빅데이터사례연구

뇌졸중
발병 가능성 예측

데이터사이언스학과
2020012021 표성주

Contents

뇌졸중 발병 가능성 예측

01 문제 정의

뇌졸중 발병 가능성 예측

- 뇌졸중은 뇌혈관 질환을 통칭하는 말
- 2019년 WHO (세계보건기구)
  전세계 사망 원인 2위
- 2021년 통계청
  한국인 사망 원인 4위
  (단일 질환 1위)
- 장애, 기억 상실 등의 증상을 초래

(단위: 인구 10만 명당 명)

순위	사망원인	사망률	'20년 순위 대비
1	악성신생물(암)	161.1	-
2	심장 질환	61.5	-
3	폐렴	44.4	-
4	뇌혈관 질환	44.0	-
5	고의적 자해(자살)	26.0	-
6	당뇨병	17.5	-
7	알츠하이머병	15.6	-
8	간질환	13.9	-
9	패혈증	12.5	⬆(+1)
10	고혈압성 질환	12.1	⬇(-1)

02 데이터 수집

뇌졸중 발병 가능성 예측

- 캐글에서 제공하는 뇌졸중 발병 여부 데이터
- 아이디, 성별, 연령, 고혈압 여부,
  심장병 여부, 체질량 지수, 흡연 상태, 뇌졸중 여부
- 총 5110개의 데이터

```
df.describe()
```

	id	age	hypertension	heart_disease	bmi	stroke
count	5110.000000	5110.000000	5110.000000	5110.000000	4909.000000	5110.000000
mean	36517.829354	43.226614	0.097456	0.054012	28.893237	0.048728
std	21161.721625	22.612647	0.296607	0.226063	7.854067	0.215320
min	67.000000	0.080000	0.000000	0.000000	10.300000	0.000000
25%	17741.250000	25.000000	0.000000	0.000000	23.500000	0.000000
50%	36932.000000	45.000000	0.000000	0.000000	28.100000	0.000000
75%	54682.000000	61.000000	0.000000	0.000000	33.100000	0.000000
max	72940.000000	82.000000	1.000000	1.000000	97.600000	1.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              5110 non-null   int64
 1   gender          5110 non-null   object
 2   age             5110 non-null   float64
 3   hypertension    5110 non-null   int64
 4   heart_disease   5110 non-null   int64
 5   bmi             4909 non-null   float64
 6   smoking_status  5110 non-null   object
 7   stroke          5110 non-null   int64
dtypes: float64(2), int64(4), object(2)
memory usage: 319.5+ KB
```

03

EDA & 전처리

뇌졸중 발병 가능성 예측

- id 칼럼, 성별이 other인 데이터 삭제
- bmi 변수 결측치 201개 존재
  → 결측 데이터를 평균값으로 대체
- 흡연 상태를 수치형으로 변환

```
df.isnull().sum()
#결측치 확인
```

```
id                 0
gender             0
age                0
hypertension       0
heart_disease      0
bmi              201
smoking_status     0
stroke             0
dtype: int64
```

```
df['gender'].value_counts()
# gender 데이터별 개수 확인
```

```
Female    2994
Male      2115
Other        1
Name: gender, dtype: int64
```

```
df.drop(columns=['id'], inplace=True)
df.drop(df[df['gender'] == 'Other'].index, axis=0, inplace=True)
# id 칼럼, 성별 other 데이터 제거
A = df['bmi'].mean()
df['bmi'] = df['bmi'].fillna(A)
# bmi 결측치 평균값으로 대체
```

```
df['smoking_status'].value_counts()
```

```
never smoked       1892
Unknown            1544
formerly smoked     885
smokes              789
Name: smoking_status, dtype: int64
```

```
len(df.loc[(df['smoking_status'] == 'never smoked') & (df['stroke'] == 0)])
```

```
1802
```

```
len(df.loc[(df['smoking_status'] == 'Unknown') & (df['stroke'] == 0)])
```

```
1497
```

```
df.loc[df['smoking_status'] == 'Unknown', 'smoking_status'] = 0
df.loc[df['smoking_status'] == 'never smoked', 'smoking_status'] = 0
df.loc[df['smoking_status'] == 'formerly smoked', 'smoking_status'] = 1
df.loc[df['smoking_status'] == 'smokes', 'smoking_status'] = 2
df = df.astype({'smoking_status':int})
```

03 EDA & 전처리

뇌졸중 발병 가능성 예측

- 뇌졸중에 걸리지 않은 사람의 데이터 개수가
  뇌졸중에 걸린 사람의 데이터 개수보다 약 20배 가량 많음
- 뇌졸중 발병 여부에 따라 데이터 분리
- 4860개의 데이터 중 랜덤하게 250개 추출
- pd.concat 함수로 두 데이터 프레임을 결합

```
df['stroke'].value_counts()
```

```
0    4860
1     249
Name: stroke, dtype: int64
```

```
S = df[df['stroke'] == 0]
len(S)
```

```
4860
```

```
s1 = S.sample(n=250, random_state=1234)
len(s1)
```

```
250
```

```
s2 = df[df['stroke'] == 1]
len(s2)
```

```
249
```

```
ddd = pd.concat([s1,s2])
ddd['stroke'].value_counts()
```

```
0    250
1    249
Name: stroke, dtype: int64
```

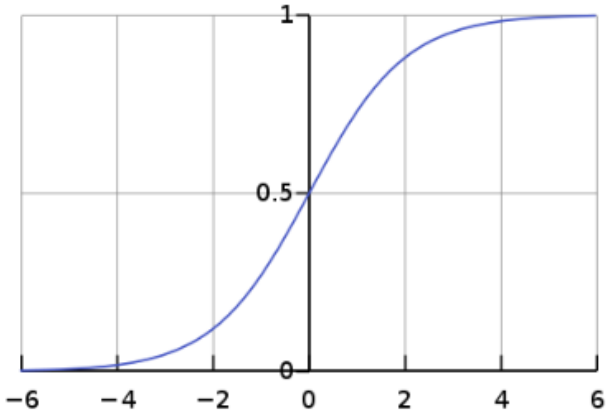
04

분석 방법

뇌졸중 발병 가능성 예측

로지스틱 회귀분석

- 변수 사이의 연관성을 알아보는 분석 방법
- 종속변수가 이진형 일 때 사용 가능
  ex) 질병 발생 여부 예측, 시험 합격 여부 예측,
  금융 고객 신용도 예측, 제품의 불량 여부 예측



표준 로지스틱 함수

상관분석

- 변수 간 관계의 정도를 알아보는 분석 방법
- 상관계수가 1에 가까울수록 강한 양의 상관관계,
  상관계수가 -1에 가까울수록 강한 음의 상관관계

05

데이터 분석

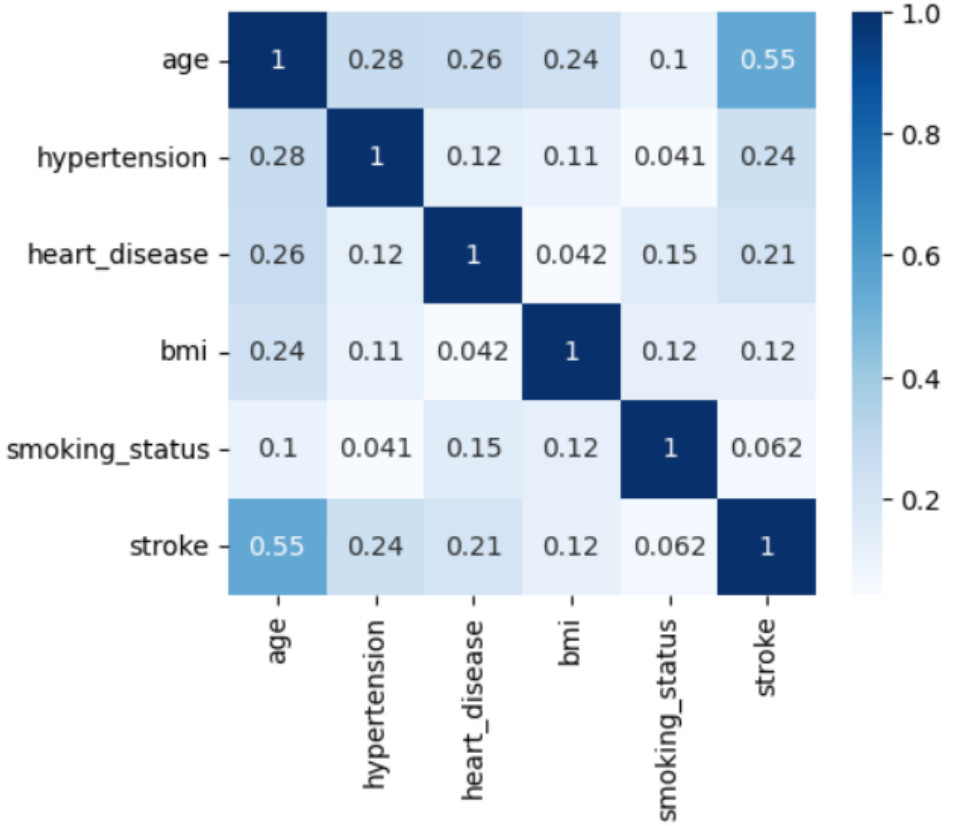
뇌졸중 발병 가능성 예측

- 독립변수 = 연령, 고혈압 여부, 심장병 여부
- 종속변수 = stroke (뇌졸중 발병 여부)
- 독립변수들 간 상관관계가 낮음
  → 회귀분석을 하기에 적합

```
# 고혈압 여부와 심장병 여부의 상관관계 (파이계수)
from scipy.stats.contingency import association
X = np.array([[371, 42],
              [68, 18]])
association(X, method="tschuprow")
```

0.12495641187948268

```
import matplotlib.pyplot as plt
import seaborn as sns
plt.figure(figsize=(5,4))
sns.heatmap(corr,annot=True, cmap = 'Blues')
```



05 데이터 분석

뇌졸중 발병 가능성 예측

- 독립변수: 연령, 고혈압 여부, 심장병 여부
- 종속변수: 뇌졸중 발병 여부
- train data, test data 분리
- 데이터 정규화
- 로지스틱 회귀 모델 생성

```
X = ddd[['age', 'hypertension', 'heart_disease', 'bmi', 'smoking_status']]
Y = ddd['stroke']
```

```
from sklearn.model_selection import train_test_split
train_X, test_X, train_Y, test_Y = train_test_split(X, Y)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
train_X = scaler.fit_transform(train_X)
test_X = scaler.transform(test_X)
```

```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(train_X, train_Y)
```

```
▾ LogisticRegression
LogisticRegression()
```

06

결과 & 예측

뇌졸중 발병 가능성 예측

- 생성된 모델이 0.84의 결정계수를 가짐
  → 독립변수들이 종속변수에 미치는 영향이 84%
  → 독립변수들이 뇌졸중 발병 여부의 84% 설명 가능

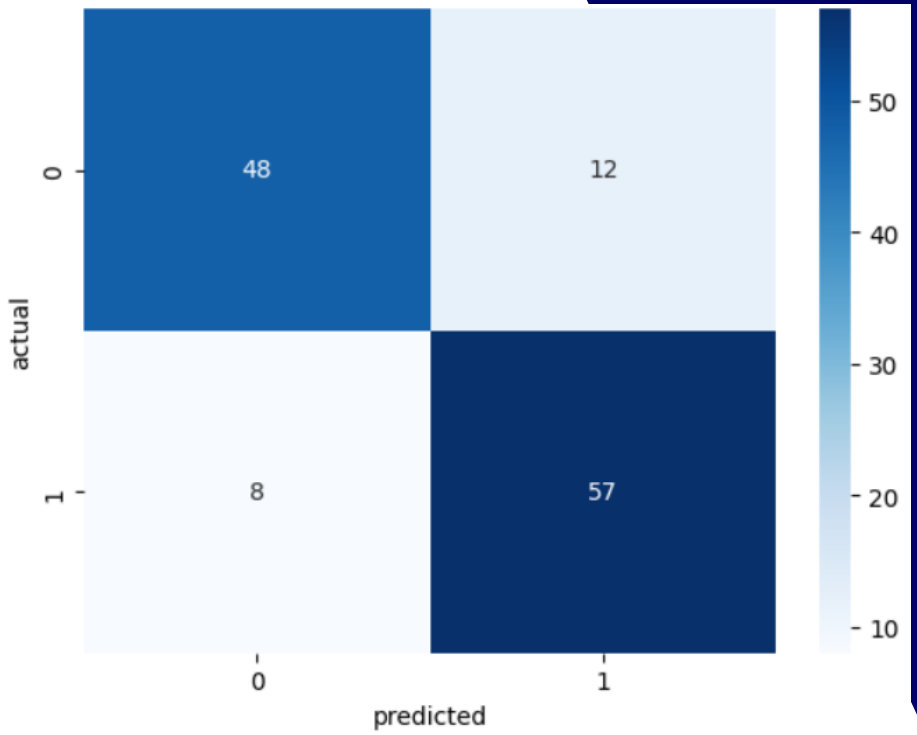
```
print(model.score(test_X, test_Y))
```

```
0.84
```

```
from sklearn.metrics import classification_report

y_pred = model.predict(test_X)
print(classification_report(test_Y, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.86      0.80      0.83        60
           1       0.83      0.88      0.85        65

    accuracy                           0.84       125
   macro avg       0.84      0.84      0.84       125
weighted avg       0.84      0.84      0.84       125
```

```
sns.heatmap(confusion_matrix (test_Y, y_pred),
            annot = True, fmt = "d",cmap = 'Blues')
plt.xlabel('predicted')
plt.ylabel('actual')
```



06

결과 & 예측

뇌졸중 발병 가능성 예측

- 연령, 고혈압 여부, 심장병 여부
- 고혈압 여부: 0 = 없음 / 1 = 있음
- 심장병 여부: 0 = 없음 / 1 = 있음
- 뇌졸중 발병 여부: 0 = 발병하지 않음 / 1 = 발병

- 사람1,2는 뇌졸중이 발병하지 않고
  사람3,4는 뇌졸중이 발병할 것이라고 예측

```
print(model.coef_)
```

```
[[1.49471267 0.35003241 0.30750689]]
```

```
MAN1 = np.array([27, 0, 0])
MAN2 = np.array([45, 1, 0])
MAN3 = np.array([64, 0, 1])
MAN4 = np.array([83, 1, 1])
```

```
sample_df = np.array([MAN1, MAN2, MAN3, MAN4])
```

```
sample_df = scaler.transform(sample_df)
```

```
print(model.predict(sample_df))
print(model.predict_proba(sample_df))
```

```
[0 0 1 1]
[[0.88110603 0.11889397]
 [0.54023396 0.45976604]
 [0.34491557 0.65508443]
 [0.07293541 0.92706459]]
```

THANK YOU