

## 1. 분석 목적

Diabetes Data Set으로 로지스틱 회귀분석(Logistic Regression) 및 랜덤포레스트(Random Forest) 분류 모델을 구축하고, 정확도(Accuracy), ROC-AUC, F1-Score 등의 평가 지표를 통해 두 모델의 성능을 비교함.

## 2. 데이터 의미 파악

### 1) Pregnancies : Number of times pregnant

- 임신성 당뇨(Gestational Diabetes)와 관련됨.
- 임신 중 당대사의 생리학적 변화가 일어나 당뇨가 생긴 경우로, 임신성 당뇨병 환자의 50% 이상은 20년 이내 제 2형 당뇨병이 발병함.
- 임신 횟수와 임신성 당뇨의 정확한 상관관계는 밝혀진 바가 없으나, 이전에 임신성 당뇨병이 있었거나 4kg 이상의 아기를 분만한 경우 고위험군에 속하므로 간접적인 관계가 있음.

### 2) Glucose : Plasma glucose concentration in a 2 hour oral glucose tolerance test(OGTT)

- 경구 포도당 부하 검사 : 당뇨병이나 내당능장애를 진단하며, 췌장 베타세포의 기능을 측정하기 위한 검사
- 공복 혈당 측정 및 헤모글로빈 A1c보다 당뇨병의 진단에 예민하나 검사 과정의 번거로움으로 인해 일상적인 검사로 진행하지는 않음.
- 공복시의 혈당치가 126mg/dL 또는 당화혈색소가 6.5%를 넘어 당뇨병이 의심될 때 확정 진단을 위해 실시
- 검사 결과 해석 기준
  - >> 정상 : 검사 전의 공복 혈당치가 100mg/dL 미만이면서 검사 2시간 후 혈당치가 140mg/dL 미만
  - >> 당뇨병 : 공복 혈당치가 126mg/dL 이상 또는 경구 당부하 검사 2시간 후 혈당치가 200mg/dL 이상
  - >> 전당뇨병 (당뇨병 발생 고위험군) : 공복 혈당치가 100~125mg/dL 또는 경구 당부하 검사 2시간 후 혈당치가 140~199mg/dL

### 3) Insulin : 2-Hour serum insulin (mu U/ml)

- OGTT 과정 중 포도당을 마신 후 2시간째 혈액에서 측정한 인슐린 농도
- 75g 포도당 복용 후 인슐린 기준치 : 공복(10 이하), 20분 후(30~80), 60분 후(25~70), 90분 후(20~60), 120분 후(20~50), 180분 후(6~15)
- 정상적으로는 혈장 인슐린이 경구 포도당 부하 검사(OGTT) 시작 30분 뒤 급격히 상승했다가 2시간 뒤면 원래대로 돌아옴.
- 검사 결과 해석 기준(2시간 인슐린 수치) : 정상(20~50), 경계형/저항성(60-100), 고도 인슐린 저항성/ 당뇨 전단계(100-200+), 병적 고인슐린혈증(>200)
- 검사 결과 인슐린 수치가 높게 측정된다면 간질환, 비만, 인슐린 자가면역질환, 쿠싱증후군, 인슐린종 등을 의심할 수 있음. 인슐린 수치가 낮다면 인슐린 생산량 자체가 적은 1형 당뇨병이나 뇌하수체기능저하증, 부신 부전을 생각해 볼 수 있음.
- 질병의 정확한 진단을 위해서는 혈당, C-peptide, 당화혈색소 등 다양한 수치를 확인하여 종합적으로 판단함.

### 4) BloodPressure : Diastolic blood pressure (mm Hg)

- 당뇨와 고혈압은 서로의 발생 위험을 높임.
- 혈압이 상승하면 인슐린 저항성이 심해져 췌장이 손상을 받아 인슐린 분비능이 저하되어 당뇨 발생 확률 증가
- 혈압 상승과 관련된 레닌-안지오텐신 시스템이 활성화되면 안지오텐신2가 상승해 당뇨 발생 증가

- 혈압 상승 시 활성산소가 증가하고 이로 인해 산화스트레스를 유발해 췌장의 베타세포 기능을 감소시켜 인슐린 분비에 이상이 생겨 당뇨병 발생 확률 증가
- 실제로 연구 결과 정상 혈압을 보인 사람(<120/80 mmHg)에 비해, 고혈압 전단계인 경우(120~139/80~89 mmHg)는 당뇨병 발생 위험이 23% 높았고, 고혈압 1단계(140~159/90~99 mmHg)에서는 26%, 고혈압 2단계( $\geq 160/100$  mmHg)에서는 그 위험도가 60%나 높았음.
- 당뇨병 환자는 고혈당으로 인해 콩팥 기능이 손상되어 혈압이 상승하는 등 고혈압 발생 위험이 일반인보다 약 2배 높음.

#### 5) SkinThickness : Triceps skin fold thickness (mm)

- 삼두근 피부주름 두께 : 삼두근 부위의 피부와 그 아래 피하 지방층의 두께를 측정하는 인체 계측법으로, 개인의 체지방량을 간접적으로 평가하는 데 사용됨.
- 체지방이 늘어나면, 즉 비만이면 지방분해가 증가되어 유리지방산이 직접 간문맥을 통해 간으로 유입되어 인슐린의 작용을 감소시키며 인슐린 저항성이 높아짐.
- 또한 비만일수록 근육에서 GLUT4에 매개되는 포도당 유입이 감소되어 인슐린저항성이 발생함.
- 검사결과 해석 기준(정상 범위) : 성인 여성(16~25), 성인 남성(10~20), 소아/ 청소년(10~22)

#### 6) BMI : Body mass index (weight in kg/(height in m)<sup>2</sup>)

- BMI는 키와 체중만을 이용해 비만을 추정하는 지표이므로 BMI가 높다고 반드시 비만이라고 할 수는 없지만 가능성은 높아짐.
- 체지방률 등의 추가 지표와 함께 비만 판단에 활용함.
- 검사결과 해석 기준(WHO기준) : 저체중(<18.5), 정상 체중(18.5~24.9), 과체중(25.0~29.9), 비만 1단계(30.0~34.9), 비만 2단계(35.0~39.9), 고도비만( $\geq 40.0$ )

#### 7) Diabetes Pedigree Function

- 가족력 기반 당뇨 발병 가능성 지수로, 한 개인의 유전적 요인이 당뇨병 발병에 미치는 상대적 위험도를 수치로 표현한 값
- 가족 중 당뇨병 환자가 많을수록 당뇨병 환자가 많다는 의미
- 일반인에 비해 가족 내에서 당뇨병이 있는 경우 2형당뇨병의 발생 위험은 일란성 쌍생아는 10배, 직계가족은 3.5배 정도 높음. 하지만 유전적 소인이 있다고 해서 전부 당뇨병 환자가 되는 것은 아니며, 여러 가지 환경적 요인이 함께 작용해 당뇨병이 생김.

#### 8) Age: Age (years)

- 성인형 당뇨병은 40대 이후에 많아지기 시작함.
- 나이가 들면 신체의 모든 세포 기능이 점차 떨어지고 포도당을 포함한 에너지 대사 능력이 저하되어 혈당을 조절하는 능력이 약해져 당뇨병 위험이 높아짐.

#### cf. 당뇨병의 정의

- 인슐린의 분비량이 부족하거나 제대로 기능을 하지 않아 발생하는 대사질환으로, 혈액 속의 포도당 수치가 정상인보다 높고 소변으로 포도당이 배출된다고 해서 이름 붙여진 병
- 정상 혈당 기준  
>> 공복혈당 : 100 mg/dl 미만

>> 경구 당부하 검사 후 2시간 혈당 : 140 mg/dl 미만

>> 당화혈색소 : 환자의 건강상태에 따라 조절목표가 달라질 수 있으나, 보통 6.5~7% 미만으로 유지 권고

### 3. 기본 데이터 분석

#### 1) 기본 데이터셋 분석

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  ---                ---
0   Pregnancies            768 non-null    int64
1   Glucose                768 non-null    int64
2   BloodPressure          768 non-null    int64
3   SkinThickness          768 non-null    int64
4   Insulin                768 non-null    int64
5   BMI                   768 non-null    float64
6   DiabetesPedigreeFunction 768 non-null    float64
7   Age                   768 non-null    int64
8   Outcome                768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

- 총 768행 존재

- Column 중 BMI와 DiabetesPedigreeFunction은 부동소수점 자료형이고, 나머지는 정수형 자료형임. 모두 수치형 자료들이므로 문자형 자료형을 위한 분류 작업은 따로 안해도 됨.

- 'Outcome' 변수는 이미 0(정상)과 1(당뇨병)으로 인코딩된 이진형 변수이므로, 별도의 인코딩 과정은 생략 가능.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
print(df['Outcome'].value_counts())
count_1 = (df['Outcome'] == 1).sum()
print("값이 1인 개수:", count_1)
```

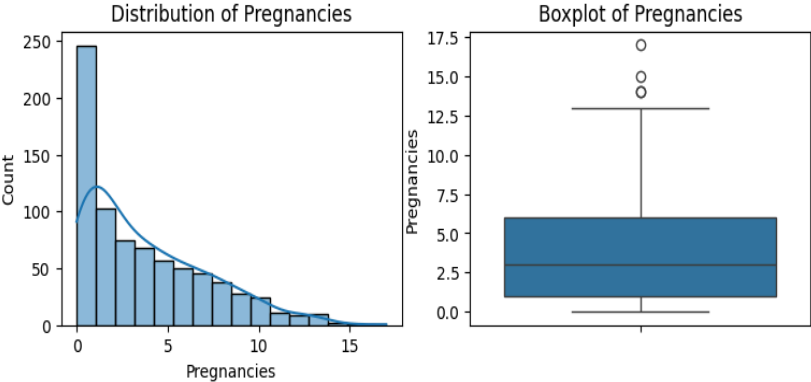
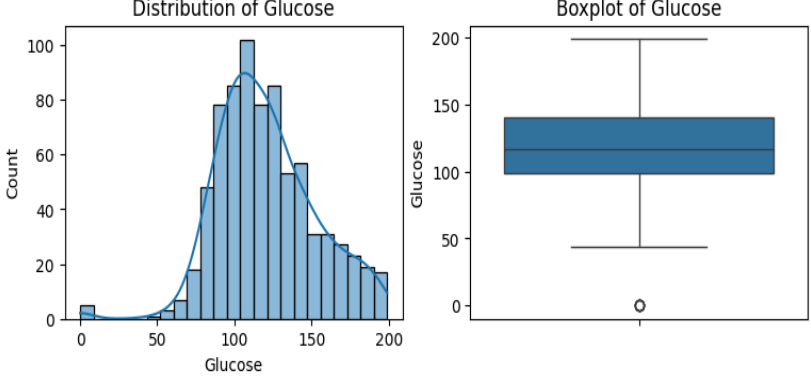
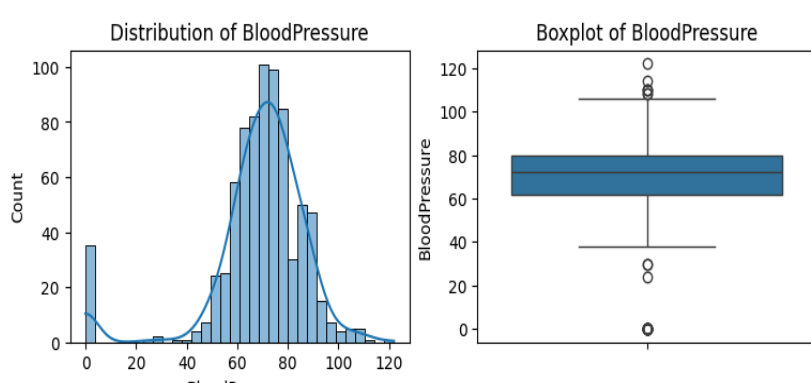
```
Outcome
0    500
1    268
Name: count, dtype: int64
값이 1인 개수: 268
```

- 'Outcome'은 0(정상)이 1(당뇨병)보다 많음.

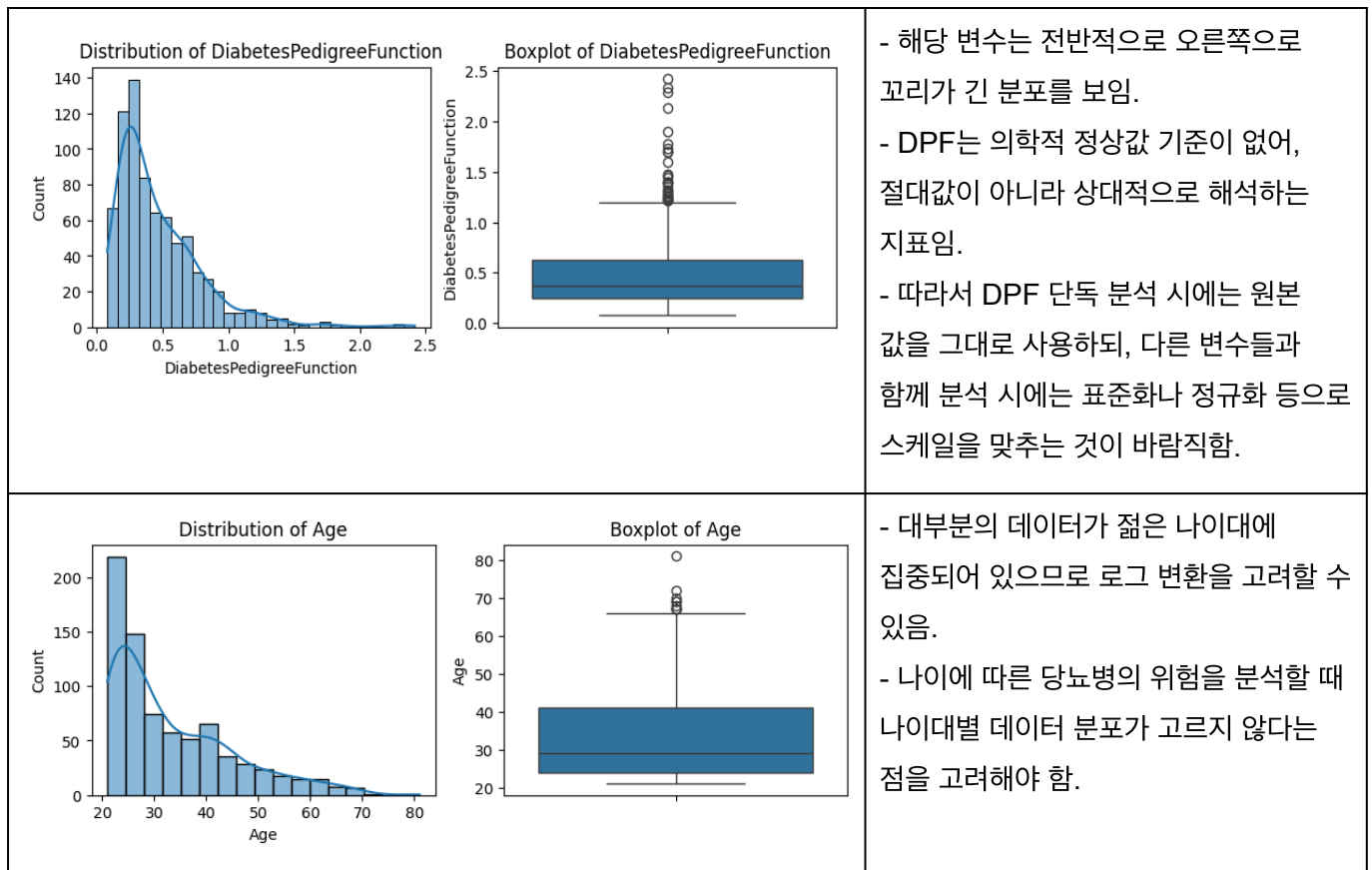
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

2) 각 변수에 대해 히스토그램과 박스플롯을 생성해 데이터의 중심 경향성, 분산, 왜도, 이상치 등을 파악.

‘Outcome’은 종속변수이므로 제외함.

	<ul style="list-style-type: none"> <li>- 해당 변수는 오른쪽으로 꼬리가 긴 분포를 보임. 정규분포는 아니지만 ‘임신 횟수’라는 변수의 특성상 이러한 치우침이 자연스러운 것이므로 전처리는 불필요함.</li> <li>- 명확하게 이산적인 값을 가지고 있으며, 중앙값은 약 3회 정도임.</li> </ul>
 <pre>count_7 = (df['Glucose']==0).sum() print("값이 0인 개수:", count_7)</pre> <p>값이 0인 개수: 5</p>	<ul style="list-style-type: none"> <li>- 해당 변수는 오른쪽으로 약간 꼬리가 긴 분포를 보이긴 하나 전반적으로 정규분포에 가까운 형태를 띠.</li> <li>- 값은 주로 100-150 사이에 많이 분포하고 있음. Glucose의 정상값이 140미만임을 고려했을 때, Outcome에서 0(정상)의 값이 더 많은 현 상황에서는 정상적인 분포임.</li> <li>- 값이 0인 이상치가 존재하며, 0은 생리학적으로 있을 수 없으므로 결측값을 0으로 대체한 것으로 판단됨. 해당 값의 수가 5개로 적으므로 제외하고 분석을 진행하는 것이 바람직함.</li> </ul>
 <pre>count_2 = (df['BloodPressure'] == 0).sum() print("값이 0인 개수:", count_2) count_3 = (df['BloodPressure'] &gt; 100).sum() print("BloodPressure가 100 초과인 개수: ",count_3)</pre> <p>값이 0인 개수: 35 BloodPressure가 100 초과인 개수: 13</p>	<ul style="list-style-type: none"> <li>- 해당 변수는 전반적으로 정규분포에 가까운 분포를 보이며, 60-80 사이에 위치해 정상적인 이완기 혈압을 보임.</li> <li>- Glucose와 마찬가지로 0은 생리학적으로 있을 수 없는 이상치임. 이때 해당값의 개수가 35개로 많지는 않으나 다른 변수에서 제거하는 값들이 많아질 경우 데이터의 수가 부족해질 수 있음.</li> <li>- 따라서 이 경우 정규분포에 가까우므로 0이 아닌 값들의 평균으로 대체하는 것이 바람직함.</li> <li>- 값이 100 이상인 이상치들이 있으며,</li> </ul>

	<p>이는 고혈압인 환자들임. 고혈압과 당뇨의 상관관계를 보기 위해서는 해당 데이터들은 전처리를 하지 않는 것이 낫다고 판단됨.</p>
<div data-bbox="132 405 940 712"> </div> <div data-bbox="132 728 940 817"> <pre>mask1 = (df['SkinThickness'] == 0)   (df['SkinThickness'] &gt; 60) count_4 = mask1.sum() print("값이 0이거나 60 초과인 것의 개수:", count_4)</pre> </div> <div data-bbox="132 840 510 869"> <p>값이 0이거나 60 초과인 것의 개수: 229</p> </div>	<ul style="list-style-type: none"> <li>- 값이 0인 이상치가 많음. 생리학적으로 0은 가능하지 않으므로 누락된 값을 0으로 대체한 것일 확률이 높음.</li> <li>- 값이 60 초과인 것들 역시 생리학적으로 비정상적인 값들임.</li> <li>- 해당 값의 비중이 높기 때문에, 이들을 제거하기보다는 분포가 살짝 오른쪽 꼬리가 길다는 것을 고려하여 중앙값으로 대체하는 것이 바람직함.</li> </ul>
<div data-bbox="132 916 940 1223"> </div> <div data-bbox="132 1238 778 1328"> <pre>mask2 = (df['Insulin'] == 0)   (df['Insulin'] &gt;= 300) count_5 = mask2.sum() print("값이 0이거나 300 이상인 개수:", count_5)</pre> </div> <div data-bbox="132 1350 474 1379"> <p>값이 0이거나 300 이상인 개수: 412</p> </div>	<ul style="list-style-type: none"> <li>- 값이 0이거나 300 이상인 경우 생리적으로 보기 어려운 극단값으로 간주할 수 있음.</li> <li>- 해당 값의 비중이 높으므로 SkinThickness와 마찬가지로 이들을 제거하기보다는 분포가 살짝 오른쪽 꼬리가 길다는 것을 고려하여 중앙값으로 대체하는 것이 바람직함.</li> </ul>
<div data-bbox="132 1449 940 1778"> </div> <div data-bbox="132 1794 778 1883"> <pre>mask3 = (df['BMI'] == 0)   (df['BMI'] &gt; 50) count_6 = mask3.sum() print("값이 0이거나 50 초과인 것의 개수:", count_6)</pre> </div> <div data-bbox="132 1906 550 1935"> <p>값이 0이거나 50 초과인 것의 개수: 19</p> </div>	<ul style="list-style-type: none"> <li>- 해당 변수는 전반적으로 정규분포에 가까운 분포를 따름.</li> <li>- 값이 0이거나 50 초과인 경우는 생리적으로 보기 어려운 이상값으로 간주될 수 있음.</li> <li>- 해당값의 수가 19개로 적으므로 이는 제거하고 분석을 진행하는 것이 좋을 것이라 판단됨.</li> </ul>



### 3) 데이터 전처리(결측치, 이상치 처리) : 변수들 간의 상관관계 분석 전 데이터 전처리로, 상관관계수(Pearson correlation) 계산 시에 scale은 영향을 주지 않으므로 우선 결측치, 이상치 처리만 수행함.

```
# glucose 전처리
df = df.drop(df[df['Glucose'] == 0].index)

# bloodpressure 전처리
mean_bp = df.loc[df['BloodPressure'] != 0, 'BloodPressure'].mean()
df['BloodPressure'] = df['BloodPressure'].replace(0, mean_bp)

# skintickness 전처리
valid_st = df.loc[(df['SkinThickness'] != 0) & (df['SkinThickness'] <= 60), 'SkinThickness']
median_st = valid_st.median()
df['SkinThickness'] = df['SkinThickness'].mask(
    (df['SkinThickness'] == 0) | (df['SkinThickness'] > 60), median_st)

# Insulin 전처리
valid_insulin = df.loc[(df['Insulin'] != 0) & (df['Insulin'] < 300), 'Insulin']
median_insulin = valid_insulin.median()
df['Insulin'] = df['Insulin'].mask(
    (df['Insulin'] == 0) | (df['Insulin'] >= 300), median_insulin)

# BMI 전처리
df = df.drop(df[(df['BMI'] == 0) | (df['BMI'] > 50)].index)
```

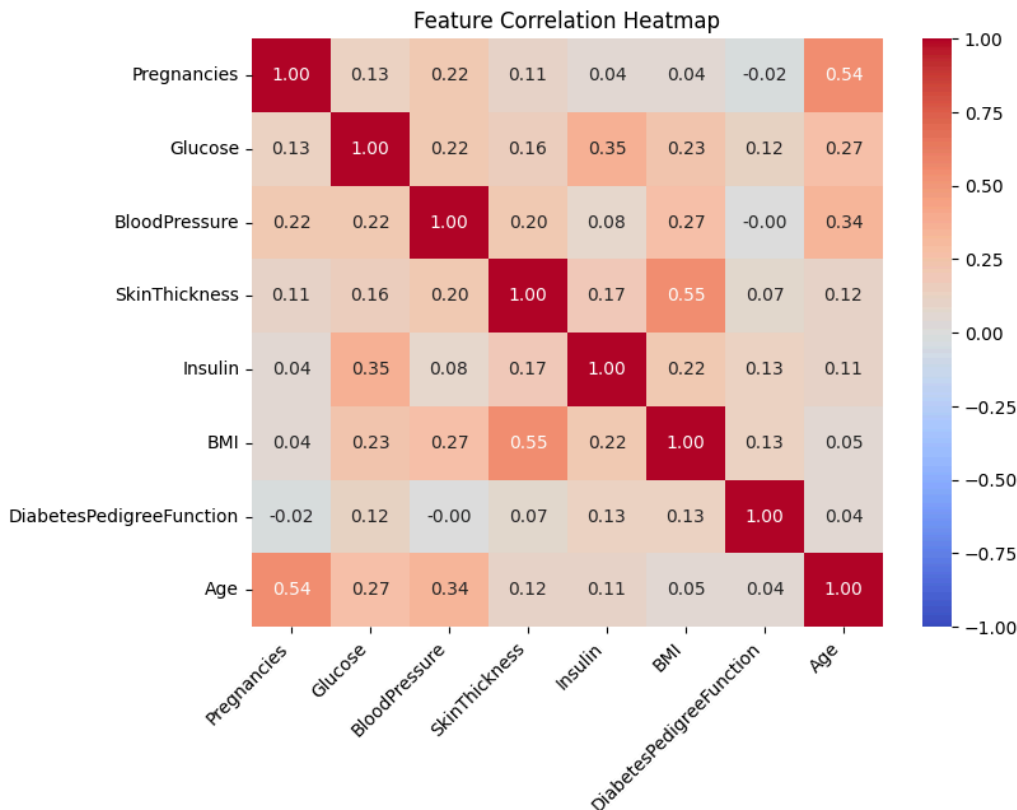
4) 독립변수 간 상관관계 정도 파악 : 독립변수 간 상관관계를 파악하기 위해 **Outcome** 변수를 제외하고 분석하였으며, 상관계수의 유의성을 검증하기 위해 **p-value**를 함께 산출함.

```
feature_df = df.drop(columns=['Outcome'])
plt.figure(figsize=(8,6))
sns.heatmap(feature_df.corr(), annot=True, fmt=".2f", cmap='coolwarm', vmin=-1, vmax=1)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.yticks(fontsize=10)
plt.title("Feature Correlation Heatmap")
plt.show()
```

```
import pingouin as pg

results = pg.pairwise_corr(
    data=feature_df,
    columns=feature_df.columns,
    method='pearson',
    alternative='two-sided',
    padjust='none')

print(results)
```



- 상관관계 스케일을 -1에서 1로 고정하여 파랑색 영역(음수)이 전혀 사용되지 않은 것을 통해, 음의 상관관계가 거의 없음을 시각적으로 확인할 수 있음.
- 강한 양의 상관관계
  1. BMI ↔ SkinThickness ( $r=0.55$ , 95% CI [0.50, 0.60],  $p<1e-60$ )  
→ 둘 다 체지방·비만 지표로, 생리학적으로 높은 양의 상관관계가 예상됨.
  2. Age ↔ Pregnancies ( $r=0.54$ , 95% CI [0.49, 0.59],  $p<2e-58$ )  
→ 나이가 많을수록 임신 횟수가 누적될 가능성이 높아 양의 상관관계를 보이는 것이 합리적임.
- 중간 수준 양의 상관관계
  1. Insulin ↔ Glucose ( $r=0.35$ , 95% CI [0.29, 0.42],  $p<1e-23$ )  
→ OGTT 검사 지표로 둘 다 당 대사 이상 시 동시에 상승하는 경향이 있음. 즉, 같은 방향성을 보임.
  2. Age ↔ BloodPressure ( $r=0.34$ , 95% CI [0.27, 0.40],  $p<5e-21$ )  
→ 고령이 고혈압 위험인자로 알려진 의학적 배경과 부합하는 결과임.
- 모든 주요 상관관계가 p값이 0.05 미만이며, 신뢰구간에 0을 포함하지 않으므로 통계적으로 유의함.
- 회귀모델 수행 시 다중공선성으로 인한 불안정한 추정값 및 해석 오류가 발생할 수 있으므로, 상관관계가 너무 높은 변수 쌍에 대해서는 주의를 기울여야 함.

#### 4. 특성 및 타겟 변수 분리와 훈련/ 검증 데이터셋 분할

```
from sklearn.model_selection import train_test_split
X = df.drop('Outcome',axis=1)
y = df['Outcome']
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.2, random_state=42)
```

- 이후 모델 성능 평가를 위해 전체의 20%를 검증(valid) 데이터로 할당하고, 나머지 80%는 학습(train) 데이터로 이용함.
- 데이터 분할의 재현성을 위한 random\_state는 값은 42로 설정함.

#### 5. 주요 피처의 표준화(StandardScaler) 및 로그 변환(log transform)

**1) 목적 :** 연속형 변수(예: Glucose, Age, DiabetesPedigreeFunction 등)들의 측정 단위와 분포 범위가 상이하여 가중치를 학습하는 회귀에서 한쪽 변수만 지나치게 영향력이 커질 수 있으므로 모든 수치형 변수에 대해 평균0, 분산1의 표준화를 적용함.

##### 2) 로그 변환할 변수 판단

```
skewness = X_train.apply(lambda col: col.skew())
print(skewness)
```

Pregnancies	0.873599
Glucose	0.543441
BloodPressure	0.003577
SkinThickness	0.087960
Insulin	1.308638
BMI	0.227206
DiabetesPedigreeFunction	1.776765
Age	1.095801
dtype:	float64

- 이상치 처리 후 각 변수의 왜도(skewness)를 계산함.

- 왜도가 1 이상인 변수에 대해서 로그 변환을 적용함.

→ Insulin (skew≈1.31)과 DiabetesPedigreeFunction(DPF) (skew≈1.78)는 왜도가 높아 로그 변환 적용

→ Age (skew≈1.10)는 1을 크게 초과하지는 않으나 해석 편의성보다 모델 성능을 중시하여 로그 변환 적용

##### 3) 로그 변환 및 표준화

```
feature_to_log_transform = ['Insulin', 'DiabetesPedigreeFunction', 'Age']
```

```
X_train[feature_to_log_transform] = np.log(X_train[feature_to_log_transform])
X_valid[feature_to_log_transform] = np.log(X_valid[feature_to_log_transform])
```

```
from sklearn.preprocessing import StandardScaler
```

```
X_train = pd.DataFrame(
    scaler.fit_transform(X_train),
    columns=X_train.columns, index=X_train.index)
X_valid = pd.DataFrame(
    scaler.transform(X_valid),
    columns=X_valid.columns, index=X_valid.index)
```



## 6. 로지스틱 회귀 모델 구축 및 검증 데이터에 대한 성능 평가

### 1) 다중공선성 점검(VIF)

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.tools import add_constant

X_train_vif = add_constant(X_train)

vif_df = pd.DataFrame()
vif_df['Variable'] = X_train_vif.columns
vif_df['VIF'] = [variance_inflation_factor(X_train_vif.values,i) for i in range(X_train_vif.shape[1])]

print(vif_df)
```

	Variable	VIF
0	const	1.000000
1	Pregnancies	1.525172
2	Glucose	1.281468
3	BloodPressure	1.224105
4	SkinThickness	1.483686
5	Insulin	1.197339
6	BMI	1.588321
7	DiabetesPedigreeFunction	1.034090
8	Age	1.710944

- 로지스틱 회귀 모델 적합 전, 변수 간 다중공선성으로 인한 계수 추정 불안정과 과적합 가능성을 방지하기 위해 VIF(Variance Inflation Factor)를 계산함. 특히, 앞서 상관관계분석에서 BMI와 SkinThickness와 같이 유의미한 양의 상관관계를 보이는 변수들이 있었으므로 값을 계산해보는 것이 바람직함.
- 계산 결과 모든 변수의 VIF 값이 1~5 범위 내에 위치하여(일반적으로  $VIF < 5$ 는 다중공선성 우려가 낮은 수준으로 간주됨) 다중공선성 문제가 크지 않음을 확인함.

### 2) 로지스틱 회귀 모델 구축

```
from sklearn.linear_model import LogisticRegression

X_train = X_train.drop('const', axis=1)
model = LogisticRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_valid)
```

- 데이터 해석이 아니라, 새로운 데이터 예측이 목표이므로 statsmodels 라이브러리 대신 scikit-learn을 사용함.

### 3) 혼동행렬(Confusion Matrix) 기반 지표를 이용한 모델 성능 평가

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score

accuracy_diabetes_LR = accuracy_score(y_valid, y_pred)
precision_diabetes_LR = precision_score(y_valid, y_pred)
recall_diabetes_LR = recall_score(y_valid, y_pred)
f1_diabetes_LR = f1_score(y_valid, y_pred)

print("Accuracy:", accuracy_diabetes_LR)
print("Precision:", precision_diabetes_LR)
print("Recall:", recall_diabetes_LR)
print("F1 Score:", f1_diabetes_LR)

cf_mat = confusion_matrix(y_valid, y_pred)
display(cf_mat)
```

Accuracy: 0.8187919463087249  
Precision: 0.6976744186046512  
Recall: 0.6818181818181818  
F1 Score: 0.6896551724137931  
array([[92, 13],  
 [14, 30]])

- 정확도(전체 데이터 중 올바르게 예측한 값) : 0.818
- 정밀도(모델이 양성으로 예측한 항목들 중 실제로 양성인 항목의 비율) : 0.697
- 재현율(실제 양성 클래스 중 모델이 올바르게 양성으로 예측한 비율) : 0.681
- F1 score(정밀도와 재현율의 조화 평균을 나타내는 지표) : 0.689

#### 4) 임계값 독립 지표(Threshold-independent)를 이용한 모델 성능 평가

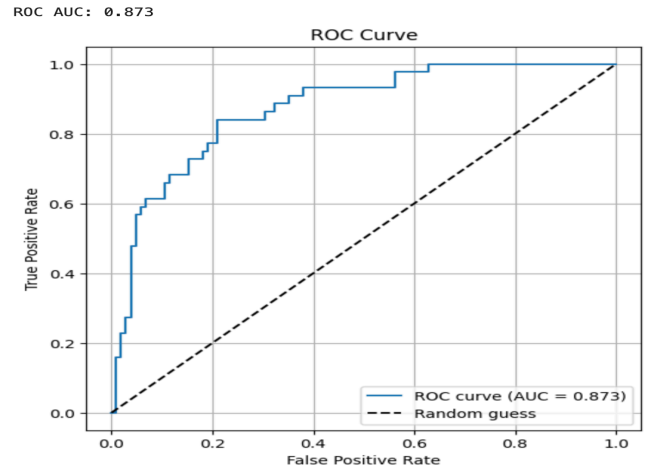
```
from sklearn.metrics import roc_auc_score, roc_curve

y_prob = model.predict_proba(X_valid)[:, 1]

roc_auc = roc_auc_score(y_valid, y_prob)
print(f"ROC AUC: {roc_auc:.3f}")

fpr, tpr, thresholds = roc_curve(y_valid, y_prob)

plt.figure(figsize=(6,6))
plt.plot(fpr, tpr, label=f'ROC curve (AUC = {roc_auc:.3f})')
plt.plot([0,1], [0,1], 'k--', label='Random guess')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend(loc='lower right')
plt.grid(True)
plt.show()
```



- FPR이 0.1 이하인 구간에서 TPR이 0.6까지 급격히 상승하여, 낮은 오탐률에서도 상당한 재현율을 확보함.
- 전체 AUC(Area Under the Curve) 값은 0.873으로 0.8~0.9 사이에 해당하므로 “우수” 수준의 분류 성능을 나타냄.

## 7. 랜덤 포레스트 모델 구축 및 검증 데이터에 대한 성능 평가

### 1) 랜덤 포레스트 모델 구축

```
from sklearn.ensemble import RandomForestClassifier

RF_model = RandomForestClassifier(random_state=42)
RF_model.fit(X_train, y_train)
RF_y_pred = RF_model.predict(X_valid)
```

- 랜덤 포레스트는 변수 표준화가 필수는 아니지만, 회귀분석 모델과의 공정한 성능 비교를 위해 동일한 전처리를 적용한 학습 데이터를 사용함.

### 2) 혼동행렬(Confusion Matrix) 기반 지표를 이용한 모델 성능 평가

```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score

accuracy_diabetes_RF = accuracy_score(y_valid, RF_y_pred)
precision_diabetes_RF = precision_score(y_valid, RF_y_pred)
recall_diabetes_RF = recall_score(y_valid, RF_y_pred)
f1_diabetes_RF = f1_score(y_valid, RF_y_pred)

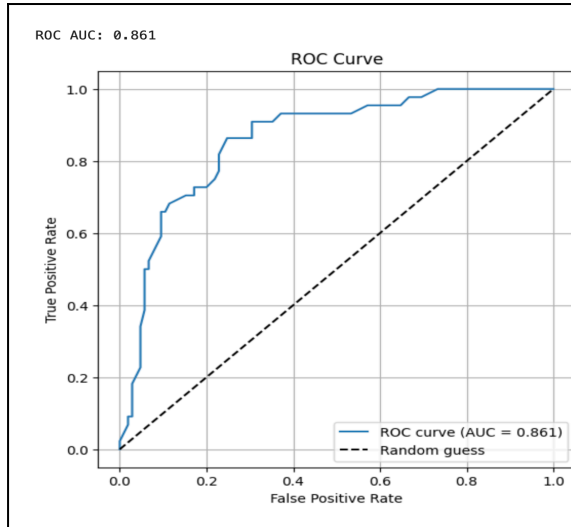
print("Accuracy:", accuracy_diabetes_RF)
print("Precision:", precision_diabetes_RF)
print("Recall:", recall_diabetes_RF)
print("F1 Score:", f1_diabetes_RF)

cf_mat_RF = confusion_matrix(y_valid, RF_y_pred)
display(cf_mat_RF)

Accuracy: 0.8053691275167785
Precision: 0.6595744680851063
Recall: 0.7045454545454546
F1 Score: 0.6813186813186813
array([[89, 16],
       [13, 31]])
```

- 정확도 : 0.805
- 정밀도 : 0.659
- 재현율 : 0.704
- F1 score : 0.681

### 3) 임계값 독립 지표(Threshold-independent)를 이용한 모델 성능 평가



- FPR이 0.1 이하인 구간에서 TPR이 0.6까지 급격히 상승하여, 낮은 오탐률에서도 상당한 재현율을 확보함.
  - 전체 AUC(Area Under the Curve)는 0.861로 0.8~0.9 사이에 해당하므로 “우수” 수준의 분류 성능을 나타냄.
- (참고 : 로지스틱 회귀 모델의 AUC=0.873 대비 소폭 낮지만, 큰 차이는 아님)

## 8. 결론

### 1) 로지스틱 회귀 모델 우위(전반적 성능)

- 전반적인 혼동행렬 기반 지표(Accuracy, Precision, F1등)와 ROC-AUC 모두 로지스틱 회귀가 조금 더 높게 나타남.
- 가능 원인
  1. 샘플 수(데이터 양) 부족 : 이번 분석에 쓰인 데이터셋의 샘플이 768개로 비교적 작아, 앙상블 트리마다 사용하는 부트스트랩 샘플이 적어 각 트리가 과도하게 분산이 커졌을 수 있음.
  2. 특성과 타깃 간의 관계가 대체로 선형적이어서, 선형 결정 경계 가정이 맞는 로지스틱 회귀가 최적화된 성능을 낼 수 있음.

### 2) 랜덤포레스트의 강점(재현율)

- 랜덤 포레스트 모델의 재현율(0.704)이 로지스틱 회귀 모델(0.681)보다 높아, 거짓 음성을 최소화하는 것이 중요한 의료 진단의 특성상 이후 실제 환자 검출률 측면에서 더 실용적일 수 있음.

### 3) 각 모델 중요 변수 파악을 통한 성능 차이 분석

# 로지스틱 회귀 계수 확인				# 랜덤포레스트 변수 중요도		
import pandas as pd				importances_df = pd.DataFrame({'Feature': X.columns,		
coefficients = model.coef_[0]				'RF_Importance': RF_model.feature_importances_		
coef_df = pd.DataFrame({'Feature': X_train.columns,				}).sort_values(by='RF_Importance', ascending=False)		
'Coefficient': coefficients})				print(importances_df)		
coef_df['Abs_Coefficient'] = coef_df['Coefficient'].abs()						
coef_df = coef_df.sort_values(by='Abs_Coefficient', ascending=False)						
print(coef_df)						
	Feature	Coefficient	Abs_Coefficient		Feature	RF_Importance
1	Glucose	1.082046	1.082046	1	Glucose	0.267192
5	BMI	0.553506	0.553506	5	BMI	0.177595
0	Pregnancies	0.375229	0.375229	6	DiabetesPedigreeFunction	0.126780
6	DiabetesPedigreeFunction	0.311521	0.311521	7	Age	0.115452
7	Age	0.171470	0.171470	2	BloodPressure	0.085429
4	Insulin	0.152516	0.152516	0	Pregnancies	0.079007
2	BloodPressure	-0.045607	0.045607	4	Insulin	0.078563
3	SkinThickness	-0.017521	0.017521	3	SkinThickness	0.069981

- 두 모델 모두 당뇨 발병 이전에 직접적으로 연관성이 큰 Glucose와 BMI를 핵심 예측 변수로 활용함.
- 특히 중요도의 순서가 다른 변수는 Pregnancies임. 앞서 언급했던 바와 같이 임신 횟수와 임신성 당뇨는 간접적 연관성만 보고된 바 있으며, 명확한 인과관계는 아직 확립되지 않음. 즉, 다른 변수들에 비해 당뇨 유무와 일관된 관계가 약함.
- 랜덤 포레스트 모델은 해당 변수를 거의 분할 기준으로 사용하지 않아, 강한 신호에 더 많이 의존하도록 모델의 의사결정 경계를 재조정했을 가능성이 있음.
- 이로 인해 실제 당뇨 환자를 놓치는 비율이 줄어들어 재현율은 상승했으나, 음성인 케이스를 양성으로 잘못 분류하는 비율도 증가하여 전체 정확도는 감소하는 트레이드오프가 발생함.

#### 4) 한계

- 주요 지표 간 격차가 소수점 둘째 자리 수준에 불과함. 따라서 부트스트래핑(bootstrap)이나 교차 검증(Cross-Validation) 등을 통해 '통계적으로 유의미한 차이'인지 확인할 필요가 있음.

#### <참고 자료>

- 서울아산병원

<https://www.amc.seoul.kr/asan/healthinfo/disease/diseaseDetail.do?contentId=31837>

<https://www.amc.seoul.kr/asan/healthinfo/management/managementDetail.do?managementId=172>

<https://www.amc.seoul.kr/asan/mobile/healthinfo/management/managementDetail.do?managementId=2>

- 대한당뇨협회

[https://www.diabetes.or.kr/general/info/info\\_05.php](https://www.diabetes.or.kr/general/info/info_05.php)

- 의협신문

<https://www.doctorsnews.co.kr/news/articleView.html?idxno=105227>

- 대한영양사협회

<https://www.dietitian.or.kr/board/file/64/63275805520070327012948>

- 세브란스

[https://sev.severance.healthcare/health/encyclopedia/treat\\_board.do?mode=view&articleNo=67023&title=%EC%9D%B8%EC%8A%90%EB%A6%B0](https://sev.severance.healthcare/health/encyclopedia/treat_board.do?mode=view&articleNo=67023&title=%EC%9D%B8%EC%8A%90%EB%A6%B0)

- 삼성서울병원

[http://www.samsunghospital.com/dept/main/index.do?DP\\_CODE=DM&MENU\\_ID=001029](http://www.samsunghospital.com/dept/main/index.do?DP_CODE=DM&MENU_ID=001029)

- 질병관리청

[https://health.kdca.go.kr/healthinfo/biz/health/gnrizHealthInfo/gnrizHealthInfo/gnrizHealthInfoView.do?cntnts\\_sn=5305](https://health.kdca.go.kr/healthinfo/biz/health/gnrizHealthInfo/gnrizHealthInfo/gnrizHealthInfoView.do?cntnts_sn=5305)

<논문> 인슐린저항성의 발생기전(한국마우스대사질환특화센터, 이길여 암당뇨연구원, 가천의과학대학교)