LG AIMERS

AI 온라인 해커톤

난임 환자 대상 임신 성공 여부 예측

라이언킹 | 이원우 손영진 이단비

LG AIMERS

목차

01 INTRO - 대회 개요

INTRO - 대회 개요

1. 배경

난임은 전 세계적으로 증가하는 중요한 의료 문제로, 많은 부부들이 오랜 기간 동안 신체적·정신적 부담을 겪고 있습니다. 난임 시술을 진행하는 환자들은 치료 과정에서 높은 비용과 심리적 스트레스를 경험하기 때문에, 최소한의 시술로 임신 성공 가능성을 높이는 것이 매우 중요합니다.

2. 주제

난임 환자 대상 임신 성공 여부 예측 AI모델 개발

3. 규칙

Train 데이터 : ID, 난임 환자 시술 데이터 (67개의 컬럼), 임신 성공 여부 (0: 실패, 1: 성공)
Test 데이터 : ID, 난임 환자 시술 데이터 (67개의 컬럼)
* 외부 데이터 사용 금지

02 데이터 EDA

데이터 EDA

📌 범주형 변수 처리

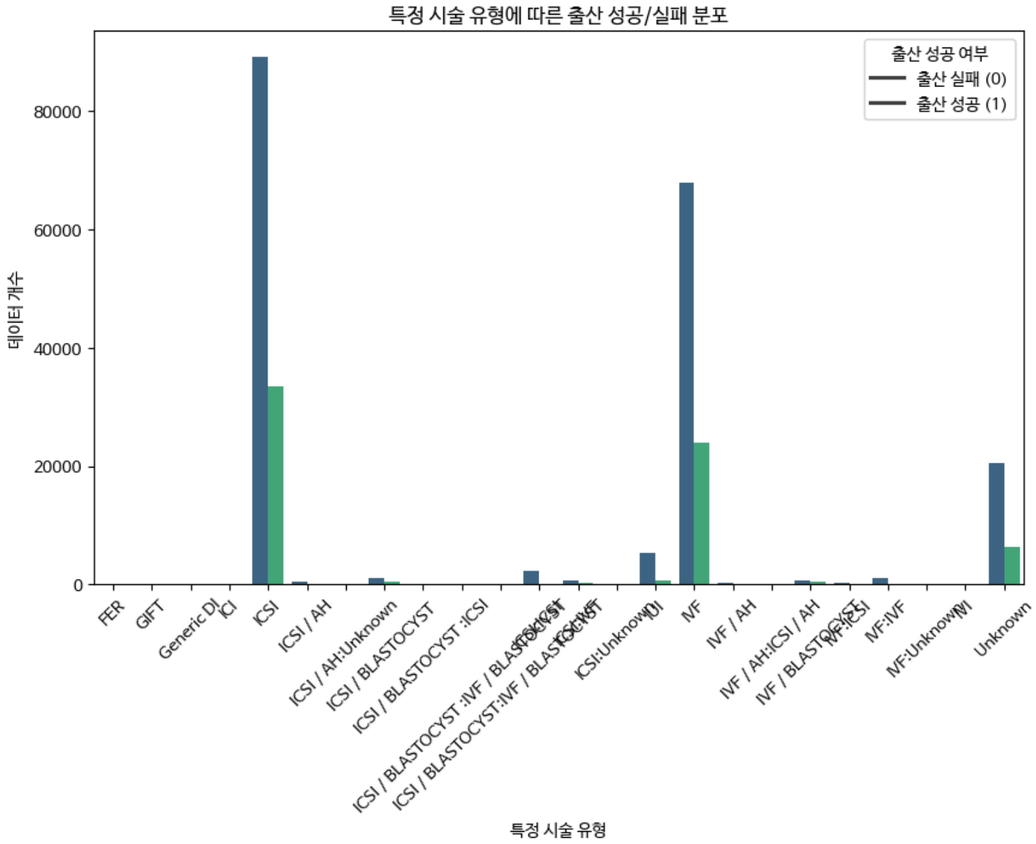
유니크 개수가 10개 이상인 Feature 다수 존재

- 대표적으로 배아 생성 이유(13개), 특정 시술 유형(24개)
- 범주 개수가 너무 많아 모델 학습 시 문제 발생 가능성 높음

3	특정 시술 유형	24	[ICSI, IVF, Unknown, IUI, IVF:IVF, IVF / BLASTOCYST, ICSI:IVF, ICSI / AH, ICSI:ICSI, IVF:ICSI, ICSI / BLASTOCYST , IVF:Unknown, ICSI:Unknown, IVF / BLASTOCYST :IVF / BLASTOCYST, IC
4	배란 유도 유형	4	[기록되지 않은 시행, 알 수 없음, 세트로타이드 (
5	배아 생성 주요 이유	13	[현재 시술용, 난자 저장용, 배아 저장용, 기증용, 현재 시술용, 기증용, 배아 저장용, 기증용, 기증용, 난자 저장용, 배아 저장용, 현재 시술용, 난자 저장용, 배아 저장용, 기 용, 연구용, 현재 시술용, 난자 저장용, 현재 시술용, 난자

데이터 EDA

1 특정 시술 유형 - 문제점



- 고윳값(Unique Values)이 24개로 매우 다양함.
- 모델이 특정 시술 유형에 과적합될 가능성이 있음.
- ICSI, IVF, Unknown이라는 특정 단어가 포함된 값이 중요한 의미를 가질 가능성이 높음.

데이터 EDA

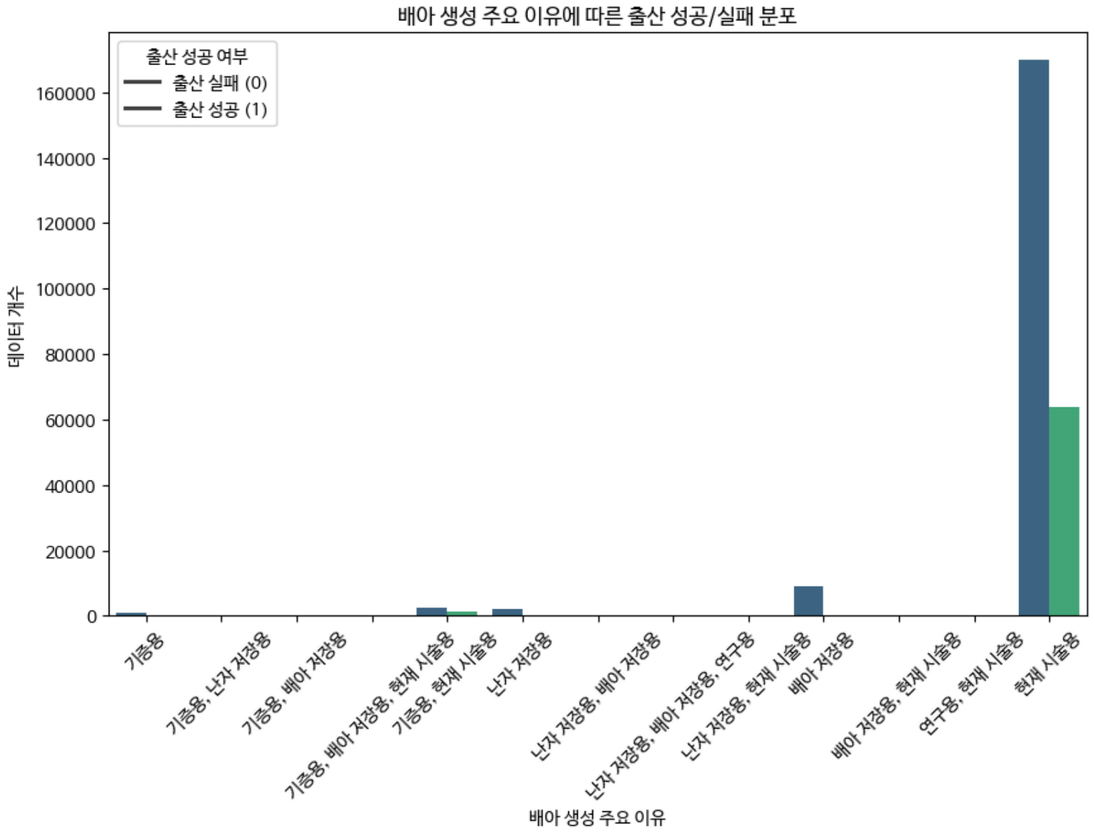
1️⃣ 특정 시술 유형 - 해결 방안

```python
# Train 데이터 처리
df_train["check_ICSI"] = df_train["특정 시술 유형"].str.contains("ICSI", na=False).astype(int)
df_train["check_IVF"] = df_train["특정 시술 유형"].str.contains("IVF", na=False).astype(int)
df_train["check_Unknown"] = (df_train["특정 시술 유형"].str.strip() == "Unknown").astype(int)

# Test 데이터 처리
df_test["check_ICSI"] = df_test["특정 시술 유형"].str.contains("ICSI", na=False).astype(int)
df_test["check_IVF"] = df_test["특정 시술 유형"].str.contains("IVF", na=False).astype(int)
df_test["check_Unknown"] = (df_test["특정 시술 유형"].str.strip() == "Unknown").astype(int)

# ✅ ID 컬럼 제거
df_train = df_train.drop(columns=["특정 시술 유형"])
df_test = df_test.drop(columns=["특정 시술 유형"])
```

- 주요 시술인 ICSI, IVF, Unknown 포함 여부를 새로운 파생 변수로 생성.
- 생성된 파생 변수
  - check_ICSI: 특정 시술 유형에 "ICSI"가 포함되면 1, 아니면 0
  - check_IVF: 특정 시술 유형에 "IVF"가 포함되면 1, 아니면 0
  - check_Unknown: 특정 시술 유형에 "Unknown"이 포함되면 1, 아니면 0

데이터 EDA

2 배아 생성 이유 - 문제점



- 고윳값(Unique Values)이 13개로, 모델이 모든 범주를 학습하기 어려움.
- 기존 데이터를 살펴보니, 현재 시술용, 배아 저장용, 기증용, 난자 저장용, 연구용 5가지로 그룹화 가능.

데이터 EDA

2 배아 생성 이유 - 해결 방안

```python
# ✅ One-Hot Encoding을 위한 카테고리 리스트
categories = ["현재 시술용", "배아 저장용", "기증용", "난자 저장용", "연구용"]

# ✅ 새로운 컬럼 생성 (각 카테고리가 존재하면 1, 없으면 0)
for cat in categories:
    df_train[cat] = df_train["배아 생성 주요 이유"].apply(lambda x: 1 if cat in str(x) else 0)

# ✅ 새로운 컬럼 생성 (각 카테고리가 존재하면 1, 없으면 0)
for cat in categories:
    df_test[cat] = df_test["배아 생성 주요 이유"].apply(lambda x: 1 if cat in str(x) else 0)

# ✅ ID 컬럼 제거
df_train = df_train.drop(columns=["배아 생성 주요 이유"])
df_test = df_test.drop(columns=["배아 생성 주요 이유"])
```

- 기존 13개 값을 5개 주요 그룹으로 변환하여 범주 개수를 줄임.
- One-Hot Encoding 적용

03 데이터 전처리

데이터 전처리 - 라벨 인코딩

STEP 1: 범주형 변수 리스트 확인

STEP 2: Label Encoding 적용

STEP 3: Test 데이터 변환 (Train에 없는 값 처리)

✔ Train에 없던 값 처리 방법
Test 데이터의 고유값(test_values)을 가져옴
Train에서 학습한 고유값(known_classes)과 비교
Train에 없는 값이면 -1로 변환
Train에 존재하는 값이면 Label Encoding 적용

```python
import numpy as np
from sklearn.preprocessing import LabelEncoder

# ✅ 범주형 컬럼 리스트 확인
cat_cols_train = df_train.select_dtypes(include=["object"]).columns
cat_cols_test = df_test.select_dtypes(include=["object"]).columns

# Train과 Test의 공통된 범주형 컬럼만 선택
common_cat_cols = list(set(cat_cols_train) & set(cat_cols_test))

# ✅ 안전한 Label Encoding 적용
for col in common_cat_cols:
    le = LabelEncoder()

    # Train 데이터 학습
    df_train[col] = le.fit_transform(df_train[col])

    # Test 데이터 변환 (Train에 없는 값은 -1로 처리)
    test_values = np.array(df_test[col])  # 원본 테스트 값 저장
    known_classes = set(le.classes_)  # 훈련 데이터에서 학습한 클래스 저장

    # 안전한 변환 수행
    test_encoded = np.array([
        le.transform([value])[0] if value in known_classes else -1
        for value in test_values
    ])

    # 변환된 값을 Test 데이터에 저장
    df_test[col] = test_encoded
```

데이터 전처리 - 피처 중요도

STEP 1 LGBM Classifier 모델로부터 feature importance 추출

```
import pandas as pd

# LGBMClassifier 모델로부터 피처 임포턴스 추출
feature_importances = lgbm_model.feature_importances_

# 피처 이름과 임포턴스를 데이터 프레임으로 생성
df_feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Importance': feature_importances
})

# 중요도 기준 내림차순으로 정렬
df_feature_importance = df_feature_importance.sort_values(by='Importance', ascending=False)
```

피쳐 중요도 상위 10개 결과

> 배아 이식 경과일, 이식된 배아 수, 시술 당시 나이, 총 생성 배아 수, 저장된 배아 수, 수집된 신선 난자 수, IVF 시술 횟수, 총 임신 횟수, 미세주입 배아 이식 수 ...

데이터 전처리 - 파생변수 생성

STEP 2 피쳐 중요도에 따른 “배아, 미세주입” 과 관련한 새로운 파생 변수 생성

1. 배아 생성 효율: 총 생성 배아 수 / (수집된 신선 난자 수)
2. 배아 이식 효율: 이식된 배아 수 / (총 생성 배아 수)
3. 미세주입 효율: 미세주입에서 생성된 배아 수 / (미세주입된 난자 수)
4. 미세주입 후 저장 비율: 미세주입 후 저장된 배아 수 / (미세주입에서 생성된 배아 수)
5. 저장 배아 효율: 저장된 배아 수 / (총 생성 배아 수)
6. 해동 배아 비율: 해동된 배아 수 / (저장된 배아 수)
7. 파트너 정자 사용 비율: 파트너 정자와 혼합된 난자 수 / (혼합된 난자 수)
8. 배아 이식 경과일 구간화: '배아 이식 경과일'을 quantile 기반으로 구간 나누기

04 모델학습 - LGBM 활용

LGBM Classifier 를 통한 모델 학습

- Left-wise tree growth

  트리를 수직으로 확장하여
  비교적 낮은 Loss 달성

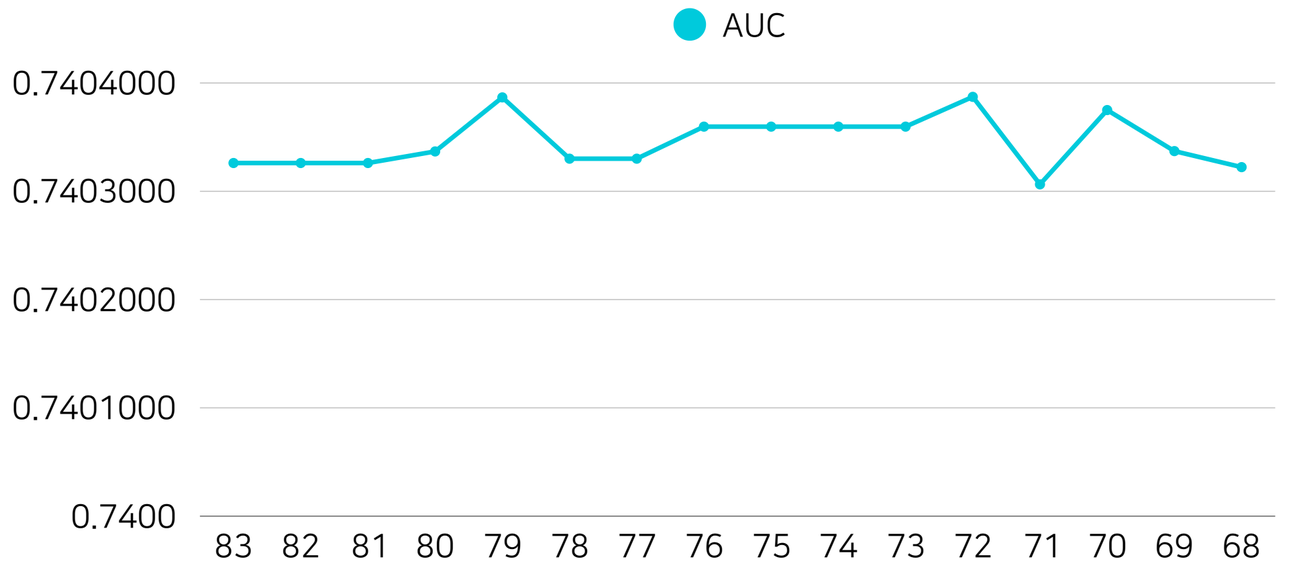
- Low Computation Cost

  효율적인 메모리 관리
  대용량 데이터 처리 능력

Leaf-wise tree growth

모델 학습 - 72개의 피처 선택

- 학습에 영향을 주지 않는 피처를 제거하여 모델 성능을 향상
- 불필요한 변수 삭제 → 데이터 노이즈 감소 → 학습 최적화



72개의 피쳐 최종 선택
AUC : 0.74038 ✅

하이퍼 파라미터 최적화 - Optuna

- Automation (자동화)

  최적의 하이퍼 파라미터 자동 탐색

- Efficiency (효율성)

  TPE 알고리즘을 통한 효율적인 탐색

- Flexibility (유연성)

  광범위한 하이퍼 파라미터 지원
  (연속/불연속, 범주형 변수 등)

하이퍼 파라미터 최적화 - Optuna

Optuna를 통해 찾은
최적의 하이퍼 파라미터

```
(
 objective='binary',
 metric = 'auc',
 boosting_type = 'gbdt',
 random_state=42,
 scale_pos_weight = 1.2469509553827316,
 learning_rate = 0.02149524140540219,
 num_leaves = 16,
 max_depth = 13,
 min_child_samples = 27,
 subsample = 0.6808936174607194,
 colsample_bytree = 0.5140506756742775,
 reg_alpha = 9.081327528794018,
 reg_lambda = 8.9592839180652,
 n_estimators = 1132,
 verbosity = -1
 )
```

테스트 데이터 예측

최종 학습된 모델을 활용한
Test 데이터의 임신 성공 확률 예측
[클래스 '1'의 확률]

	prediction_score
0	0.001343
1	0.001307
2	0.182194
3	0.130158
4	0.538235
...	...
90062	0.001602
90063	0.345649
90064	0.538853
90065	0.237165
90066	0.001408

90067 rows × 1 columns

감사합니다

라이언킹 | 이원우 손영진 이단비