TEAM 디비디비딥! (DBDB deeplearning)

mini project

머신러닝을 이용한 당뇨예측

Prepared by 조서현/이수현/김유진

INDEX

Index_ *팀소개*

깃허브 및 공동작업 플랫폼 활용하여 협업.

01

조서현

- 도메인 사전조사
- 사전 분석 및 시각화

02

이수현

- 데이터 전처리
- 머신러닝 모델 학습

03

김유진

- 머신러닝 모델 웹 적용
- 웹 설문작성 및 시각화

project Summary

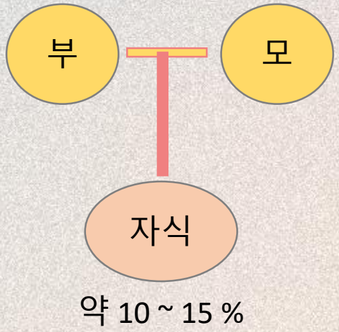
Index_*프로젝트 방향*

당뇨의 발병 원인

유전적 원인

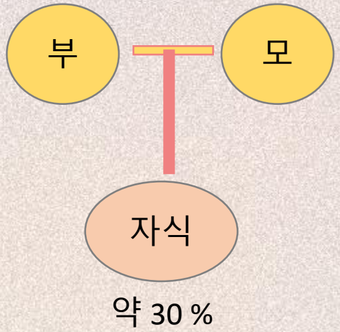
췌장의 베타세포 유전적 결함
인슐린 수용체 유전적 결함
인슐린 작용력을 감소시키는 유전자

부모 중 한 명만 당뇨 有



약 10 ~ 15 %

부모 둘 다 당뇨 有



약 30 %

환경적 원인

스트레스
노화
비만증
운동부족
감염
외상
수술
임신 및 약물
나쁜 식생활

Index_*사용데이터*

- **사용 데이터**
  - [National Health Interview Survey 2018 Data Release] https://www.cdc.gov/nchs/nhis/nhis_2018_data_release.htm
  - Sample Adult file : samadult.csv 사용

- **데이터 분석 캡쳐**

```
import pandas as pd

df_a = pd.read_csv('samadult.csv')
df_a = df_a[['SEX','AGE_P','R_MARITL','DIBEV1','HYPEV','PREGNOW','DEP_2','BMI',
             'AFLHCA18','AFLHCA18','AFLHC29_','AFLHC31_','AFLHC32_','AFLHC33_',
             'SMKEV','ALC1YR','CHLEV','VIGNO','AUSUALPL','ASICNHC','HIT1A']]
df_a
```

	SEX	AGE_P	R_MARITL	DIBEV1	HYPEV	PREGNOW	DEP_2	BMI	AFLHCA18	AFLHCA18	...
0	2	79	4	1	1	NaN	2	2358	2.0	2.0	...
1	1	37	1	2	2	NaN	2	3279	NaN	NaN	...
2	1	29	1	2	2	NaN	2	4363	NaN	NaN	...
3	1	75	4	2	1	NaN	2	2229	2.0	2.0	...
4	1	39	1	2	2	NaN	2	2372	2.0	2.0	...
...	...	...	...	...	...	...	...	...	...	...	...
25412	2	19	7	2	2	2.0	2	2090	NaN	NaN	...
25413	2	49	7	2	1	2.0	8	2745	2.0	2.0	...
25414	2	40	7	2	1	2.0	2	2585	2.0	2.0	...
25415	2	61	1	2	2	NaN	2	2663	2.0	2.0	...
25416	2	70	5	1	1	NaN	2	2495	2.0	2.0	...

25417 rows × 21 columns

Index_*사전 데이터 분석*

- 먼저 분석할 컬럼을 추려내기 위한 시각화 하기 전 결측치(Nan) 값을 fillna( )함수를 사용 → 0 값으로 대체

```
df.isnull().sum()
```

```
FPX              0
FMX              0
HHX              0
INTV_QRT         0
WTIA_SA          0
               ...
RCS_AFD          0
PAIN_2A          0
ANX_3R        9582
DEP_3R       14683
COGCAUS2     20457
Length: 742, dtype: int64
```

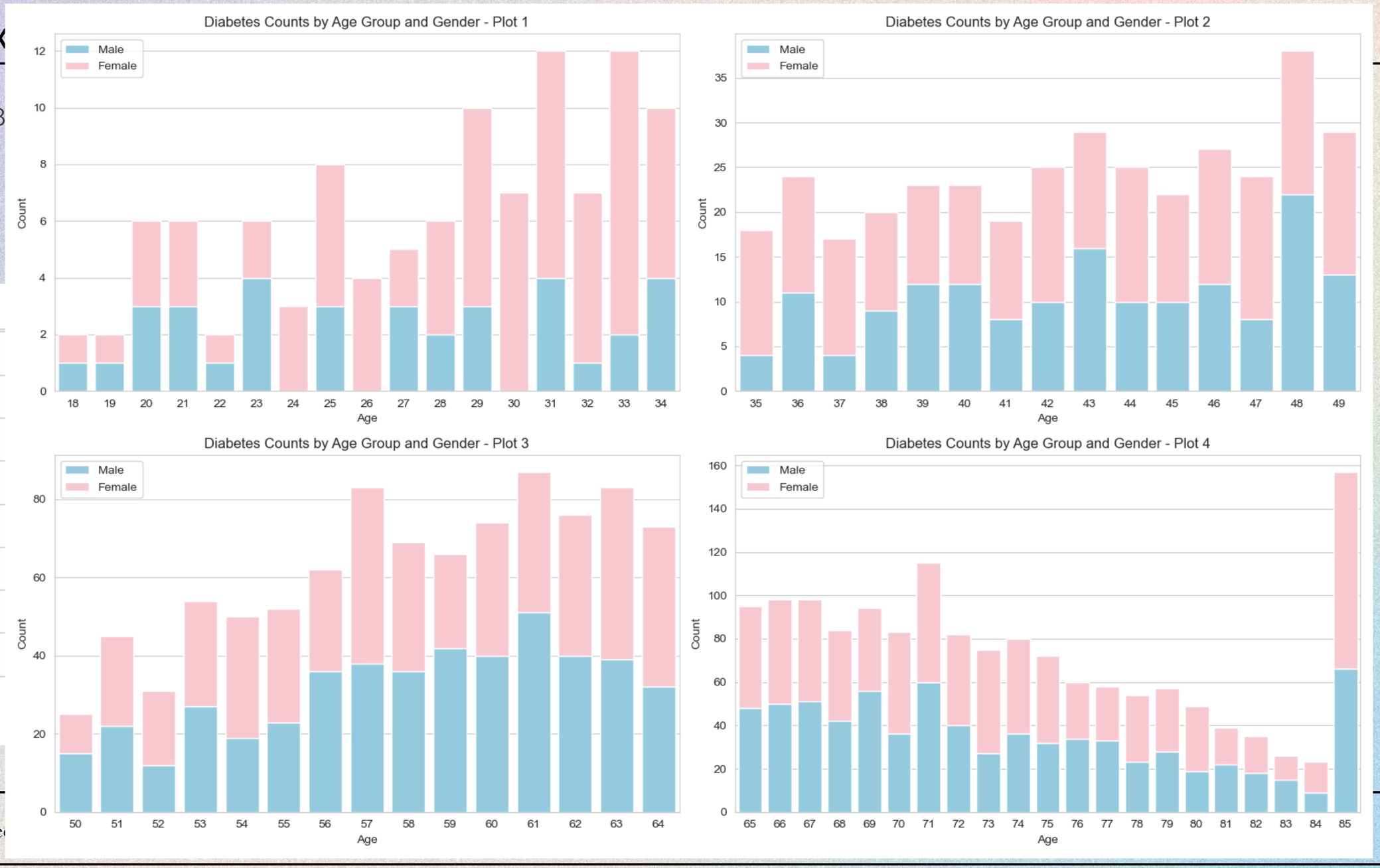
fillna( )

```
df.fillna(0, inplace=True)
df.isnull().sum()
```

```
FPX          0
FMX          0
HHX          0
INTV_QRT     0
WTIA_SA      0
              ..
RCS_AFD      0
PAIN_2A      0
ANX_3R       0
DEP_3R       0
COGCAUS2     0
Length: 742, dtype: int64
```

Index

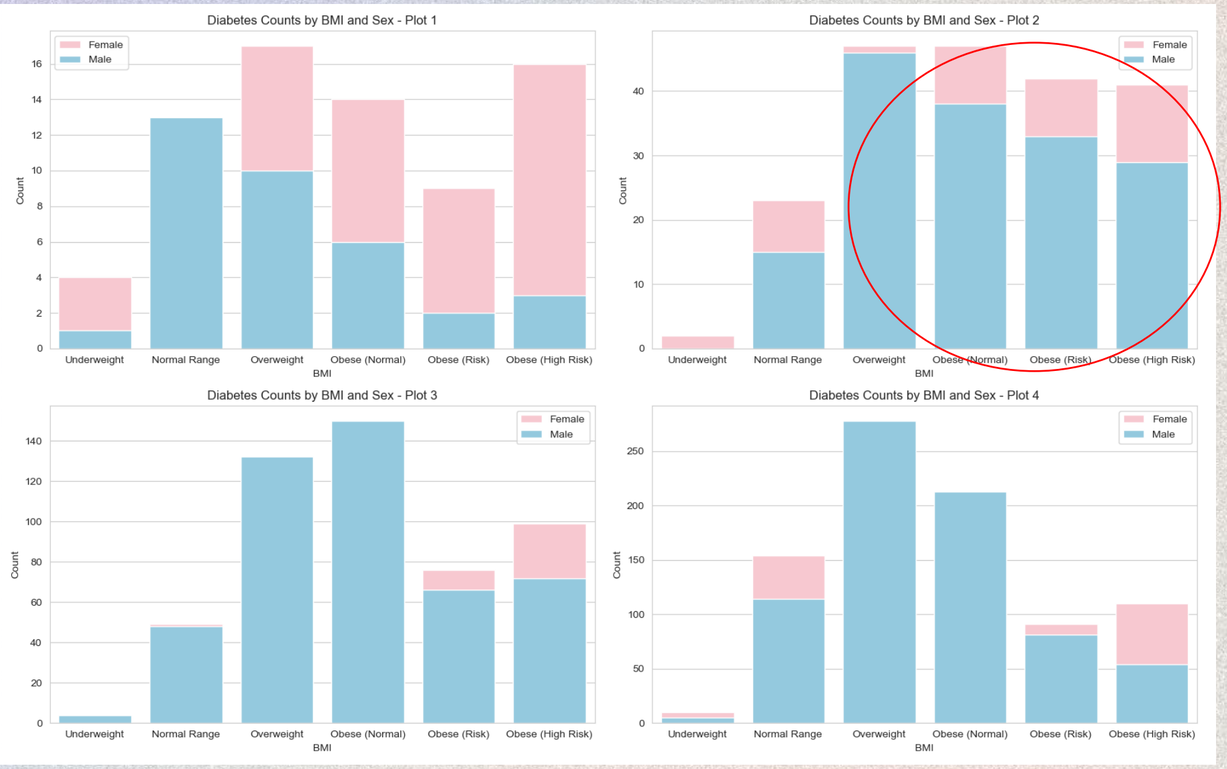
- 만 18
- 35세
- 50세
- 65세



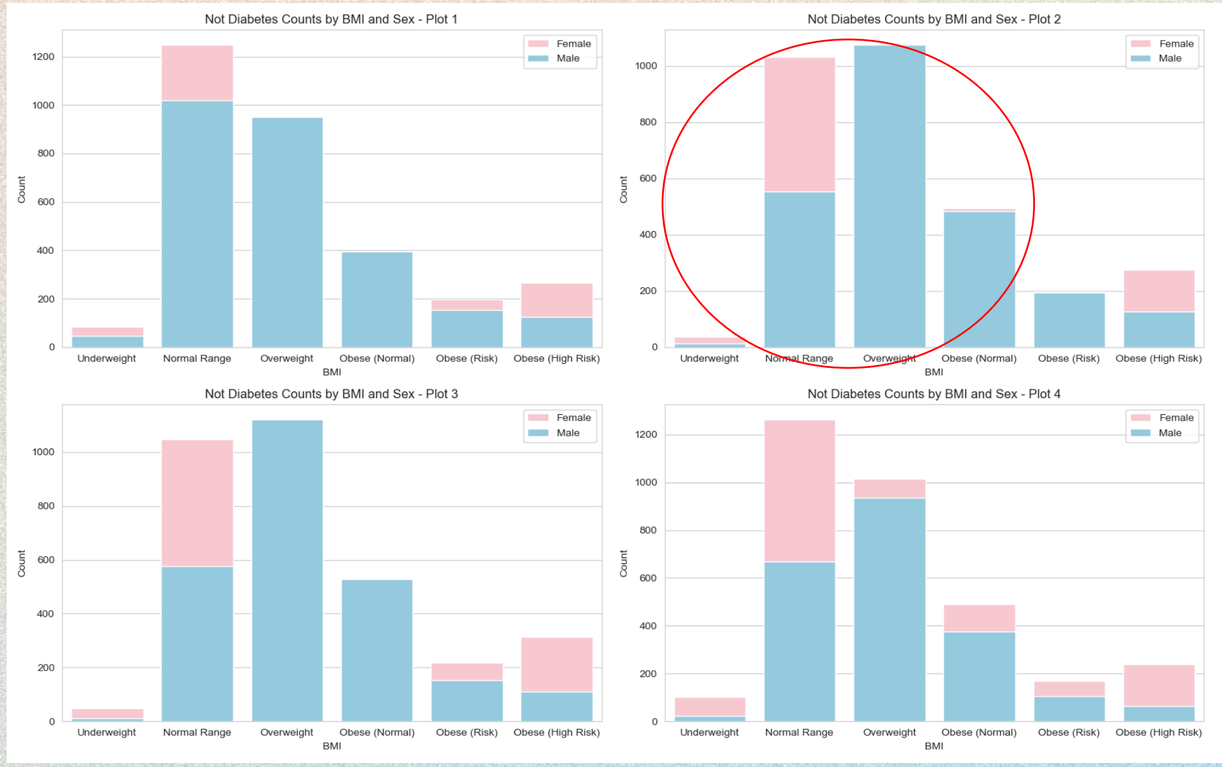
proje

Index_*사전 데이터 분석*

- **BMI**



△당뇨병 환자 BMI 그래프



△비 당뇨병 환자 BMI 그래프

당뇨병 환자들 경우 비 당뇨병 환자 BMI 보다 과체중에서 비만 빈도가 높을 것을 보아 BMI가 높을 수록 당뇨 발생이 높은 걸 볼 수 있음

project Summary

체중과 관련 있는 컬럼인 체중관리, BMI, 음주, 고지혈증, 신체활동빈도, 약물

Index_*사전 데이터 분석*

- 임신 당뇨병

호르몬 및 체중 변화

체지방 증가 ↑

인슐린 분비 충분 X

임신 당뇨병 발생

임신 당뇨병이 있었던 여성은 임신 때 제발 위험이 50%정도로 추산됨

[질병관리청 국민건강포털]
https://health.kdca.go.kr/healthinfo/biz/health/gnrlzHealthInfo/gnrlzHealthInfo/gnrlzHealthInfoView.do?cntnts_sn=5271

Index_*세부 요건 분석*

환경 생활 요인에 따라 당뇨 발병률에 영향을 줄 것이라 판단.
성별, 연령 등 인구통계학적 요인 포함한 환경 요인별 당뇨 예측을 진행

1	DIBEV1	Everbeen told that you have diabetes	당뇨유무
2	HYPEV	Everbeen told you have hypertension	고혈압유무
3	PREGNOW	Currently pregnant	현재 임신여부
4	AFLHCA17	Depression/anxiety/emotional problem causes difficulty with a	우울증 여부
5	AFLHCA18	Weight problem causes difficulty with activity	체중문제
6	BMI	BMI	체질량지수
7	AFLHC29_	Alcohol/drug/substance abuse problem causes difficulty with	알콜 및 약물 남용
8	AFLHC31_	Surgical after-effects/medical treatment causes difficulty with	수술 후유증
9	AFLHC32_	"Old age"/elderly/aging-related problem causes difficulty with	신체노화
10	AFLHC33_	Fatigue/tiredness/weakness causes difficulty with activity	피로무기력증
11	SMKEV	Eversmoked 100 cigarettes	흡연유무
12	ALC1YR	Everhad 12+ drinks in any one year	1년내 음주경험
13	CHLEV	Evertold you had high cholesterol	고지혈증없음
14	VIGNO	Freq vigorous activity:# of units	신체활동빈도
15	AUSUALPL	Place USUALLY go when sick	보통 아플 때 가는 곳
16	ASICNHC	How worried are you about...medical costs of healthcare	의료비지출에 대한 인식
17	HIT1A	Looked up health information on Internet, past 12 m	인터넷 건강정보 검색유무 1년내
18	SEX	SEX	sex
19	R_MARITL	R_MARITL	결혼여부
20	AGE_P	AGE_P	나이
21	ACPTCHLD	Number of children in HH that responded	응답한 가구 내 아동의 수는 몇 명?
22	ACPT_PER	Number of persons in HH responding	응답한 가구 내 사람의 수는 몇 명?
23	FSBALANC	Could not afford to eat balanced meals	균형 잡힌 식사를 할 여유가 없었습니다

분석 라이브러리

```
# 분석 라이브러리
import pandas
import numpy
import sklearn
import streamlit
import joblib
```

Index_**데이터전처리**

기본 데이터프레임 생성

```
df_adu = pd.read_csv('samadult.csv')
df_family = pd.read_csv('familyxx.csv')
df_household = pd.read_csv('househld.csv', index_col='HHX')
# df_adu와 df_family를 HHX와 FMX 기준으로 inner join
merged_df = pd.merge(df_adu, df_family, how='inner', on=['HHX', 'FMX'])

# df_household와 merged_df를 HHX 기준으로 inner join
df = pd.merge(df_household, merged_df, how='inner', on='HHX')
df_col = pd.read_excel('분석변수_총정리.xls')
col = df_col['Unnamed: 0'].tolist()
col = [i.replace(' ','') for i in col]
df = df[col]
col_list = df_col['Unnamed: 2'].tolist()
df.columns = col_list
```

	당뇨 유무	고혈압 유무	현재 임신 여부	우울증 여부	체중문제	체질량지수	알콜 및 약물 남용	수술 후 유증	신체노화
0	1	1	NaN	2.0	2.0	2358	2.0	2.0	2.0
1	2	2	NaN	NaN	NaN	3279	NaN	NaN	NaN
2	2	2	NaN	NaN	NaN	4363	NaN	NaN	NaN
3	2	1	NaN	1.0	2.0	2229	2.0	2.0	2.0
4	2	2	NaN	2.0	2.0	2372	2.0	2.0	2.0

Index_**데이터전처리**

답변 1 / 2 로 정형화

```
df = df[~df['당뇨유무'].isin([3, 7, 9])]
df.fillna(0, inplace=True)
# 응답거부, 모름 등등을 0으로 처리
col_exclude = ['체질량지수', '나이', '결혼여부']
for column in df.columns:
    if column not in col_exclude:
        df.loc[df[column].isin([3, 7, 8, 9, 996,997,998,999]), column] = 0
# bmi가 7000 넘는데 당뇨가 아님. 무조건 잘못기입한 자료일것
df.loc[df['체질량지수'] >= 7000, '체질량지수'] = 0
# bmi가 4000 넘는데 당뇨가 아닐 리 없음
df.loc[df['체질량지수'] >= 4000, '당뇨유무'] = 1
# 신체활동빈도가 10이상 50미만일 때는 한달 기준으로 기입한 것으로 가정 -> /4 , 50이상일 때는 1년 기준으로 기입한 것으로 가정 -> /48
df['신체활동빈도'] = df['신체활동빈도'].apply(lambda x: round(x / 4) if 10 <= x < 50 else round(x / 48) if x >= 50 else x)
# 걱정됨->1, 걱정되지않음->2
df.loc[df['의료비지출에 대한 인식'].isin([1,2]), '의료비지출에 대한 인식']= 1
df.loc[df['의료비지출에 대한 인식'].isin([3,4]), '의료비지출에 대한 인식']= 2
# 결혼 함->1 , 결혼 안함->2
df.loc[df['결혼여부'].isin([1,2,3]), '결혼여부']= 1
df.loc[df['결혼여부'].isin([4,5,6,7,8,9]), '결혼여부']= 2
```

Index_데이터전처리

컬럼명 정리

```
# 원-핫 인코딩 적용
df_encoded = pd.get_dummies(df, columns=col_2)

# _1 대신 _yes로 변경
df_encoded = df_encoded.loc[:, ~df_encoded.columns.str.endswith('_0.0') & ~df_encoded.columns.str.endswith('_0')]
df = df_encoded
df.loc[df['당뇨유무']==2, '당뇨유무']=0
df.columns = [col.replace('_1', '_yes').replace('_2', '_no').replace('_1.0', '_yes').replace('_2.0', '_no').replace('.0', '') for col in df.columns]

df_1 = df[df['당뇨유무'] == 1]
df_2 = df[df['당뇨유무'] == 0]

df_2_1 = df_2.iloc[0:3786]
df_2_2 = df_2.iloc[3786:7572]

df_01 = pd.concat([df_1, df_2_1], axis=0)
df_02 = pd.concat([df_1, df_2_2], axis=0)

dfs = [df_01, df_02, df_03, df_04, df_05, df_06]
```

	당뇨 유무	체질량 지수	신체활 동빈도	나 이	응답한 가구 내 아동의 수는 몇 명?	응답한 가구 내 사람의 수는 몇 명?	고혈압유 무_yes	고혈압유 무_no	현재 임신 여부_yes	현재 임신 여부_no
0	1	2358	0	79	0.0	1.0	True	False	False	False
2	1	4363	4	29	1.0	0.0	False	True	False	False
10	1	2616	2	68	0.0	2.0	True	False	False	False
11	1	2080	0	76	0.0	1.0	True	False	False	False
20	1	3905	0	61	0.0	1.0	True	False	False	False

Index_**데이터전처리**

스케일링 및 데이터 scv 저장

```
# 원-핫 인코딩 적용
df_encoded = pd.get_dummies(df, columns=col_2)

# _1 대신 _yes로 변경
df_encoded = df_encoded.loc[:, ~df_encoded.columns.str.endswith('_0.0') & ~df_encoded.columns.str.endswith('_0')]
df = df_encoded
df.loc[df['당뇨유무']==2, '당뇨유무']=0
df.columns = [col.replace('_1', '_yes').replace('_2', '_no').replace('_1.0', '_yes').replace('_2.0', '_no').replace('.0', '') for col in df.columns]

df_1 = df[df['당뇨유무'] == 1]
df_2 = df[df['당뇨유무'] == 0]

df_2_1 = df_2.iloc[0:3786]
df_2_2 = df_2.iloc[3786:7572]

df_01 = pd.concat([df_1, df_2_1], axis=0)
df_02 = pd.concat([df_1, df_2_2], axis=0)

dfs = [df_01, df_02, df_03, df_04, df_05, df_06]
```

	당뇨유무	체질량지수	신체활동빈도	나이	응답한 가구 내 아동의 수는 몇 명?	응답한 가구 내 사람의 수는 몇 명?	고혈압유무_yes	고혈압유무_no	현재 임신 여부_yes	현재 임신 여부_no
0	1	2358	0	79	0.0	1.0	True	False	False	False
2	1	4363	4	29	1.0	0.0	False	True	False	False
10	1	2616	2	68	0.0	2.0	True	False	False	False
11	1	2080	0	76	0.0	1.0	True	False	False	False
20	1	3905	0	61	0.0	1.0	True	False	False	False

Index_**머신러닝**

머신러닝 GSCV 파라미터

```python
# 서포트 벡터 머신에 대한 그리드 서치 파라미터
svc_params = {
    'C': [0.1, 1, 10, 100],
    'kernel': ['linear', 'rbf', 'poly'],
    'gamma': ['auto', 0.1, 1, 10],
    'max_iter': [1000, 2000, 3000]
}
```

```python
# Decision Tree에 대한 그리드 서치 파라미터
dt_params = {
    'criterion': ['gini', 'entropy'],
    'splitter': ['best', 'random'],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
}
```

```python
# K-최근접 이웃에 대한 그리드 서치 파라미터
knn_params = {
    'n_neighbors': [3, 5, 7, 10, 15, 20, 25],
    'weights': ['uniform', 'distance'],
    'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute']
}
```

```python
# Adaboost에 대한 그리드 서치 파라미터
adaboost_params = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.2],
    'base_estimator': [
        DecisionTreeClassifier(criterion='gini', max_features=7, max_depth=1),
        DecisionTreeClassifier(criterion='gini', max_features=7, max_depth=2),
        DecisionTreeClassifier(criterion='entropy', max_features=7, max_depth=1),
        DecisionTreeClassifier(criterion='entropy', max_features=7, max_depth=2),
        DecisionTreeClassifier(criterion='gini', max_features=8, max_depth=1),
        DecisionTreeClassifier(criterion='gini', max_features=8, max_depth=2),
        DecisionTreeClassifier(criterion='entropy', max_features=8, max_depth=1),
        DecisionTreeClassifier(criterion='entropy', max_features=8, max_depth=2)
    ],
}
```

```python
# Naive Bayes에 대한 그리드 서치 파라미터
nb_params = {
    'var_smoothing': [1e-9, 1e-8, 1e-7, 1e-6, 1e-5, 1e-4, 1e-3]
}
```

```python
# Random Forest에 대한 그리드 서치 파라미터
rf_params = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt', 'log2'],
    'n_jobs': [-1]
}
```

```python
# XGBoost에 대한 그리드 서치 파라미터
xgb_params = {
    'learning_rate': [0.01, 0.1, 0.2],
    'n_estimators': [50, 100, 200],
    'max_depth': [3, 4, 5],
    'min_child_weight': [1, 2, 3],
    'gamma': [0, 1, 5],
    'subsample': [0.8, 0.9, 1.0],
    'colsample_bytree': [0.8, 0.9, 1.0],
    'n_jobs': [-1]
}
```

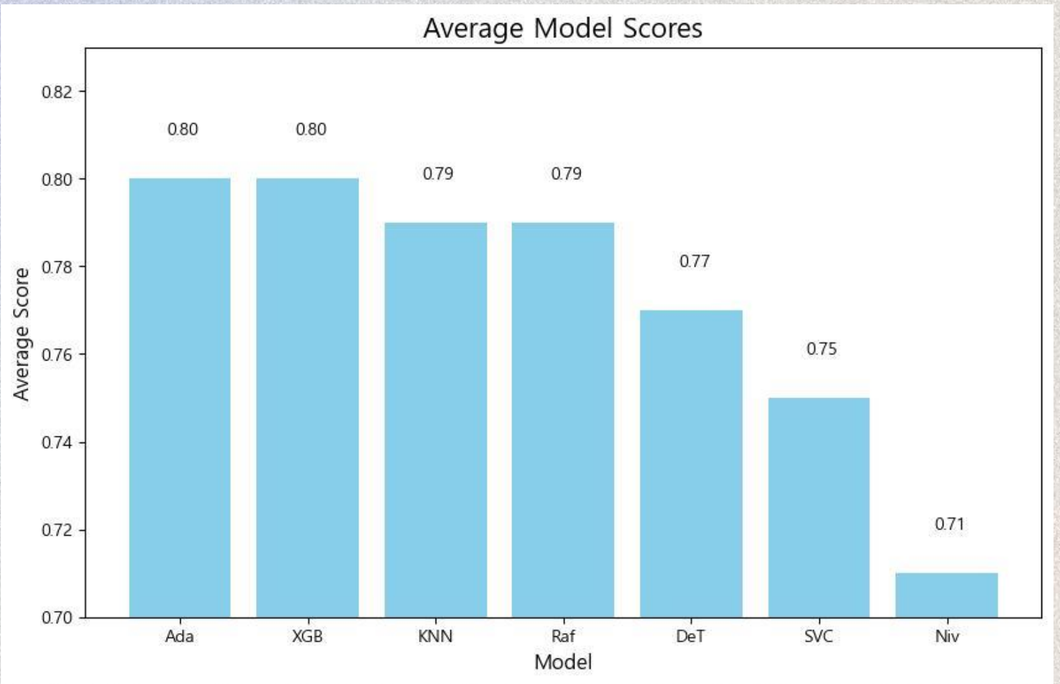
Index_**머신러닝**

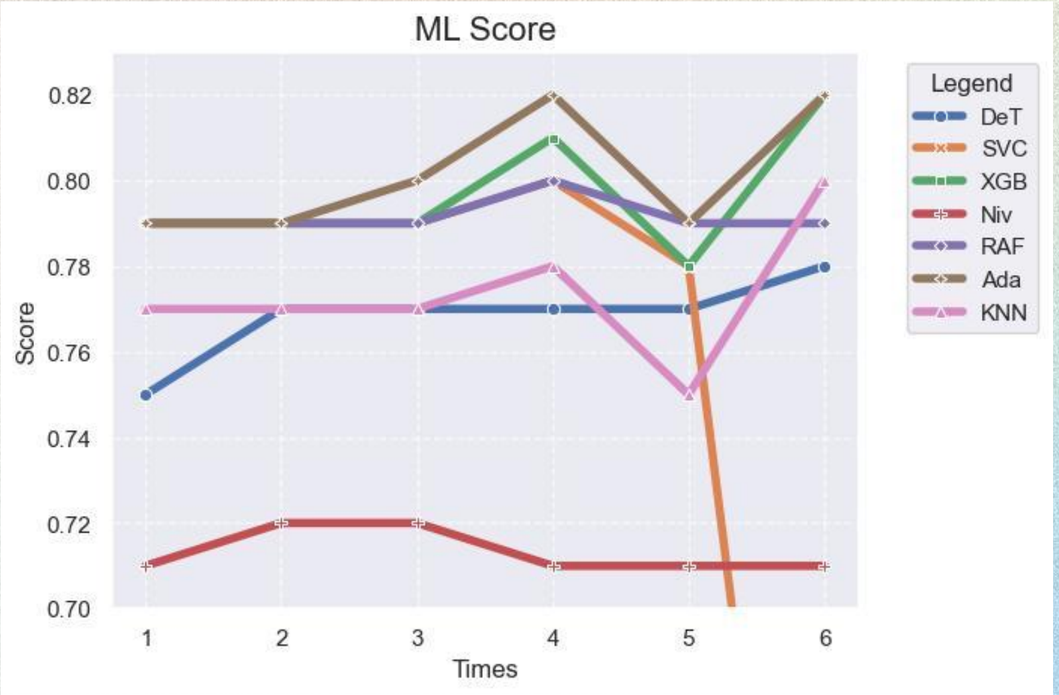
머신러닝 기본 틀

```python
# 각 데이터프레임에 대해 Decision Tree 모델 학습과 평가
for i in range(1, 7):
    train_csv_path = f'train_data_{i}.csv'
    test_csv_path = f'test_data_{i}.csv'

    print(f"\n---- Training and Evaluating Decision Tree for {train_csv_path} ----")

    # CSV 파일을 데이터프레임으로 읽기
    df_train = pd.read_csv(train_csv_path)

    # X, y 설정
    y_train = df_train['당뇨유무']
    X_train = df_train.drop('당뇨유무', axis=1)

    # 대응하는 test 데이터셋
    df_test = pd.read_csv(test_csv_path)
    X_test = df_test.drop('당뇨유무', axis=1)
    y_test = df_test['당뇨유무']

    # Decision Tree에 대한 그리드 서치
    dt_model = DecisionTreeClassifier()
    dt_gscv = GridSearchCV(dt_model, dt_params, cv=5, scoring='f1_macro')

    # 모델 학습
    dt_gscv.fit(X_train, y_train)

    # 결과 출력
    print(f"Decision Tree for {train_csv_path} - Best Parameters:", dt_gscv.best_params_)
    print(f"Decision Tree for {train_csv_path} - Best F1 Score:", dt_gscv.best_score_)

    # 모델에 대한 분류 리포트 출력
    dt_report = classification_report(y_test, dt_gscv.predict(X_test))
    print(f"Decision Tree Classification Report for {test_csv_path} (on Test Set):\n", dt_report)
```

Index_**머신러닝**

머신러닝 결과





Index_머신러닝

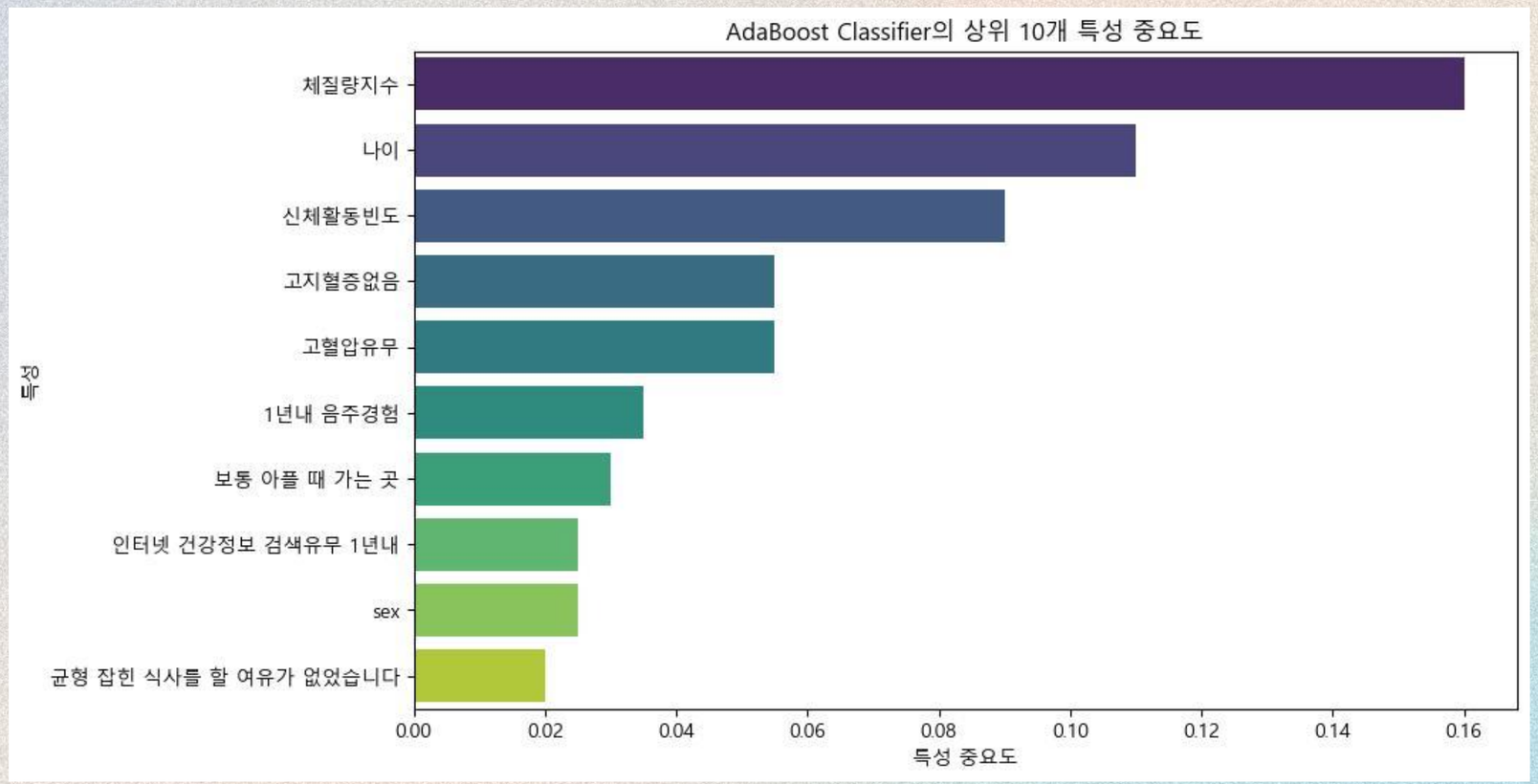
AdaBoost 최종 선택

```python
# AdaBoost 모델 생성
adaboost_model = AdaBoostClassifier(
    base_estimator=DecisionTreeClassifier(criterion='entropy', max_depth=1, max_features=8),
    learning_rate=0.1,
    n_estimators=100
)
```

```python
# 각 데이터프레임에 대해 Adaboost 모델 학습과 평가
while os.path.exists(train_csv_path) and os.path.exists(test_csv_path):
    print(f"\n---- Training and Evaluating Adaboost for {train_csv_path} ----")

    # CSV 파일을 데이터프레임으로 읽기
    df_train = pd.read_csv(train_csv_path)

    # X, y 설정
    y_train = df_train['당뇨유무']
    X_train = df_train.drop('당뇨유무', axis=1)

    # 대응하는 test 데이터셋
    df_test = pd.read_csv(test_csv_path)
    X_test = df_test.drop('당뇨유무', axis=1)
    y_test = df_test['당뇨유무']

    # 모델 학습
    adaboost_model.fit(X_train, y_train)

    # 모델에 대한 평가
    y_pred = adaboost_model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print(f'{i}번째 {accuracy}')

    # 모델이 이전에 저장한 모델보다 더 나은 경우 갱신
    if (accuracy > best_accuracy) and (len(df_train) >= best_length):
        best_accuracy = accuracy
        best_model = adaboost_model
        best_length = len(df_train)
        print(f"Best Model Updated! Accuracy: {best_accuracy}")

    i += 1
    train_csv_path = f'train_data_{i}.csv'
    test_csv_path = f'test_data_{i}.csv'

if best_model is not None:
    # 저장된 최적의 모델을 파일로 저장
    joblib.dump(best_model, 'ada_best_model.pkl')
    print("Best Model Saved Successfully!")
    print(best_accuracy)
else:
    print("No models met the criteria for saving.")
```

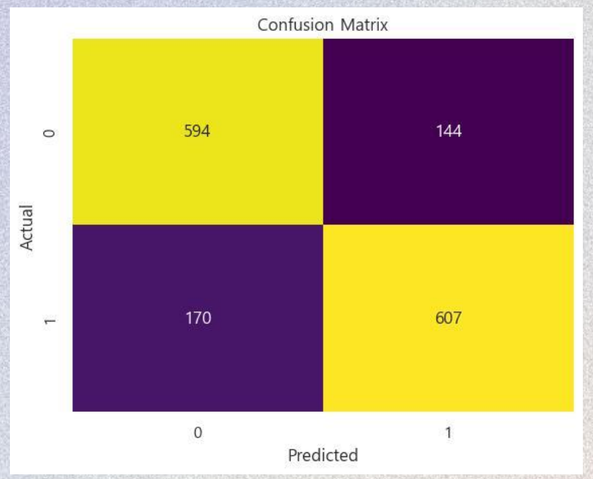
Index_머신러닝 분석/성능 평가

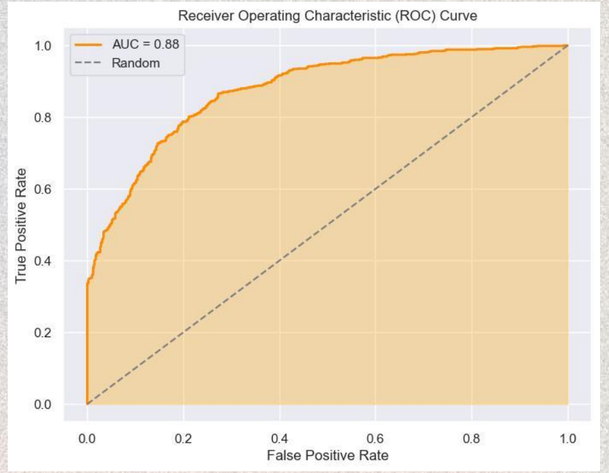
상위 10개 특성 중요도

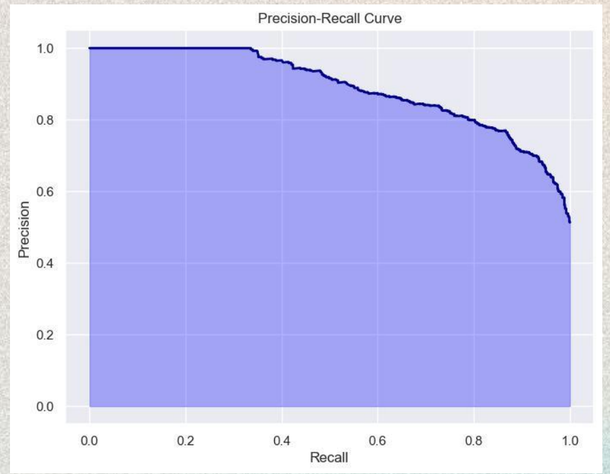


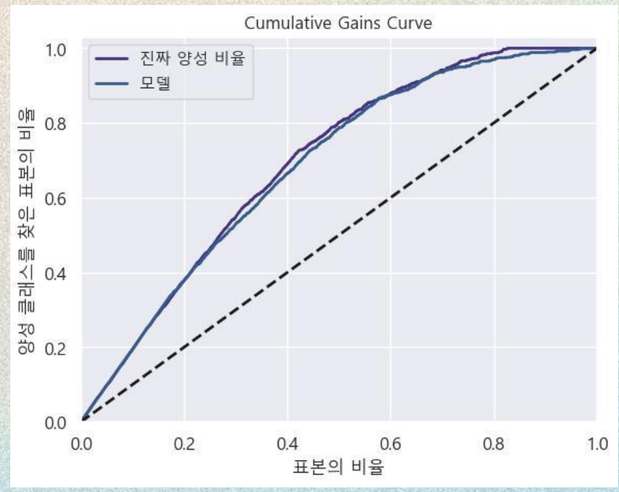
Index_머신러닝 분석/성능 평가

AdaBoost 성능평가 그래프



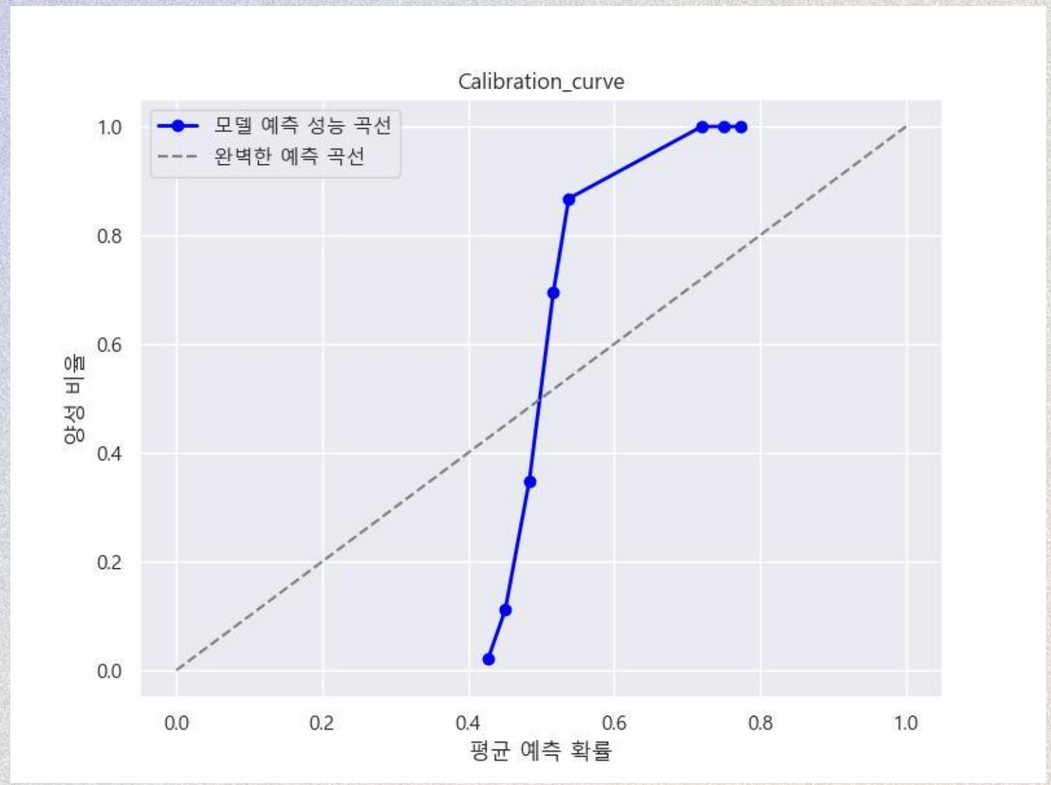


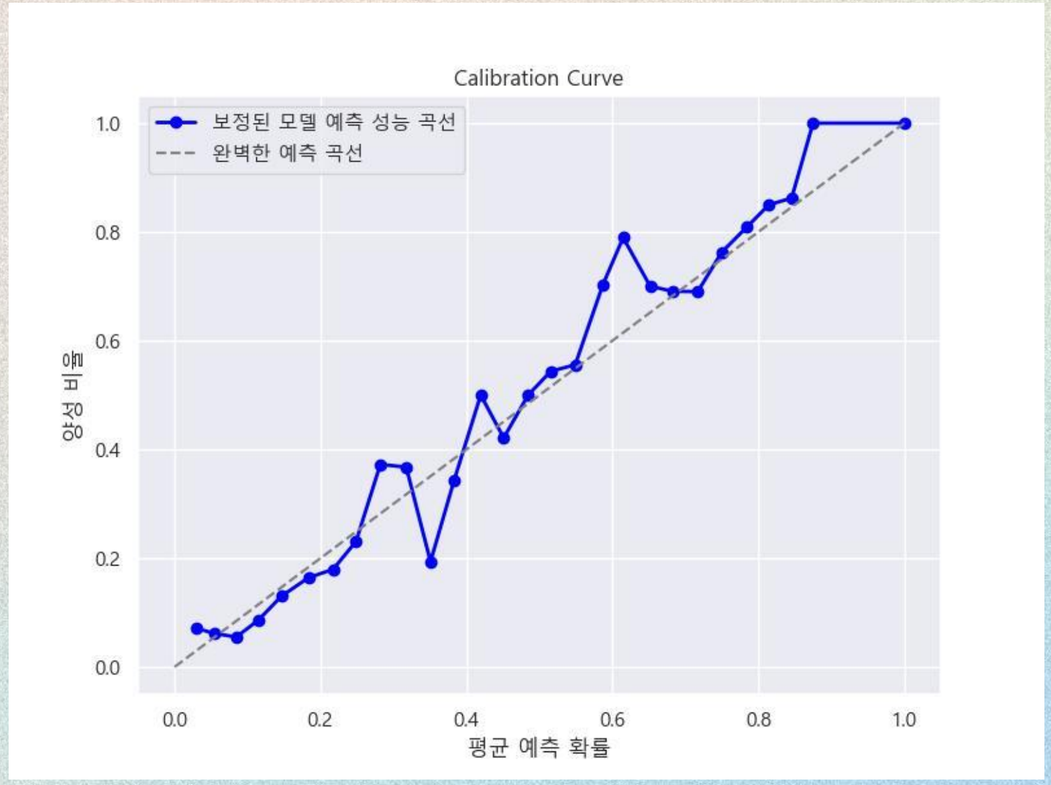




Index_머신러닝 분석/성능 평가

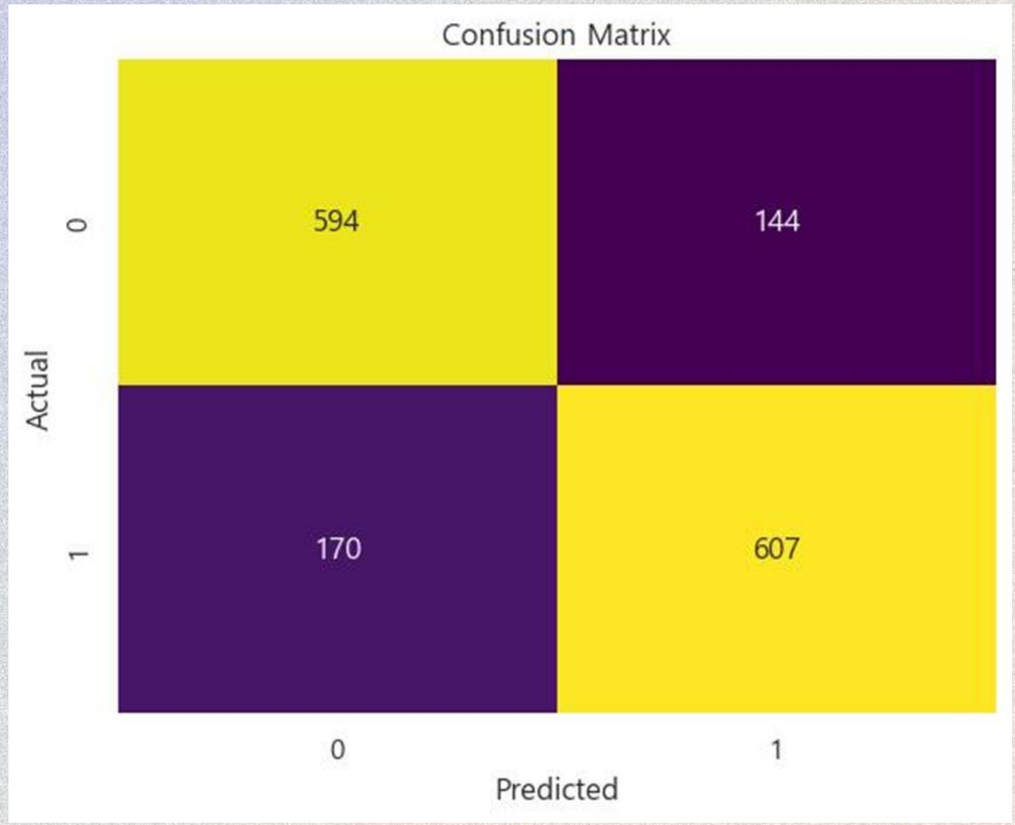
Calibration curve

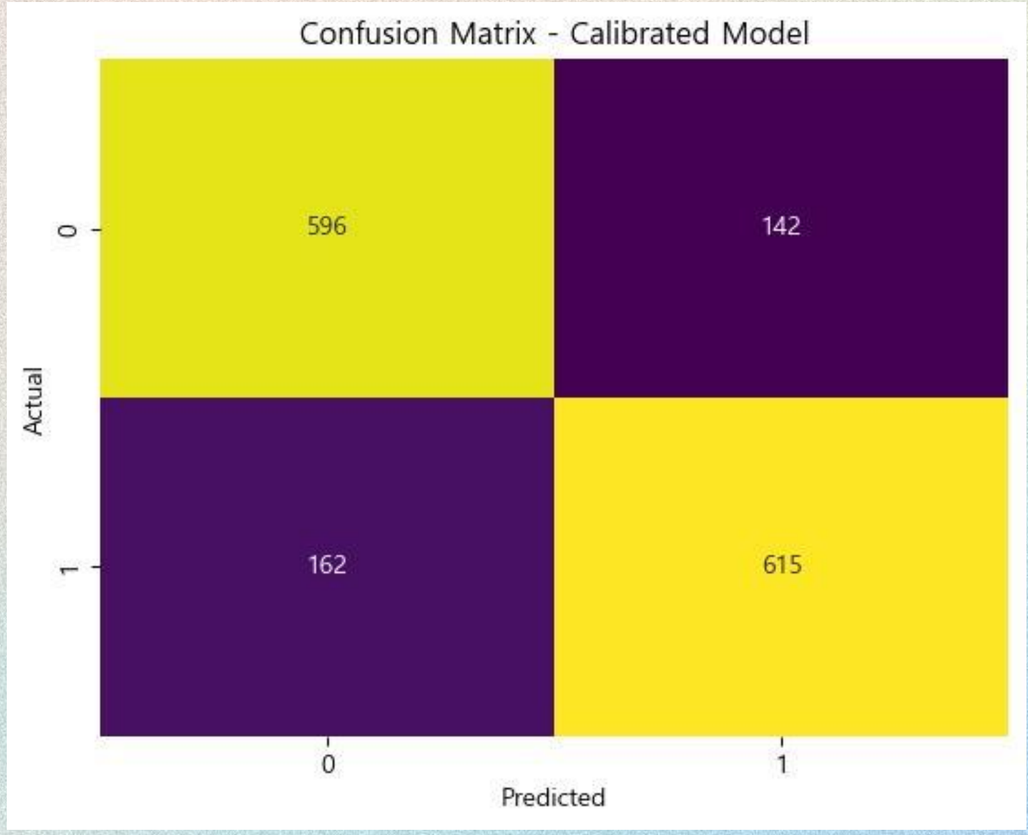




Index_머신러닝 분석/성능 평가

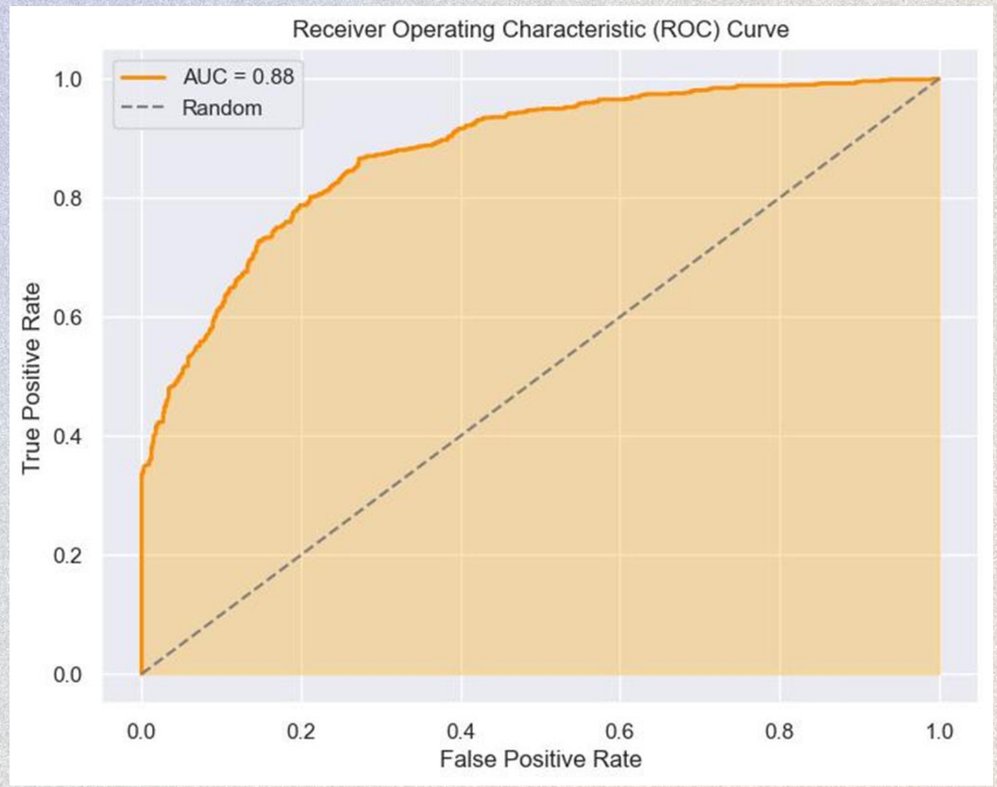
Calibration 모델 변경 후 Confusion Matrix

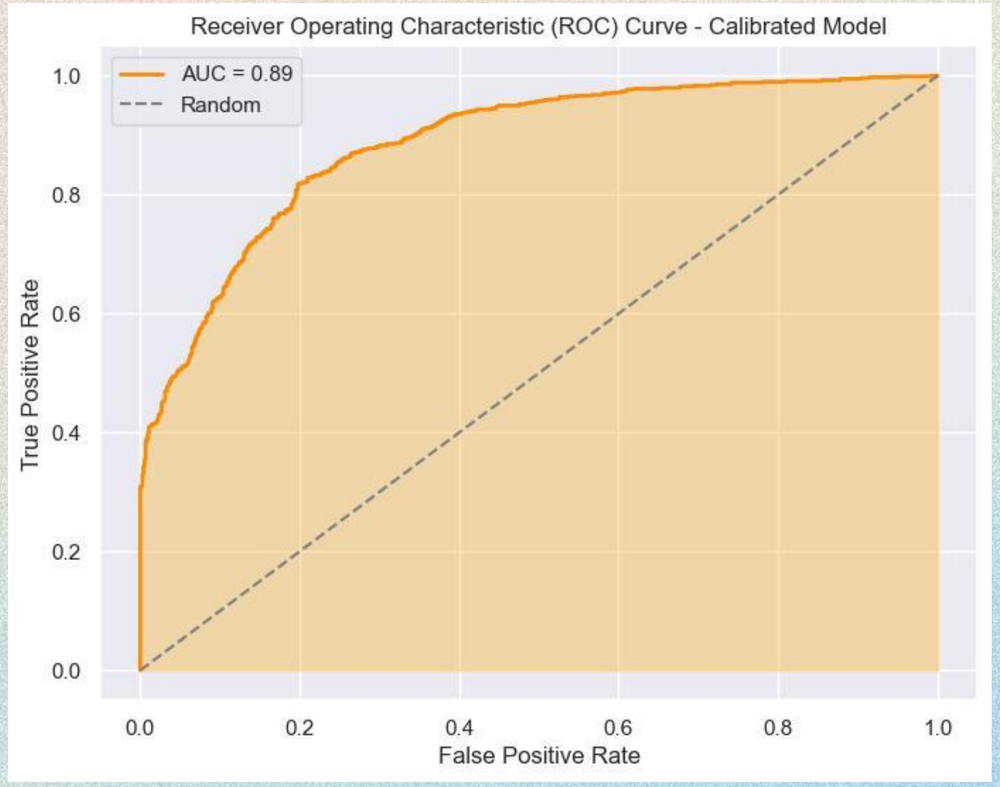




Index_머신러닝 분석/성능 평가

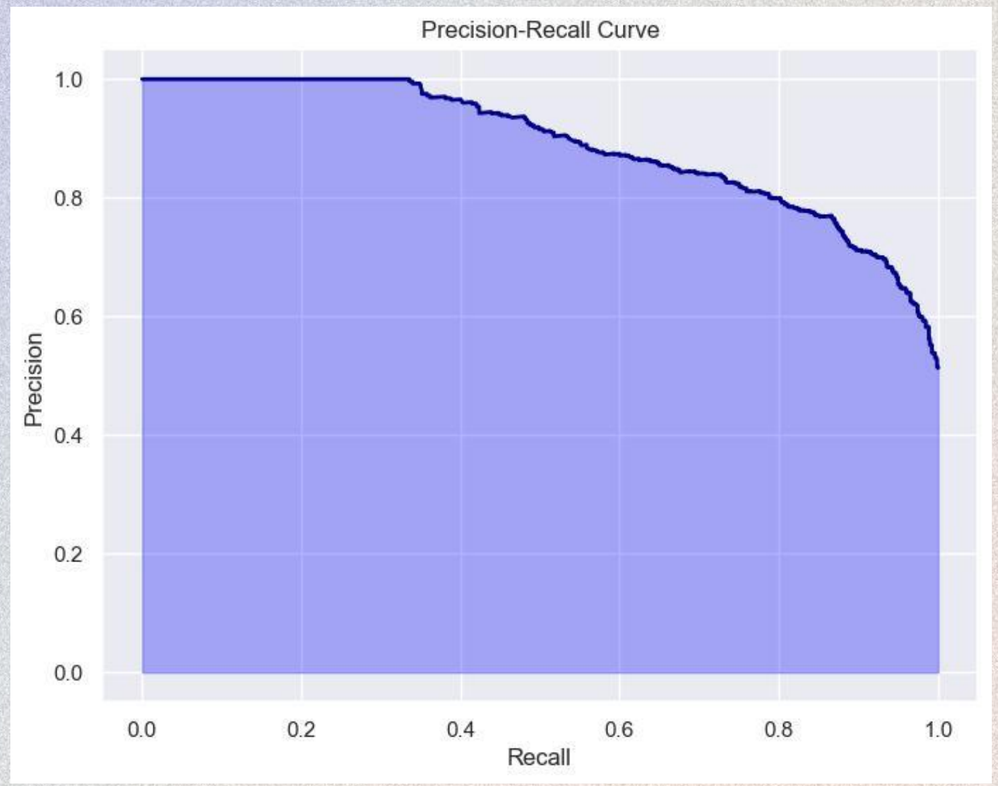
Calibration 모델 변경 후 ROC 그래프

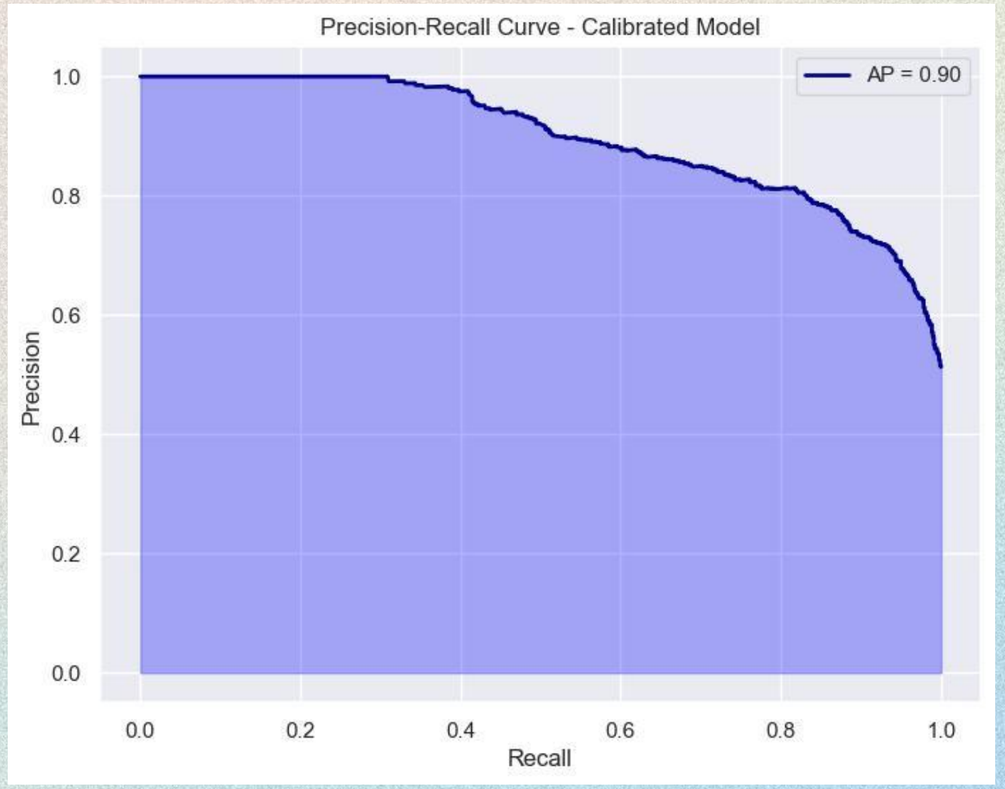




Index_머신러닝 분석/성능 평가

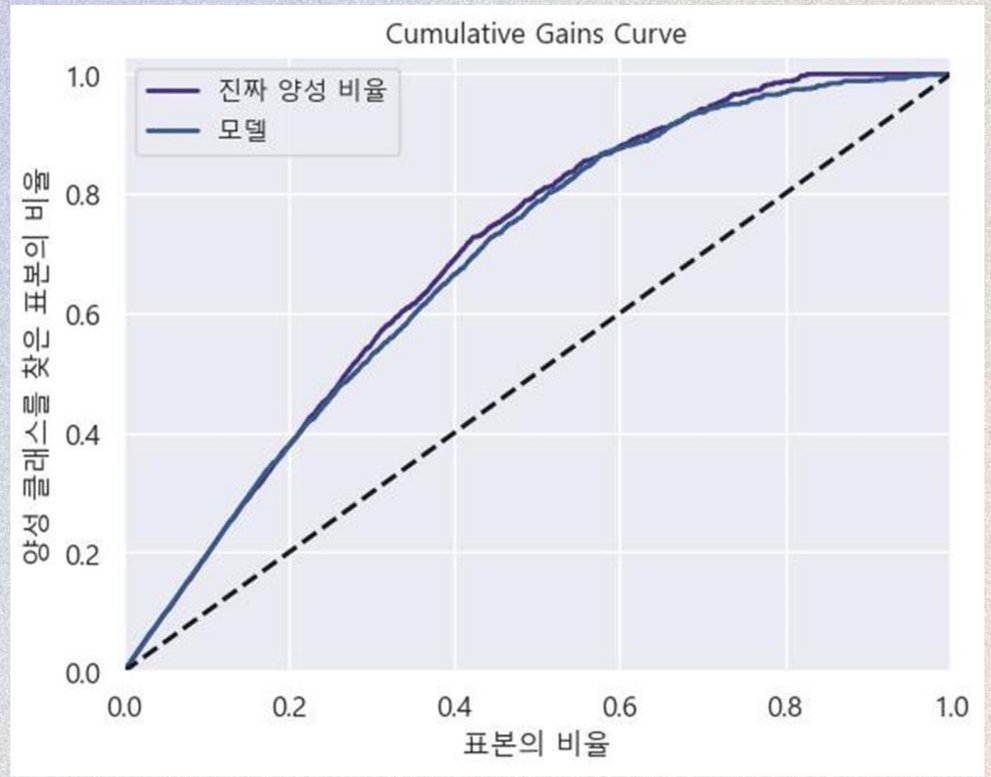
Calibration 모델 변경 후 Precision-Recall

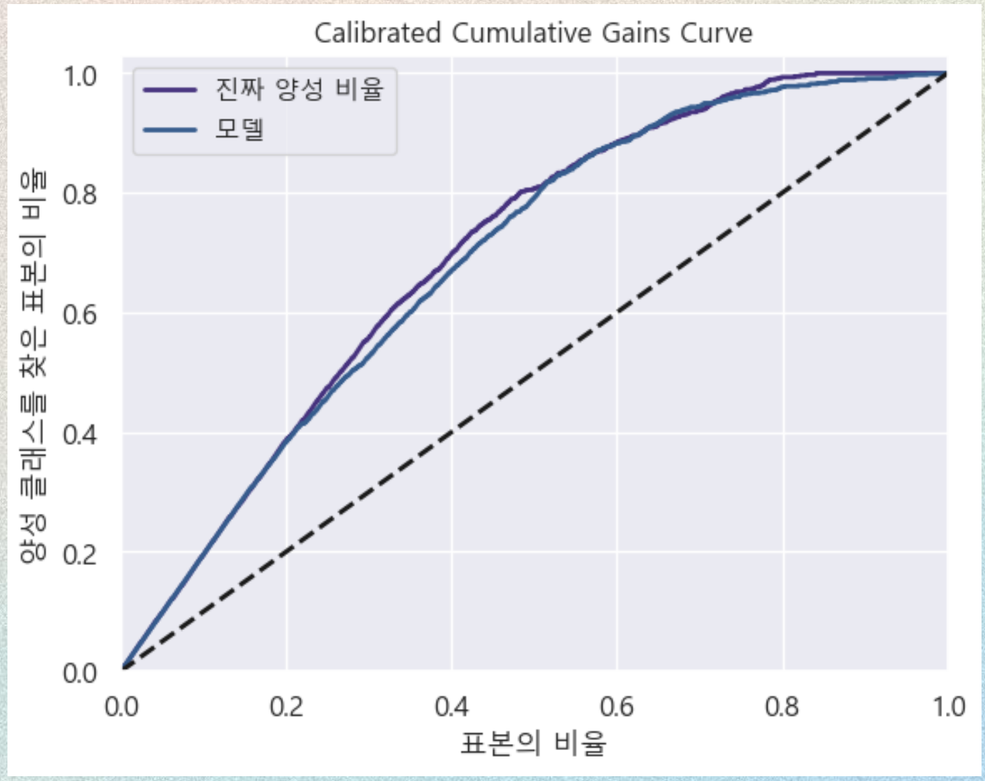




Index_머신러닝 분석/성능 평가

Calibration 모델 변경 후 Cumulative Gains Curve





Index_**머신러닝 시뮬레이션**

NHIS 2018 데이터와 유사한 데이터 확보를 위해 웹 설문 진행

- ***본 설문문항***

("고혈압 진단을 받은 적이 있으십니까?", ["예(1)", "아니오(2)"]),
("현재 임신 중이십니까?", ["예(1)", "아니오(2)"]),
("우울증 진단을 받은 적이 있으십니까?", ["예(1)", "아니오(2)"]),
("활동에 영향을 줄 정도의 과체중이십니까?", ["예(1)", "아니오(2)"]),
("알콜 중독이나 약물 중독 경험이 있으십니까?", ["예(1)", "아니오(2)"]),
("수술 후유증을 경험한 적이 있으십니까?", ["예(1)", "아니오(2)"]),
("신체 노화를 체감하십니까?", ["예(1)", "아니오(2)"]),
("쉽게 피로해지고 피곤을 잘 느끼시나요?", ["예(1)", "아니오(2)"]),
("흡연을 하십니까?", ["예(1)", "아니오(2)"]),
("음주를 하십니까?", ["예(1)", "아니오(2)"]),
("고지혈증 진단을 받은 적이 있으십니까?", ["예(1)", "아니오(2)"]),
("몸이 아플 때 병원이나 약국을 가는 편이십니까?", ["예(1)", "아니오(2)"]),
("의료비를 지출에 경제적인 부담을 느끼십니까?", ["예(1)", "아니오(2)"]),
("인터넷 등 의료 정보를 획득하기 위한 활동을 자주 하시나요?", ["예(1)", "아니오(2)"]),
("균형잡힌 식사를 하시는 편인가요?", ["예(1)", "아니오(2)"]),
("결혼여부", ["예(1)", "아니오(2)"])

- ***설문하러 가기***

Appendix_향후 개선방향

▪ 데이터 셋	당뇨 발병 주요 요인에 대한 변수 부족, 정확한 예측 어려움
▪ 모델관련	Calibration 모델로 변경 하였으나 F1점수, 정확도 등등 모든 지표가 오히려 변경 전보다 안 좋아짐. 원인 찾아야 됨
▪ 적용관련	웹이나 애플리케이션으로 배포하여 실제 활용도 높임

TEAM: 디비디비딥! DBDBdeep

Thank you

※ 프로젝트 관련 질문은 Slack message 로 보내주시기 바랍니다.