```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_1.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 5

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_1.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 128)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 1s 697us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics        ANTBT_NM  W
0            6      1875.0   6.00          경쾌     Clindamycin
1           68       500.0   5.00          경쾌      Rifampicin
2           63      1125.0   2.00          경쾌        Imipenem
3           51       750.0  25.67          경쾌        Imipenem
4           64       600.0   2.00          경쾌   Ciprofloxacin
...        ...         ...    ...         ...             ...
66551       62      4500.0   1.00          경쾌      Ampicillin
66552       53      3000.0   8.33          경쾌     Ceftazidime
66553       63      1125.0   8.00          경쾌      Vancomycin
66554       19      6000.0  23.00          경쾌  Nitrofurantoin
66555       63      1125.0  13.00          경쾌     Ceftazidime

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.018794              S
1                 0.136717              S
2                 0.109668              S
3                 0.138876              S
4                 0.502279              R
...                    ...            ...
66551             0.283344              S
66552             0.160341              S
66553             0.108715              S
66554             0.061019              S
66555             0.148830              S
```

```
[66556 rows x 7 columns]
예측 정확도: 0.6914177534707615
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_2.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 6

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_2.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 127)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 759us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics     ANTBT_NM  ₩
0           29      6000.0   8.00          경쾌   Gentamicin
1           53      2000.0  13.00          경쾌   Penicillin
2           59     18000.0  17.00          경쾌     Cefepime
3           59     13500.0   4.33          경쾌   Tigecyline
4           64       400.0  43.00          경쾌  Ceftazidime
...        ...         ...    ...         ...          ...
66551       51       500.0  10.00          경쾌  Teicoplanin
66552       37      1000.0  14.00          경쾌  Ceftazidime
66553       56       400.0   3.00          경쾌    Cefoxitin
66554       28       200.0  14.00          경쾌    Aztreonam
66555       33      2000.0   5.00          경쾌    Meropenem

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.004430              S
1                 0.040788              S
2                 0.006926              S
3                 0.006153              S
4                 0.050838              S
...                    ...            ...
66551             0.004919              S
66552             0.031086              S
66553             0.005224              S
66554             0.018200              S
66555             0.029259              S
```

```
[66556 rows x 7 columns]
예측 정확도: 0.6785113288058177
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_3.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 9

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_3.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 124)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 751us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA   DAY antibiotics     ANTBT_NM  ₩
0           36      2000.0   4.0          경쾌     Amikacin
1           36      2000.0   1.0          경쾌    Ertapenem
2           59     18000.0   9.0          경쾌     Amikacin
3           59     18000.0   4.0          경쾌     Cefepime
4           33      2000.0   8.0          경쾌  Ceftazidime
...        ...         ...   ...         ...          ...
66551       15      2000.0  10.0          경쾌    Ertapenem
66552       53       675.0   8.0          경쾌   Penicillin
66553       36      4000.0  43.0          경쾌    Linezolid
66554       20       200.0   5.0          경쾌   Penicillin
66555       54      1500.0   8.0          경쾌     Imipenem

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.092387              S
1                 0.013686              S
2                 0.116850              S
3                 0.123644              S
4                 0.034857              S
...                    ...            ...
66551             0.009700              S
66552             0.043191              S
66553             0.082688              S
66554             0.034391              S
```

```
66555            0.117912             S

[66556 rows x 7 columns]
예측 정확도: 0.6762876374782139
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_4.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 5

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_4.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 128)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 722us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics       ANTBT_NM  ₩
0           33      2000.0   3.00          경쾌    Ceftazidime
1           51       500.0  21.00          경쾌      Meropenem
2           68       900.0   4.00          경쾌       Amikacin
3           59     18000.0   4.00          경쾌  Ciprofloxacin
4           58      6750.0  29.09          경쾌     Gentamicin
...        ...         ...    ...         ...            ...
66551       36      2000.0   3.00          경쾌     Gentamicin
66552       38       800.0  27.00          완쾌     Penicillin
66553        0       500.0   4.00          경쾌     Gentamicin
66554       51      1000.0  18.37          경쾌     Cefotaxime
66555       28       200.0  10.00          경쾌    Ceftazidime

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.069357              S
1                 0.172963              S
2                 0.021076              S
3                 0.011328              S
4                 0.015443              S
...                    ...            ...
66551             0.008790              S
66552             0.056020              S
66553             0.031944              S
```

```
66554            0.017498             S
66555            0.072126             S

[66556 rows x 7 columns]
예측 정확도: 0.6827934371055953
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_5.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 7

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_5.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 126)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 721us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics                      ANTBT_NM  W
0           20       200.0  15.00          경쾌                    Penicillin
1           68      2000.0   6.00          경쾌                     Oxacillin
2           59     18000.0   2.75          경쾌  Trimethoprim/Sulfamethoxazole
3           52      1200.0   5.00          경쾌                     Aztreonam
4           33      2000.0   4.00          경쾌                     Cefazolin
...        ...         ...    ...         ...                           ...
66551       58      6750.0  36.00          경쾌                    Vancomycin
66552       51      1000.0  18.37          경쾌                      Amikacin
66553       20       400.0   2.00          경쾌                    Ampicillin
66554       68      2000.0   7.44          경쾌                    Minocycline
66555       68      2000.0   5.85          경쾌                    Minocycline

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.197144              S
1                 0.474212              S
2                 0.208477              S
3                 0.457580              S
4                 0.024013              S
...                    ...            ...
66551             0.120058              S
66552             0.079718              S
```

```
66553            0.597302             R
66554            0.034754             S
66555            0.034754             S

[66556 rows x 7 columns]
예측 정확도: 0.6764378868922412
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_6.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 9

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_6.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 124)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 732us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA   DAY antibiotics     ANTBT_NM  W
0           33      2000.0   4.0          경쾌   Ampicillin
1           36      2000.0   2.0          경쾌  Teicoplanin
2           51      1000.0   7.0          경쾌   Cefotaxime
3           19      1000.0  44.0          경쾌     Cefepime
4           20       200.0   7.0          경쾌     Imipenem
...        ...         ...   ...         ...          ...
66551       38       800.0   7.0          경쾌   Ampicillin
66552       19      1000.0  15.0          경쾌  Ceftriaxone
66553       20       200.0   9.0          경쾌   Cefotaxime
66554       46      1000.0   6.0          경쾌    Aztreonam
66555       63      1125.0   7.0          경쾌     Imipenem

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.297980              S
1                 0.189013              S
2                 0.097373              S
3                 0.011581              S
4                 0.006535              S
...                    ...            ...
66551             0.438646              S
```

```
66552            0.162407             S
66553            0.072912             S
66554            0.132001             S
66555            0.009860             S

[66556 rows x 7 columns]
예측 정확도: 0.6827032874571789
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_7.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 6

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_7.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 127)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 751us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics                      ANTBT_NM  ₩
0           20       200.0   6.00          경쾌         Cefoperazone/Sulbactam
1           20       200.0   3.50          경쾌    Amoxicillin/Clavulanic acid
2           59     13500.0   3.00          경쾌                     Gentamicin
3           68      1800.0  10.59          경쾌  Trimethoprim/Sulfamethoxazole
4           56       400.0  52.00          경쾌                       Amikacin
...        ...         ...    ...         ...                            ...
66551       46      1000.0  15.00          경쾌                   Tetracycline
66552        6      1875.0   7.00          경쾌                  Ciprofloxacin
66553       19      4000.0  15.50          경쾌                      Linezolid
66554       53      3000.0   7.00          경쾌                       Imipenem
66555       41      1800.0   9.00          경쾌                     Gentamicin

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.064534              S
1                 0.010034              S
2                 0.005308              S
3                 0.040290              S
4                 0.044010              S
...                    ...            ...
```

```
66551            0.052596             S
66552            0.111715             S
66553            0.043213             S
66554            0.073968             S
66555            0.002781             S

[66556 rows x 7 columns]
예측 정확도: 0.6773694332592103
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_8.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 6

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_8.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 127)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 867us/step
예측한 데이터:
        INGR_NM  TPRSC_CAPA    DAY antibiotics      ANTBT_NM  ₩
0            53      3000.0   5.00          경쾌      Imipenem
1            63      1125.0   7.00          경쾌     Ampicillin
2            68      1600.0  10.16          경쾌     Gentamicin
3            56       400.0  29.00          경쾌   Levofloxacin
4            15      2000.0  26.00          경쾌     Gentamicin
...         ...         ...    ...         ...            ...
66551        68      2000.0  14.00          경쾌     Vancomycin
66552        33      2000.0   8.00          경쾌     Vancomycin
66553        19      2000.0  10.00          경쾌       Cefepime
66554        56       400.0  20.00          경쾌      Aztreonam
66555        11       500.0   3.00          경쾌       Amikacin

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.073968              S
1                 0.003811              S
2                 0.004007              S
3                 0.050435              S
4                 0.002356              S
```

```
...                  ...            ...
66551            0.026860              S
66552            0.018212              S
66553            0.055598              S
66554            0.077866              S
66555            0.042780              S

[66556 rows x 7 columns]
예측 정확도: 0.6772943085521966
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_9.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 4

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_9.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:Wn", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 129)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 2s 841us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics                    ANTBT_NM  W
0           33      2000.0   3.00          경쾌                 Ceftazidime
1           38       400.0  10.00          경쾌       Ampicillin/Sulbactam
2           28       400.0   7.00          완쾌                  Cefazolin
3           62      2700.0   2.00          경쾌                 Penicillin
4           38       800.0   3.12          경쾌                   Cefepime
...        ...         ...    ...         ...                        ...
66551       38       500.0   8.60          경쾌               Ciprofloxacin
66552       20       200.0   5.00          경쾌                 Ceftazidime
66553       59     18000.0   1.00          경쾌                  Ampicillin
66554       63      1125.0   7.00          경쾌  Trimethoprim/Sulfamethoxazole
66555       53      3000.0   8.00          경쾌                   Cefepime

      PREDICTED_RSLT_PROB PREDICTED_RSLT
0                0.080757              S
1                0.120574              S
2                0.226620              S
3                0.051607              S
```

```
4                0.010768              S
...                   ...            ...
66551            0.272694              S
66552            0.066530              S
66553            0.293385              S
66554            0.029105              S
66555            0.006919              S

[66556 rows x 7 columns]
예측 정확도: 0.678601478454234
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
import numpy as np

# 데이터 불러오기
test_data = pd.read_csv('C:/Users/Hesong/test/test_data_10.csv')

# 'DAY'는 수치형 데이터이므로 따로 선택
numeric_feature = ['TPRSC_CAPA', 'DAY']
X_numeric_test = test_data[numeric_feature]

# 'INGR_NM'을 Label Encoding (학습할 때 사용한 label_encoder를 재사용)
test_data['INGR_NM'] = label_encoder.transform(test_data['INGR_NM'])

# 'ANTBT_NM'과 'antibiotics' 컬럼을 원-핫 인코딩 (학습할 때 사용한 것과 동일한 방법 사용)
X_categorical_test = pd.get_dummies(test_data[categorical_features], columns=categorical_features)

# 입력 데이터 차원 확인
print("이전 입력 데이터 차원:", X_categorical_test.shape)  # (None, 128)

# 추가할 피처 개수
additional_features = 8

# 추가 피처를 0으로 초기화한 배열 생성
additional_features_array = np.zeros((X_categorical_test.shape[0], additional_features))

# 추가 피처를 입력 데이터에 연결
X_test = np.hstack((X_categorical_test, additional_features_array))

# 입력 데이터 차원 다시 확인
print("조정된 입력 데이터 차원:", X_test.shape)  # (None, 130)

# 모델을 사용하여 RSLT_CONT 예측
y_pred = model.predict(X_test)

# 예측 결과를 데이터프레임에 추가 (확률로 표시)
test_data['PREDICTED_RSLT_PROB'] = y_pred

# 확률을 'R' 또는 'S'로 변환하여 예측 결과를 데이터프레임에 추가
test_data['PREDICTED_RSLT'] = ['R' if prob >= 0.5 else 'S' for prob in y_pred]

# 예측 결과를 새로운 CSV 파일에 저장
test_data.to_csv('predicted_test_data_10.csv', index=False)

# 예측 결과 출력
print("예측한 데이터:₩n", test_data[['INGR_NM', 'TPRSC_CAPA', 'DAY', 'antibiotics', 'ANTBT_NM', 'PREDICTED_RSLT_PROB', 'PREDICTED_RSLT']])

# 예측 정확도 계산
accuracy = (test_data['PREDICTED_RSLT'] == test_data['RSLT_CONT']).mean()
print("예측 정확도:", accuracy)
```

```
이전 입력 데이터 차원: (66556, 125)
조정된 입력 데이터 차원: (66556, 133)
2080/2080 [==============================] - 1s 686us/step
예측한 데이터:
       INGR_NM  TPRSC_CAPA    DAY antibiotics                      ANTBT_NM  W
0           38       800.0  15.00          경쾌                      Imipenem
1           20       200.0   5.00          경쾌  Trimethoprim/Sulfamethoxazole
2           64       800.0   7.50          경쾌                     Oxacillin
3           59     13500.0   4.00          경쾌                    Penicillin
4           36      2000.0   8.00          경쾌    Amoxicillin/Clavulanic acid
...        ...         ...    ...         ...                           ...
66551       15      6000.0   2.00          완쾌                Ciprofloxacin
66552       59     18000.0   5.00          경쾌                    Aztreonam
66553       49      1500.0   5.00          경쾌                    Cefoxitin
66554       41      1800.0   2.00          경쾌                  Erythromycin
66555       59     18000.0   9.25          경쾌                    Gentamicin

       PREDICTED_RSLT_PROB PREDICTED_RSLT
0                 0.192335              S
1                 0.069324              S
2                 0.262294              S
```

```
3                0.444400            S
4                0.309737            S
...                   ...          ...
66551            0.610234            R
66552            0.307942            S
66553            0.098626            S
66554            0.197023            S
66555            0.037932            S

[66556 rows x 7 columns]
예측 정확도: 0.6800138229460905
```