)
- 이탈한 고객(1) : 4,518명(15.92%)
- 은행에서 이탈한 고객 비율은 전체 고객의 20%미만
- null 값 제거 후 15.92%로 감소

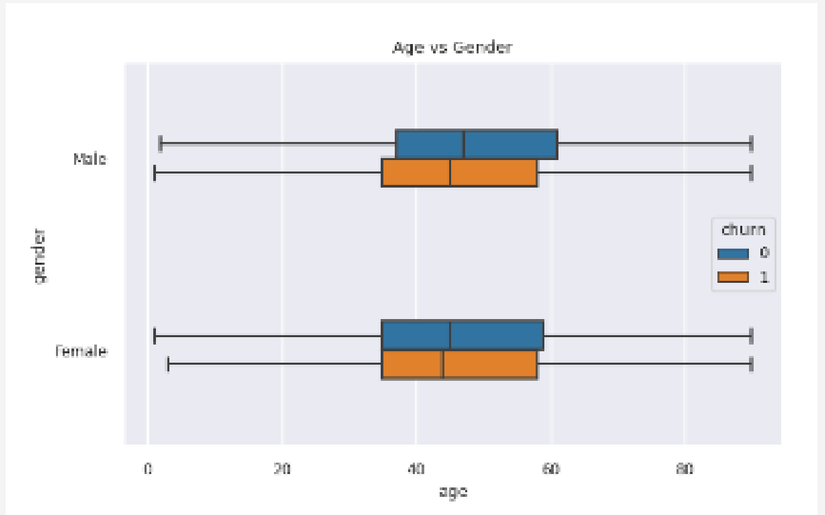
03 데이터 분석 및 시각화

age, gender, dependents, occupation과 churn의 관계 분석

```
social_columns = ['age','gender','dependents','occupation']

fig, axes = plt.subplots(2,4, figsize=(20,6))

axes=axes.flatten()

for i, col in enumerate(social_columns):
    ## 'age', 'gender', 'dependents', 'occupation'에 따른 고객 수 확인
    sns.histplot(data=df1, x=col, multiple='dodge', legend=False, color='blue', shrink=0.6, ax=axes[i])
    axes[i].set_title('Number of customers based on {}'.format(col), fontsize=15)
    axes[i].set_xlabel(xlabel='')
    ## Plot the relationship between 'age', 'gender', 'dependents', 'occupation', and 'churn'
    sns.histplot(data=df1, x=col, hue='churn', multiple='dodge', palette=sns.color_palette('tab10', 2), shrink=1, ax=axes[i+4])
    axes[i+4].set_title('Churned vs retained customers based on {}'.format(col).replace('_', ' '), fontsize=12)
    axes[i+4].set_xlabel(xlabel='')

plt.tight_layout()
plt.show()
```

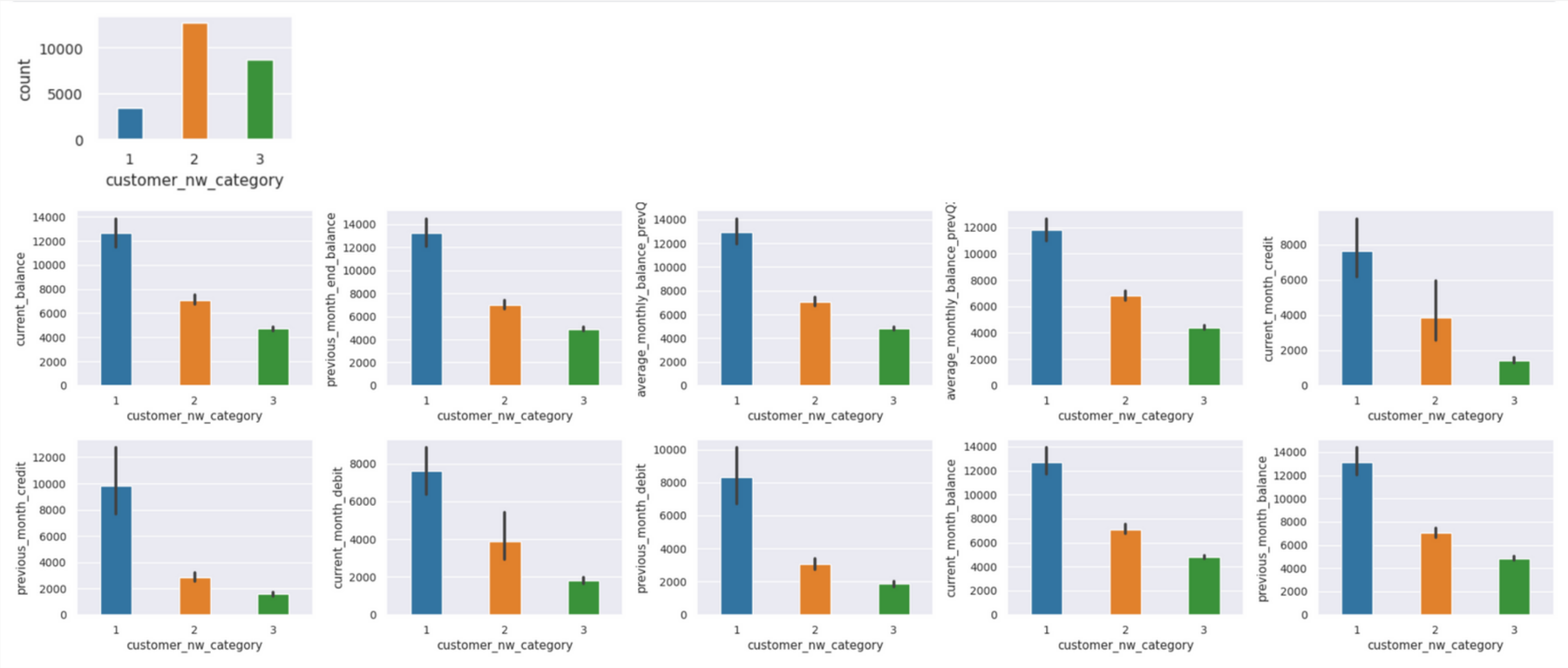


- 30~50세 연령대에서 이탈과 유지 고객이 많음
- 여성 고객에 비해 남성 고객이 더 많이 이탈
- 이탈한 고객 중 부양가족이 없는 고객이 대부분
- 많은 수의 고객이 자영업자 범주에 속함.



03 데이터 분석 및 시각화

customer_nw_category 범주 조사

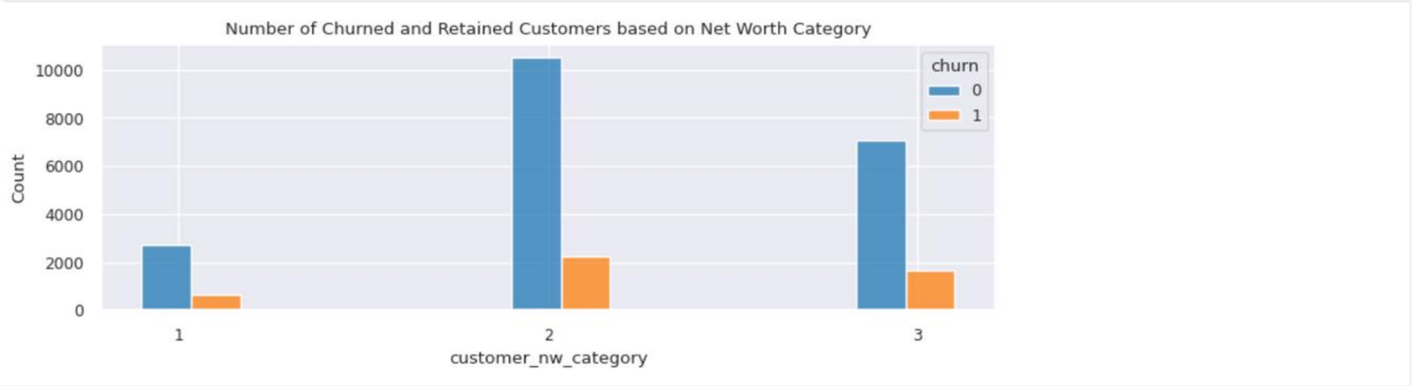


- 순 자산 카테고리가 1에 속하는 고객은 잔액, 신용, 차변이 가장 높다
- 카테고리 1,2,3 순으로 순 자산 분류(1의 순 자산이 가장 높음)
- 순 자산 카테고리 2에 속한 고객이 가장 많다

03 데이터 분석 및 시각화

순 자산과 이탈 여부 확인

```
fig = plt.subplots(1,1, figsize=(8,3))

## Plot the relationship between 'customer_nw_category' and 'churn'
sns.histplot(data=df1, x='customer_nw_category', hue='churn', multiple='dodge', shrink=4, palette=sns.color_palette('tab10', 2))
plt.xticks([1,2,3])
plt.title('Number of Churned and Retained Customers based on Net Worth Category')

plt.tight_layout()
plt.show()
```

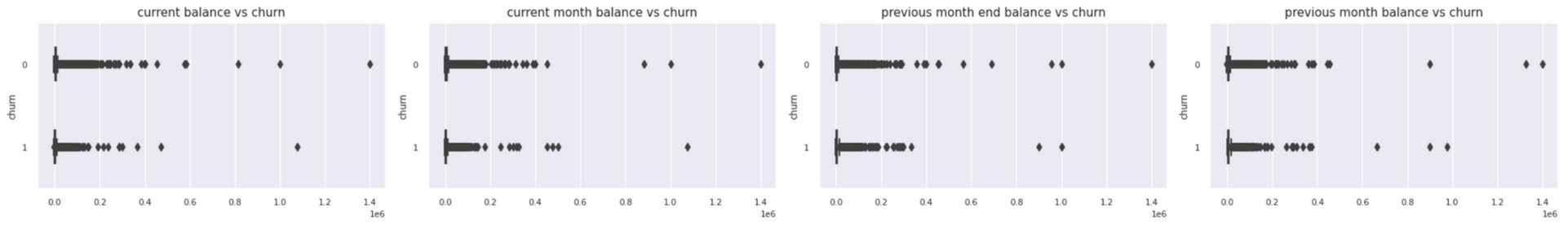


- 순자산 카테고리2에 속하는 고객이 이탈할 가능성이 가장 높다.
- 이상치 제거 후 순 자산 카테고리 2,3의 이탈 고객 수가 비슷해 짐

03 데이터 분석 및 시각화

순 자산과 이탈 여부 확인

```
fig, axes = plt.subplots(1,4, figsize=(20,3))
axes=axes.flatten()
sns.set(font_scale=0.9)

curr_prev_month_balance = ['current_balance','current_month_balance','previous_month_end_balance','previous_month_balance']

for i, col in enumerate(curr_prev_month_balance):
    sns.boxplot(data=df1, x=col, y='churn', orient='h', palette=sns.color_palette('tab10', 2), width=0.4, ax=axes[i])
    axes[i].set_title('{} vs churn'.format(col.replace('_',' ')))
    axes[i].set_xlabel(xlabel='')

plt.tight_layout()
plt.show()
```



- 이탈한 고객은 유지한 고객에 비해 현재 잔액이 더 낮다.
- 이탈한 고객은 전월 대비 잔액도 낮다.
- boxplot으로 시각화 한 결과 전체적으로 네 변수 모두 이탈하지 않은 사람들의 값이 좀 더 높음을 확인 가능

04 모델 생성 및 학습

사용모델 - XGBoost

Boosting

- 앙상블 기법 중 하나
- 약한 예측 모형들의 학습 에러에 가중치를 두고 이후에 이를 반영하여 강한 예측 모형 생성

Gradient Boost

XGBoost

- 병렬 처리로 학습하여 분류 속도가 빠름
- 과적합 규제(강한 내구성)
- 분류, 회귀영역에서 뛰어난 예측 성능

04 모델 생성 및 학습

XGBoost 모델 생성

```
df1_model_xgb = df1_model.drop(['age','gender'], axis=1)

df1_model_xgb = pd.get_dummies(data=df1_model_xgb, columns=['occupation'], drop_first=True, dtype=int)

le = LabelEncoder()
df1_model_xgb['last_transaction_period_encoded'] = le.fit_transform(df1_model['last_transaction_period'])

df1_model_xgb.head()
```

- 앞서 데이터 분석을 통해 나이, 성별은 이탈 여부와 유의미한 관계가 없음을 확인하여 해당 열 제거
- 직업 열의 경우 범주형 데이터를 포함하며, 마지막 접속 시간 열은 기간 데이터 유형으로 인코딩 필요

04 모델 생성 및 학습

데이터 분할

```
target_var = df1_model_xgb['churn']

predictor_var = df1_model_xgb.drop(['customer_id','churn','last_transaction','last_transaction_period'], axis=1)

X_train, X_test, y_train, y_test = train_test_split(predictor_var, target_var, test_size=0.30, stratify=target_var, random_state=42)
```

- churn 열을 target data로 설정
- 훈련 데이터와 테스트 데이터 분할(70% 30%)

04 모델 생성 및 학습

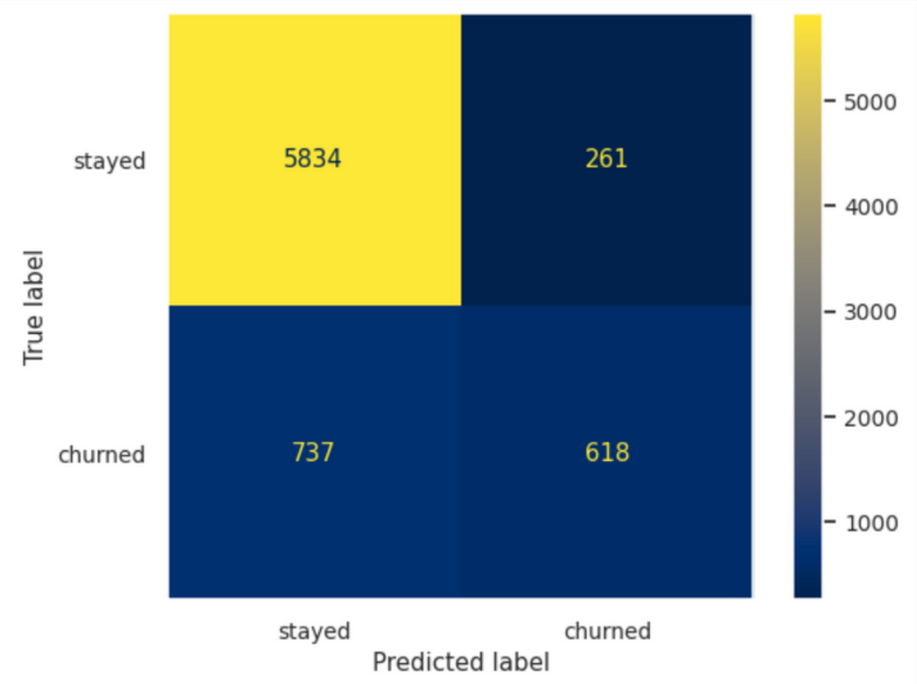
하이퍼파라미터 튜닝 후 모델 예측

```
xgb = XGBClassifier()

# GridSearch로 하이퍼파라미터 튜닝
params = {'max_depth':[4,5,6,7,8],
          'min_child_weight':[1,2,3,4,5],
          'learning_rate':[0.1,0.2,0.3],
          'n_estimators':[75,100,125]
          }

scores = make_scorer(lambda y_true, y_pred: np.sqrt(mean_squared_error(y_true, y_pred)), greater_is_better=False)

xgb1 = GridSearchCV(xgb, param_grid=params, cv=5, scoring=scores)

xgb1.fit(X_train, y_train)

best_params = xgb1.best_params_
y_preds = cross_val_predict(xgb1.best_estimator_, X_train, y_train, cv=5)

y_pred = xgb1.predict(X_test)
```

- 하이퍼파라미터 튜닝 한 후
- XGBoost 모델 예측

04 모델 생성 및 학습

성능 분석




```python
## confusion matrix생성
cm_xgb = confusion_matrix(y_test, y_pred)
disp_xgb = ConfusionMatrixDisplay(confusion_matrix=cm_xgb, display_labels=['stayed','churned'])
disp_xgb.plot(values_format='', cmap='cividis')

TP = cm_xgb[1,1]
TN = cm_xgb[0,0]
FP = cm_xgb[0,1]
FN = cm_xgb[1,0]

plt.grid(False)
plt.show()

print(f'The Number of True Positive results of the Confusion Matrix is {TP}')
print(f'The Number of True Negative results of the Confusion Matrix is {TN}')
print(f'The Number of False Positive results of the Confusion Matrix is {FP}')
print(f'The Number of False Negative results of the Confusion Matrix is {FN}')
```

- stayed : 실제 값이 0인 클래스(이탈하지 않음)
- churned : 실제 값이 1인 클래스(이탈함)
- TP : churned로 예측하고 실제로도 churn인 경우의 수
- TN : stayed로 예측하고 실제로도 stayed인 경우의 수
- FP : churned로 예측했지만 실제로 stayed인 경우의 수
- FN : stayed로 예측했지만 실제로 churned인 경우의 수

04 모델 생성 및 학습

성능 분석

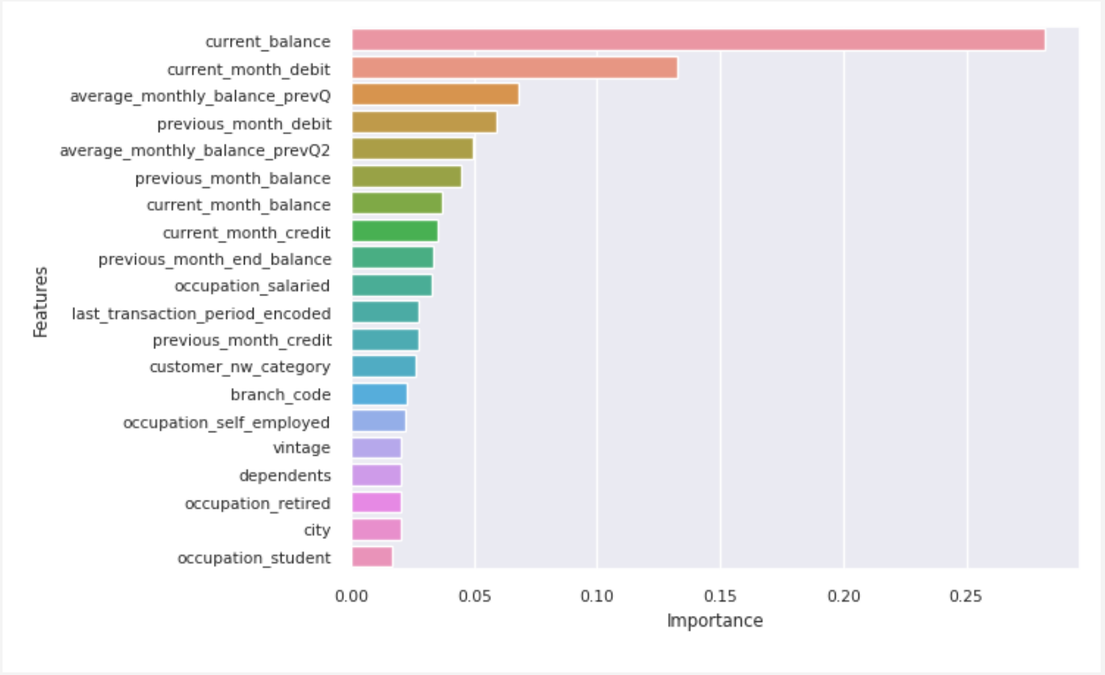
	precision	recall	f1-score	support
이탈하지 않음으로 예측	0.89	0.96	0.92	6704
이탈로 예측	0.72	0.46	0.56	1491
정확도			0.87	8195
macro 평균	0.80	0.71	0.74	8195
weighted평균	0.86	0.87	0.86	8195

- 전체 예측 중 올바르게 분류된 비율은 87%이다.
- 이탈 고객을 예측해 이에 대한 유지 전략을 세우는 것이 목표이기 때문에 모델이 관심 클래스를 얼마나 잘 예측하지는지 정밀도를 살펴보아야 함.
- 이탈하지 않음으로 예측한 경우 정밀도는 89%이며 이탈로 예측한 경우 72%가 실제 해당 클래스에 속함을 알 수 있다.
- 이후 모델이 데이터셋에서 잘못 예측한 행을 확인한 결과 총 1078개의 행으로 전체 예측 중 약 13.15%차지함을 확인

04 모델 생성 및 학습

가장 큰 영향을 미치는 feature 확인

```
## feature importance를 barplot으로 확인
xgb1_importance = pd.DataFrame(xgb1.best_estimator_.feature_importances_, columns=['gini_importance'], index=predictor_var.columns)

xgb1_importance = xgb1_importance.sort_values(by='gini_importance', ascending=False)

xgb1_importance

sns.barplot(data=xgb1_importance, x='gini_importance', y=xgb1_importance.index, orient='h')
plt.ylabel(ylabel='Features')
plt.xlabel(xlabel='Importance')

plt.show()
```



- gini importance로 각 feature들의 중요도를 확인해 본 결과 고객 계좌의 현재 잔액, 이번 달 차변 금액, 전 분기의 평균 월별 잔액 세 변수가 큰 영향을 미침을 확인 할 수 있음.

04 모델 생성 및 학습

변수 중요도를 고려하여 여러 모델로 학습

```
X = df1_model_test[['current_balance','current_month_debit','average_monthly_balance_prevQ',]]
y = df1_model_test['churn']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, stratify=y, random_state=42)
```

- 고객 계정의 현재 잔액, 이번 달 차변 금액, 전 분기의 평균 월별 잔액 세 변수를 바탕으로 고객 이탈을 예측

04 모델 생성 및 학습

모델 비교

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV

X = df1_model_test[['current_balance','current_month_debit','average_monthly_balance_prevQ',]]
y = df1_model_test['churn']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, stratify=y, random_state=42)

models = [
    ("Decision Tree", DecisionTreeClassifier()),
    ("Random Forest", RandomForestClassifier()),
    ("XGBoost", XGBClassifier())
]


for name, model in models:
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    confusion = confusion_matrix(y_test, y_pred)

    print(f"Model: {name}")
    print(f"Accuracy: {accuracy}")
    print("Confusion Matrix:")
    print(confusion)
    print("\n")
```

Model	Accuracy	Confusion Matrix
Decision Tree	0.798	[[5362 733] [765 590]]
RandomForest	0.856	[[5786 309] [762 593]]
XGBoost	0.857	[[5780 315] [750 605]]

05 결론

- 고객의 순 자산이나, 계좌 잔액 등을 확인하여 이탈 여부를 확인할 수 있었다.

- 은행에서는 이를 바탕으로 관련된 마케팅 전략을 구축하거나 맞춤형 서비스를 제공할 수 있다.

NAME OF PROJECT
your name

DATE
11/22/2023

감사합니다

NAME OF COMPANY
your name

PRESENTED BY
your name