COVID-19

국내 코로나 현황

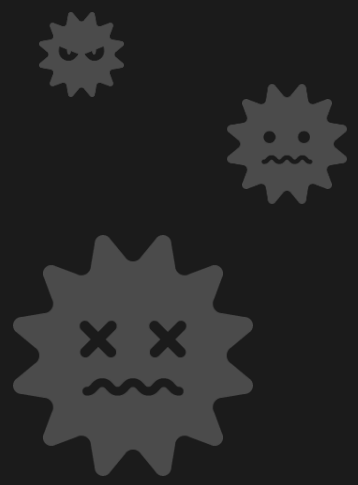
코로나 완치 기간 예측

산업경영공학과
20182890
임성민

파이썬 활용

박영호 교수님

#COVID-19



#목차

(1)
분석 주제

(2)
데이터 전처리

(3)
데이터
분석 및 결과

(4)
Q&A

1) 분석 주제

분석 주제

: **연령, 감염경로, 성별에 따른 코로나 완치 기간 예측**

연령

연령에 따라 코로나 완치 기간이 다를까?

가장 활발한 연령이 옮겨 가장 불편한 연령이 죽는다

성별

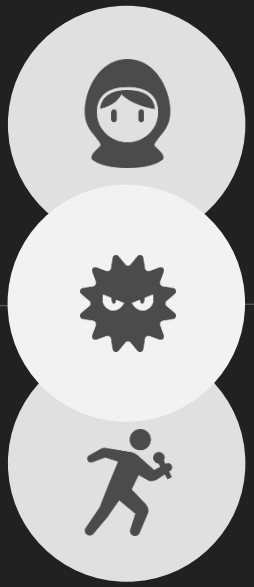
성별이 코로나 면역력 연구에 도움이 될까?

무차별적으로 걸리는 듯 하지만 남성은 감기에 덜 걸린다는 유사과학

감염경로

이태원, 신천지 어느 곳이 더 빨리 번질까?

용서 받지 못할 사람들 #고강도 사회적 거리두기



2) 데이터 전처리

사용한 데이터 원본

PatientInfo.csv (477.4 KB)

Detail | Compact | Column | 10 of 14 columns

country	province	city	infection_case	infected_by	
the country of the patient	the province of the patient	the city of the patient	the case of infection	the ID of who infected the patient	
orea 99%	Seoul 25%	Gyeongsan-si 12%	contact with patient 31%		
hina 0%	Gyeongsangbuk-do 24%	Seongnam-si 3%	[null] 18%		
ther (31) 1%	Other (2599) 50%	Other (4353) 84%	Other (2636) 51%	12.7k 7.00b	
orea	Seoul	Gangseo-gu	overseas inflow		75
orea	Seoul	Jungnang-gu	overseas inflow		31
orea	Seoul	Jongno-gu	contact with patient	2002000001	17
orea	Seoul	Mapo-gu	overseas inflow		9
orea	Seoul	Seongbuk-gu	contact with patient	1000000002	2
orea	Seoul	Jongno-gu	contact with patient	1000000003	43
orea	Seoul	Jongno-gu	contact with patient	1000000003	0
orea	Seoul	etc	overseas inflow		0

불러오기

```
import numpy as np
from pandas import DataFrame
import pandas as pd
df = pd.read_csv('C:/Users/smcom/Desktop/sm_20/python/free/new2/datasets_527325_1205308_PatientInfo.csv',sep=',')
df
```

	patient_id	global_num	sex	birth_year	age	country	province	city	disease	infection_case	infection_order	infected_by	contact_number
0	1000000001	2.0	male	1964	50s	Korea	Seoul	Gangseo-gu	NaN	overseas inflow	1.0	NaN	75
1	1000000002	5.0	male	1987	30s	Korea	Seoul	Jungnang-gu	NaN	overseas inflow	1.0	NaN	31
2	1000000003	6.0	male	1964	50s	Korea	Seoul	Jongno-gu	NaN	contact with patient	2.0	2002000001	17
3	1000000004	7.0	male	1991	20s	Korea	Seoul	Mapo-gu	NaN	overseas inflow	1.0	NaN	9
4	1000000005	9.0	female	1992	20s	Korea	Seoul	Seongbuk-gu	NaN	contact with patient	2.0	1000000002	2
...	...	...	...	...	...	...	...	...	...	...	...	...	...
3999	7000000010	NaN	female	NaN	20s	Korea	Jeju-do	Jeju-do	NaN	overseas inflow	NaN	NaN	18
4000	7000000011	NaN	male	NaN	30s	Korea	Jeju-do	Jeju-do	NaN	contact with patient	NaN	7000000009	5
4001	7000000012	NaN	female	NaN	20s	Korea	Jeju-do	Jeju-do	NaN	overseas inflow	NaN	NaN	9
4002	7000000013	NaN	female	NaN	10s	China	Jeju-do	Jeju-do	NaN	overseas inflow	NaN	NaN	6
4003	7000000014	NaN	female	NaN	30s	Korea	Jeju-do	Jeju-do	NaN	Itaewon Clubs	NaN	NaN	NaN

4004 rows × 18 columns

데이터 출처 https://www.kaggle.com/kimjihoo/coronavirusdataset?select=TimeAge.csv

원하는 데이터 뽑아서 확인하기

```
In [2]: sel = df.loc[:,['sex','city', 'infection_case',
                        'confirmed_date', 'released_date']]
        sel2 = df.loc[:,['infection_case', 'birth_year']]

In [3]: sel['diff_date'] = 100
        sel2['diff_year'] = 100

In [4]: sel.info()

        <class 'pandas.core.frame.DataFrame'>
        RangeIndex: 4004 entries, 0 to 4003
        Data columns (total 6 columns):
         #   Column          Non-Null Count  Dtype
        ---  ------          --------------  -----
         0   sex             3674 non-null   object
         1   city            3926 non-null   object
         2   infection_case  3211 non-null   object
         3   confirmed_date  4001 non-null   object
         4   released_date   1508 non-null   object
         5   diff_date       4004 non-null   int64
        dtypes: int64(1), object(5)
        memory usage: 187.8+ KB
```

2차데이터 분리 (diff_date 형성)

```
In [10]: sel['confirmed_date'] = sel['confirmed_date'].astype('datetime64[ns]')
         sel['released_date'] = sel['released_date'].astype('datetime64[ns]')

In [11]:
         sel['diff_date'] =   sel['released_date']-sel['confirmed_date']
         sel2['diff_year'] = 100
         sel2.dropna(inplace = True)
         sel2['birth_year'].dropna()
         sel2['birth_year'] = pd.to_numeric(sel2['birth_year'], errors='coerce')

In [12]: sel2['birth_year'].count()
         sel2

Out[12]:
```

	infection_case	birth_year	diff_year
0	overseas inflow	1964.0	100
1	overseas inflow	1987.0	100
2	contact with patient	1964.0	100
3	overseas inflow	1991.0	100
4	contact with patient	1992.0	100
...	...	...	...
3990	etc	1998.0	100
3991	etc	1998.0	100
3992	etc	1972.0	100
3993	etc	1974.0	100
3996	overseas inflow	1996.0	100

2559 rows × 3 columns

- 원본데이터의 컬럼은 18개 -> 필요한 컬럼 6개로 줄인다.
- 나중에 계산한 값이 들어갈 컬럼을 미리 생성해준다 (diff_date, diff_year)
- 연산을 편리하게 하기 위해 데이터를 한번 더 분리한다.
- 연산을 위해 null값 제거
- Object형태의 날짜 데이터를 datetime으로 바꿔주고 계산하여 diff_date에 넣는다.
- 연도만 존재하는 데이터는 float형태로 바꿔준다.

2) 데이터 전처리

연도데이터로 대상자의 연령을 확인하기

```
qwe = sel2['birth_year'] <= 2020.0
asd = sel2['birth_year'] > 2020.0
is_qwe = sel2[qwe]
is_asd = sel2[asd]
is_qwe.diff_year = 2020.0 - is_qwe.birth_year
is_asd.diff_year = 2020.0 - is_asd.birth_year
#df1 = pd.merge(is_qwe , is_asd, left_on = "infection_case", right_index=True)
df1 = pd.concat([is_qwe,is_asd],axis = 1)
df1.columns = ["infection_case","birth_year","diff_year","sd","df","fg"]
df1.drop(["sd","df","fg"], axis='columns', inplace=True)
df1.head(5)
```

```
C:₩Users₩smcom₩anaconda3₩lib₩site-packages₩pandas₩core₩generic.py:5303: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self[name] = value
```

	infection_case	birth_year	diff_year
0	overseas inflow	1964.0	56.0
1	overseas inflow	1987.0	33.0
2	contact with patient	1964.0	56.0
3	overseas inflow	1991.0	29.0
4	contact with patient	1992.0	28.0

concat()으로 데이터를 붙여주고 기초데이터 완성

```
df = pd.concat([sel,sel2,df1],axis = 1 , sort = False)
df.columns = ["sex","city","infection_case","confirmed_date","released_date","diff_date","xxx","birth_year","sd","df","qw","diff_year"]
df.drop(["xxx","confirmed_date","released_date","sd","df","qw","birth_year"], axis='columns', inplace=True)
df
```

	sex	city	infection_case	diff_date	diff_year
0	male	Gangseo-gu	overseas inflow	13 days	56.0
1	male	Jungnang-gu	overseas inflow	32 days	33.0
2	male	Jongno-gu	contact with patient	20 days	56.0
3	male	Mapo-gu	overseas inflow	16 days	29.0
4	female	Seongbuk-gu	contact with patient	24 days	28.0
...	...	...	...	...	...
3999	female	Jeju-do	overseas inflow	18 days	NaN
4000	male	Jeju-do	contact with patient	NaT	NaN
4001	female	Jeju-do	overseas inflow	32 days	NaN
4002	female	Jeju-do	overseas inflow	12 days	NaN
4003	female	Jeju-do	Itaewon Clubs	NaT	NaN

4004 rows × 5 columns

데이터 출처 https://www.kaggle.com/kimjihoo/coronavirusdataset?select=TimeAge.csv

데이터 확인하기

```
In [19]: df.info()
         #결측값 무시하고 진행
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 4004 entries, 0 to 4003
Data columns (total 7 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   sex             4004 non-null   int32
 1   city            3926 non-null   object
 2   infection_case  3211 non-null   object
 3   date            4004 non-null   int64
 4   age             4004 non-null   int64
 5   diff_date       1508 non-null   float64
 6   diff_year       2558 non-null   float64
dtypes: float64(2), int32(1), int64(2), object(2)
memory usage: 234.6+ KB
```

성별, 연령과 날짜 데이터를 값으로 표현

```
In [17]: df.sex = (df.sex == "female").astype(int)#10l 여성, 00l 남성
         df['age'] = 100
         df['age'] = np.where(df['diff_year'] >= 0, 0, df['age']) # 0~10
         df['age'] = np.where(df['diff_year'] >= 10, 10, df['age']) # 10~20
         df['age'] = np.where(df['diff_year'] >= 20 , 20, df['age']) # 20~30
         df['age'] = np.where(df['diff_year'] >= 30 , 30, df['age']) # 30~40
         df['age'] = np.where(df['diff_year'] >= 40 , 40, df['age']) # 40~all
         df.diff_date = df.diff_date / np.timedelta64(1, 'D')
         df['date'] = 100
         df['date'] = np.where(df['diff_date'] >= 0 , 0, df['date']) # 0~10
         df['date'] = np.where(df['diff_date'] >= 10 , 10, df['date']) # 0~10
         df['date'] = np.where(df['diff_date'] >= 20 , 20, df['date']) # 10~20
         df['date'] = np.where(df['diff_date'] >= 30 , 30, df['date'])  # 20~30
```

```
In [20]: df = df[['sex','city','infection_case','date','age','diff_date','diff_year']]
         dfs = df.dropna()
         dfs.head(5)
```

Out[20]:

	sex	city	infection_case	date	age	diff_date	diff_year
0	0	Gangseo-gu	overseas inflow	10	40	13.0	56.0
1	0	Jungnang-gu	overseas inflow	30	30	32.0	33.0
2	0	Jongno-gu	contact with patient	20	40	20.0	56.0
3	0	Mapo-gu	overseas inflow	10	20	16.0	29.0
4	1	Seongbuk-gu	contact with patient	20	20	24.0	28.0

- 2020년을 기준으로 논리연산을 이용해 diff_year라는 이름으로 감염자들의 연령을 계산한다.
- 이용할 모든 데이터를 concat()함수로 붙여준다. 이때 컬럼명과 위치도 바꿈
- 데이터의 최종형태를 확인한다.
- diff_date를 숫자형으로 바꿔주고, 완치일자와 연령대, 성별을 묶어서 표현해준다.
  - 모든 null값을 제거한 최종 데이터 갯수

679 rows × 7 columns

3) 데이터 분석

1차 분석

사용한 분석기법 : svm (비선형 분류)

입력변수 : age, sex

출력변수 : date

주제 : 성별과 나이로 완치 기간을 얼마나 예측 할 수 있을까?

출력된 acuracy

: 0.30514705882352944

결론 : 예측이 거의 불가 하다고 할 수 있다.

```
x = dfs[['age','sex']]
x.std = StandardScaler().fit_transform(x)
y = dfs['date']
```

```
### Train & test data
x_train, x_test,y_train, y_test = train_test_split(x,y,test_size = 0.4)
```

```
### SVM
svc = SVC(kernel = 'rbf', C = 100, gamma=0.01)
model = svc.fit(x_train,y_train)
```

```
### 예측
y_pred = model.predict(x_test)
y_pred
```

```
array([20, 20, 20, 20, 10, 10, 10, 20, 20, 10, 10, 10, 10, 10, 20, 10, 20,
       20, 10, 10, 10, 20, 20, 20, 20, 10, 20, 10, 10, 10, 20, 10, 20, 10,
       10, 10, 10, 10, 10, 20, 20, 10, 10, 20, 20, 10, 10, 10, 10, 10, 10,
       10, 10, 20, 10, 20, 20, 10, 20, 20, 10, 10, 10, 20, 10, 10, 20, 20,
       20, 20, 20, 10, 20, 20, 20, 10, 10, 20, 10, 10, 10, 20, 10, 20, 20,
       10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 20, 20, 20, 10, 10, 10, 20,
       20, 20, 10, 10, 10, 10, 10, 10, 10, 20, 20, 20, 20, 10, 20, 20, 10,
       10, 10, 10, 10, 20, 20, 20, 20, 10, 20, 20, 10, 10, 20, 10, 10, 20,
       20, 10, 20, 10, 10, 10, 10, 10, 10, 20, 10, 10, 10, 10, 10, 10, 10,
       20, 10, 20, 10, 20, 10,  0, 10, 10, 10, 10, 20, 10, 10, 10, 20, 20,
       20, 20, 20, 20, 10, 10, 10, 10, 20, 20, 20, 20, 20, 10, 20, 10, 10,
       10, 10, 10, 20, 10, 10, 20, 10, 10, 10, 20, 20, 20, 20, 10, 10, 10,
       20, 20, 10, 20, 10, 20, 10, 20, 10, 10, 10, 10, 10, 10, 10, 20, 20,
       10, 20, 20, 10, 10, 10, 10, 10, 10, 10, 20, 20, 20, 10, 10, 10, 20,
       10, 20, 20, 10, 20, 10, 10, 20, 20, 10, 10, 10, 10, 20, 20, 10, 20,
       10, 10, 20, 20, 20, 10, 10, 10, 10, 10, 10, 10, 20, 20, 10, 10, 20],
      dtype=int64)
```

```
### 교차표
pd.crosstab(y_test, y_pred)
```

col_0	0	10	20
date			
0	0	21	10
10	0	53	39
20	0	57	30
30	1	30	31

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        31
          10       0.33      0.58      0.42        92
          20       0.27      0.34      0.30        87
          30       0.00      0.00      0.00        62

    accuracy                           0.31       272
   macro avg       0.15      0.23      0.18       272
weighted avg       0.20      0.31      0.24       272
```

```
C:₩Users₩smcom₩anaconda3₩lib₩site-packages₩sklearn₩metrics₩_classification.py:1272: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

```
### acuracy
model.score(x_test,y_test)
```

```
0.30514705882352944
```

3) 데이터 분석

2차 분석

사용한 분석기법 : svm (비선형 분류)

입력변수 : date 외 전부

출력변수 : date

주제 : 성별과 나이, 접촉경로로 완치 기간을 얼마나 예측 할 수 있을까?

출력된 acuracy

: 0.37209302325581395

결론 :1차 분석보다는 유의미한 결과를 낳았으나, 참고자료 정도가 적합한 활용방안이다.

	date	sex	age	infection_case_contact with patient	infection_case_etc	infection_case_Eunpyeong St. Mary's Hospital	infection_case_Geochang Church	infection_case_Guro-gu Call Center	infection_case_Gye Cham Joeun Com
0	20	0	40	1	0	0	0	0	
1	10	1	40	1	0	0	0	0	
2	10	0	20	1	0	0	0	0	
3	20	0	20	0	0	0	0	0	
4	20	0	20	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...
423	10	1	40	1	0	0	0	0	
424	30	1	40	1	0	0	0	0	
425	10	0	40	1	0	0	0	0	
426	20	0	20	0	1	0	0	0	
427	0	0	20	0	1	0	0	0	

```
x = aa.iloc[:,1:]
x.std = StandardScaler().fit_transform(x)
y = aa['date']
```

```
### Train & test data
x_train, x_test,y_train, y_test = train_test_split(x,y,test_size = 0.4)
```

```
### SVM
svc = SVC(kernel = 'rbf', C = 100, gamma=0.01)
model = svc.fit(x_train,y_train)
```

```
### 예측
y_pred = model.predict(x_test)
y_pred
```

```
array([20, 30, 20, 20, 10, 20, 20, 30, 10, 20, 10, 20, 10, 30, 30, 20, 30,
       20, 20, 10, 30, 10, 30, 20, 30, 20, 20, 20, 30, 30, 20, 20, 20, 30,
       30, 10, 20, 20, 30, 20, 20, 30, 20, 30, 30, 30, 20, 20, 20, 10, 20,
       10, 30, 30, 20, 30, 10, 20, 30, 20, 30, 30, 10, 10, 30, 30, 10, 30,
       30, 20, 30, 20, 30, 20, 20, 10, 20, 30, 10, 30, 30, 10,  0, 30, 20,
       20, 20, 20, 20, 30, 20, 20, 10, 10, 20, 20, 10,  0, 20, 10, 10, 30,
       30, 10, 10, 30, 10, 30, 20, 20, 20, 20, 20, 20, 30, 30, 30, 10, 20,
       30, 10, 20, 20, 10, 20, 30, 20, 10, 30, 10, 20, 20, 30, 30, 30, 10,
       30, 20, 30, 10, 30, 20, 20, 30, 30, 20, 10, 30, 10, 10, 30, 20, 10,
       20, 10, 20, 30, 10,  0, 20, 20, 10, 20, 20, 10, 20, 30, 20, 20, 30,
       10, 20], dtype=int64)
```

```
### 교차표
pd.crosstab(y_test, y_pred)
```

col_0	0	10	20	30
date				
0	0	5	3	4
10	1	12	22	11
20	1	11	25	15
30	1	12	22	27

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00        12
          10       0.30      0.26      0.28        46
          20       0.35      0.48      0.40        52
          30       0.47      0.44      0.45        62

    accuracy                           0.37       172
   macro avg       0.28      0.29      0.28       172
weighted avg       0.36      0.37      0.36       172
```

```
### acuracy
model.score(x_test,y_test)
```

```
0.37209302325581395
```

3) 데이터 분석

3차 분석

사용한 분석기법 : 로지스틱 회귀분석

입력변수 : date 외 전부

출력변수 : date

주제 : 성별과 나이, 접촉경로로 완치 기간을 얼마나 예측 할 수 있을까?

출력된 acuracy

: 0.45348837209302323

결론 : 성별과 나이, 접촉경로로 완치기간을 정확히 예측 할 수는 없지만 1,2차 분석 보다 더 높은 예측율을 보였다.

```
x = aa.iloc[:,1:]
y = aa['date']
y,y_levels = pd.factorize(y)
```

```
y_levels
```

```
Int64Index([20, 10, 30, 0], dtype='int64')
```

```
### Train & test data
x_train, x_test,y_train, y_test = train_test_split(x,y,test_size = 0.4)
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
### logistic regression
logistic = LogisticRegression()
model = logistic.fit(x_train, y_train)
```

```
### 교차표
pd.crosstab(y_test, y_pred)
```

col_0	0	10	20	30
row_0				
0	2	11	31	10
1	2	11	20	15
2	1	13	32	14
3	0	4	4	2

In [84]:

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.40      0.04      0.07        54
           1       0.00      0.00      0.00        48
           2       0.00      0.00      0.00        60
           3       0.00      0.00      0.00        10
          10       0.00      0.00      0.00         0
          20       0.00      0.00      0.00         0
          30       0.00      0.00      0.00         0

    accuracy                           0.01       172
   macro avg       0.06      0.01      0.01       172
weighted avg       0.13      0.01      0.02       172
```

```
### acuracy
model.score(x_test,y_test)
```

```
0.45348837209302323
```

Q & A

| 코로나 완치 기간 예측

감사합니다

| 코로나 완치 기간 예측

파이썬 활용

박영호 교수님 분석 과제