여행업 현황과 코로나19의 관련성 분석

김성희

목 차

Ⅰ. 프로젝트 개요

Ⅱ. 탐색적 데이터 분석

Ⅲ. 기계학습 분석

Ⅳ. 향후 계획

Ⅰ. 프로젝트 개요

1. 탐색적 분석

전국 지자체의 '국내외여행업 인허가' 데이터를 통해서
여행업의 영업 현황과 위치 현황을 분석해 본다

2. 기계학습 분석

'국내외여행업 인허가' 데이터와 '코로나19 시도별 발생동향' 데이터를
통하여 여행업과 코로나 발생간의 관계를 분석해본다

분석 환경

◇ 분석 툴

- Python 3.9.7
- Visual Studio Code

◇ Local PC

- CPU : Intel® i5-6400 CPU @ 2.70GHz
- RAM: 16.0 GB
- OS: Windows10 Home 64bit

◇ 사용 라이브러리

- matplotlib 3.2.1
- numpy 1.18.1
- pandas 1.0.3
- scikit-learn 0.23.1

Ⅱ. 탐색적 데이터 분석

1. 데이터 개요
2. 데이터 전처리
3. 탐색적 분석
4. 결론

1. 데이터 개요

1. 데이터 출처

지방행정 인허가 데이터 개방 (https://www.localdata.go.kr/main.do)

- 데이터 다운로드/ 업종 다운로드/ 문화/ 여행 / 국내외여행업

2. 데이터 개요

전국 지자체의 '국내외여행업' 인허가 데이터

- 기준일 : 1978-03-10 ~ 2022-6-02
- 데이터 수 : 19,998건
- 컬럼 수 : 63개

2. 데이터 전처리 – 결측값 처리

1. 결측값 조사

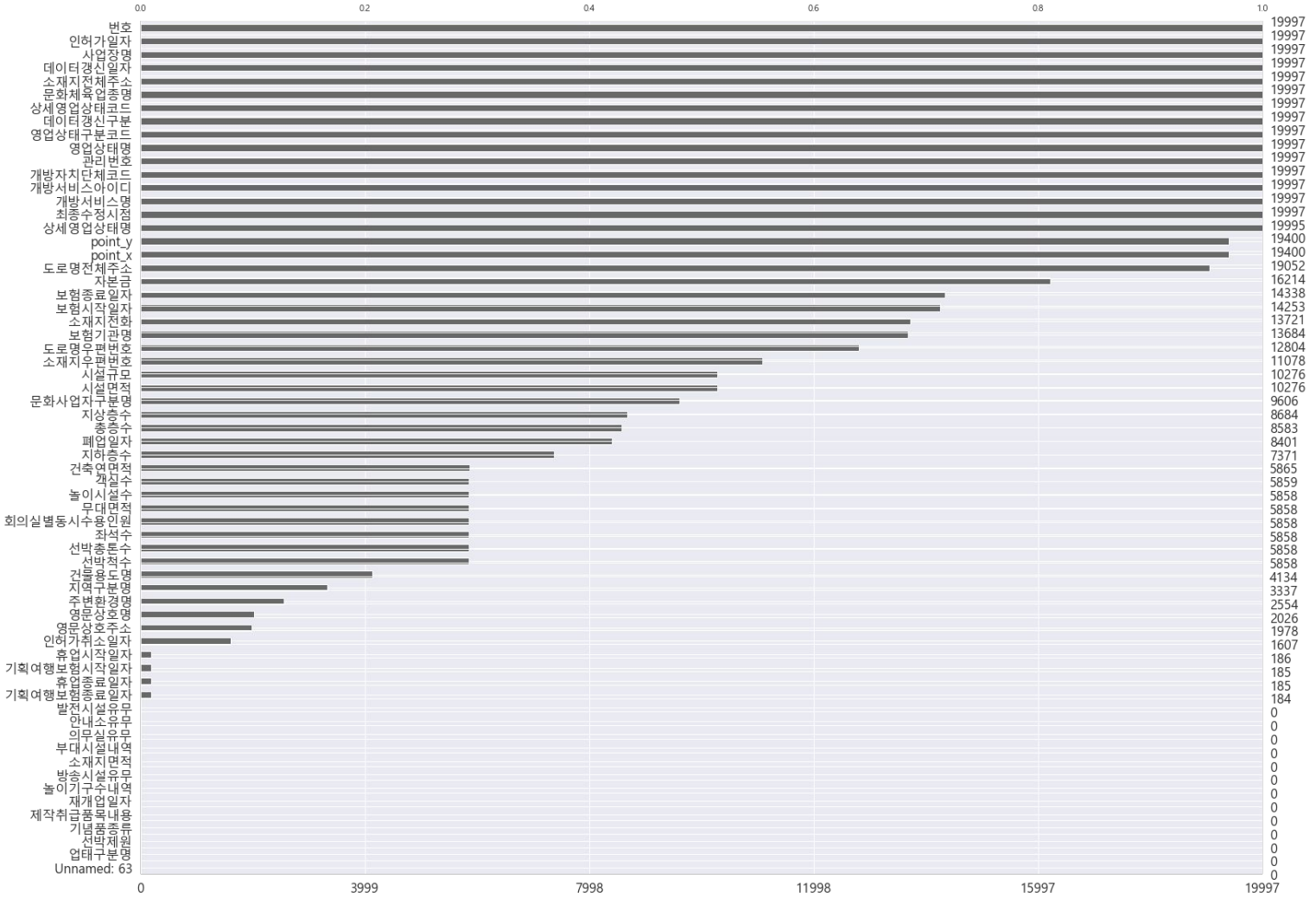
```
msno.bar(df_origin, sort= ' ascending ' )
```

2. 결측값 처리

1) 결측값이 20%이상인 컬럼 제거
(단, 인허가 관련 날짜는 제외)

2) 자본금의 결측값은 숫자 0으로 입력

3) 좌표 컬럼의 결측값은
위치 분석시에 제거하고 분석



2. 데이터 전처리–중복 데이터 처리

1. 동일 정보가 들어 있는 컬럼 삭제

– 개방서비스명, 개방서비스아이디, 문화사업자구분명, 보험기관명

2. 주소 관련 컬럼 통합

개방자치단체코드, 소재지전화, 소재지우편번호,
소재지전체주소, 도로명전체주소, 도로명우편번호

=> “**소재지**” 컬럼을 새로 만들어서 ‘특별시, 광역시, 도’ 정보만 저장

– 엑셀 함수 처리

=LEFT(소재지전체주소,SEARCH(" ", 소재지전체주소)–1)

2. 데이터 전처리– 파생변수 추가

1. 영업상태변화일자

인허가 관련 날짜를 하나의 컬럼에 통합

: '영업상태구분코드'에 따라 다른 인허가 날짜를 통합

1 (영업중) : 2022-06-02(최종시점날짜) 입력

2 (휴업) : 휴업시작일자 입력 (NULL일 경우 최종수정시점 입력)

3 (폐업) : 폐업일자 입력 (NULL일 경우 최종수정시점 입력)

4 (취소/말소/만료/정지/중지) : 인허가취소일자 입력 (NULL일 경우 최종수정시점 입력)

–엑셀 함수로 처리 :

=IF(코드 =1,"2022-06-02",IF(코드 =2,IF(ISBLANK(휴업시작일자), 최종수정시점, 휴업시작일자),IF(코드=3,IF(ISBLANK(폐업일자), 최종수정시점, 폐업일자),IF(코드=4,IF(ISBLANK(인허가취소일자), 최종수정시점, 인허가취소일자)))))

2. 데이터 전처리– 파생변수 추가

2. 인허가연도, 영업상태변화연도

‘인허가일자’, ‘영업상태변화일자’에서 연도만 추출해서 저장

–엑셀에서 함수로 처리 : 날짜형식에서 연도만 추출

=YEAR()

3. 영업 연수

‘인허가일’과 ‘영업상태변화일자’를 이용해서 영업연수 구하기

–엑셀에서 함수로 처리 :

=DATEDIF(인허가일자, 영업상태변화일자,"Y")

3. 탐색적 데이터 분석 – 영업현황

1. 분석할 데이터 개요

df.info()

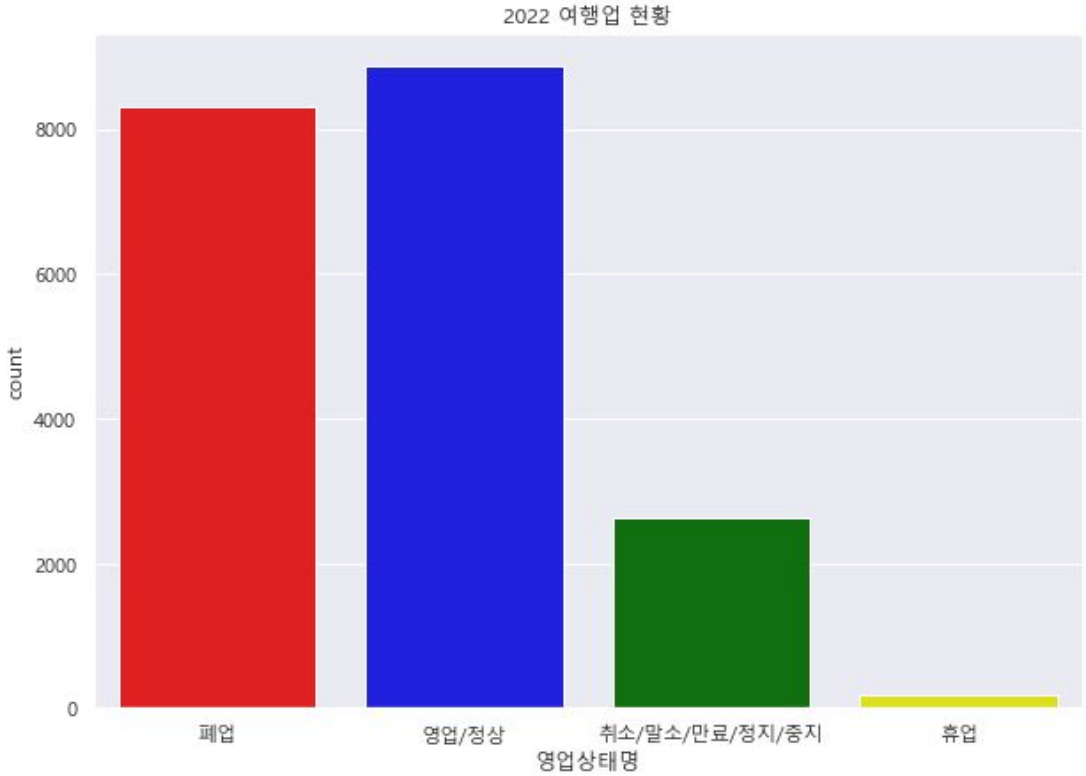
- 데이터 수 : 19,997 건
- 컬럼 수 : 13개

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19997 entries, 0 to 19996
Data columns (total 13 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   인허가일자     19997 non-null  object
 1   인허가연도     19997 non-null  int64
 2   영업상태구분코드  19997 non-null  int64
 3   영업상태명     19997 non-null  object
 4   영업상태변화일자  19997 non-null  object
 5   영업상태변화연도  19997 non-null  int64
 6   영업연수      19997 non-null  int64
 7   상세영업상태코드  19997 non-null  object
 8   상세영업상태명   19995 non-null  object
 9   소재지       19997 non-null  object
 10  point_x   19400 non-null  float64
 11  point_y   19400 non-null  float64
 12  자본금       19997 non-null  int64
dtypes: float64(2), int64(5), object(6)
memory usage: 2.0+ MB
```

3. 탐색적 데이터 분석 – 영업현황

2. 1978년~ 2022년 영업상태별 영업 현황

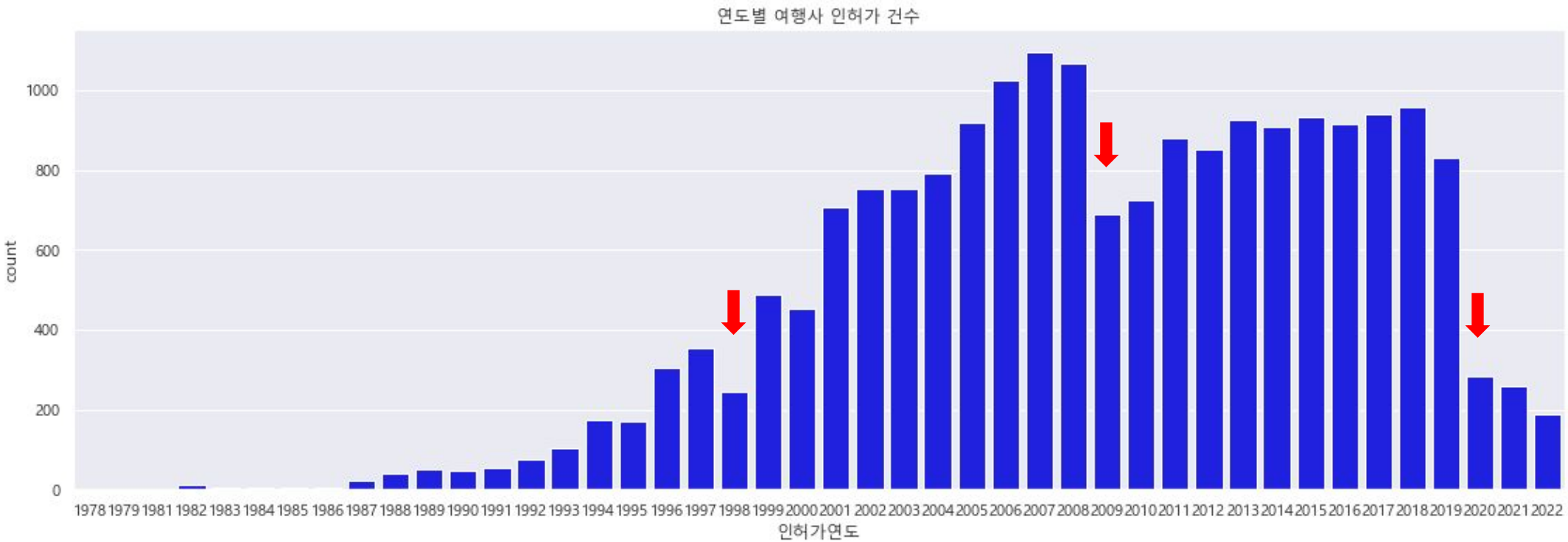
총 19,997 개의 여행사 중에서
영업/정상 : 8872개 , 폐업 : 8313개 , 취소/말소/만료/정지/중지 : 2636개 , 휴업 : 176개



3. 탐색적 데이터 분석 – 영업현황

3. 연도별 영업 상태 변화

1) 신규 여행사

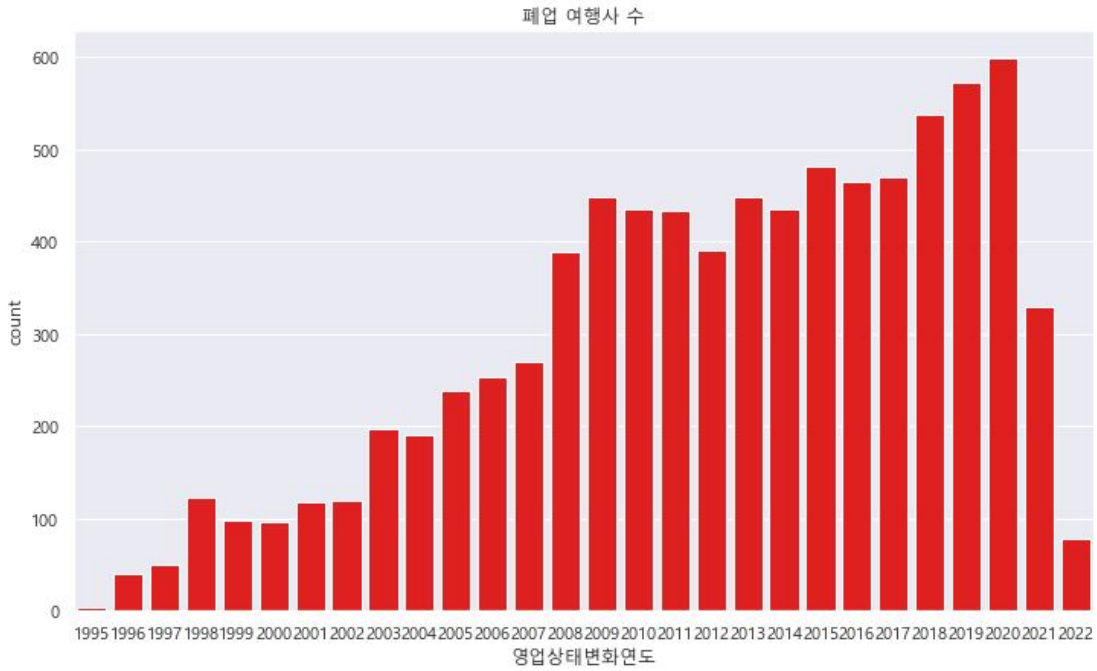


1987년 부터 2019년까지는 전반적으로 우상향하고 있음.
1998년(외환위기), 2009년(서브프라임 사태), 2020년(코로나19)에는 급격히 감소함.

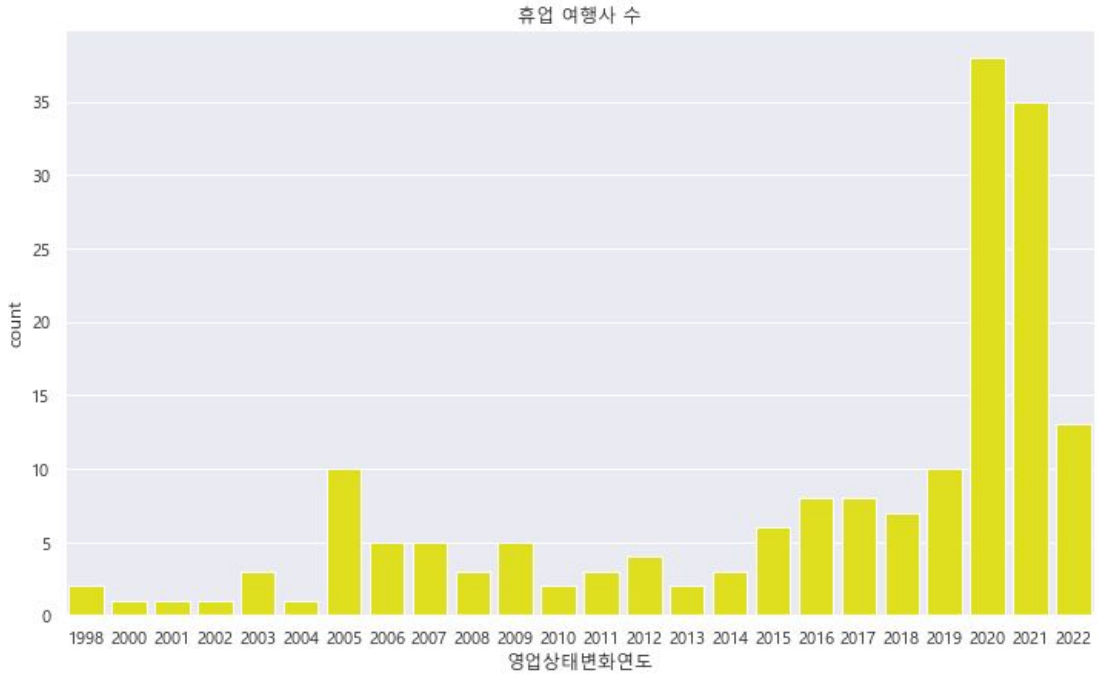
3. 탐색적 데이터 분석 – 영업현황

3. 연도별 영업 상태 변화

2) 폐업한 여행사



3) 휴업한 여행사



폐업한 여행사는 2020년까지 꾸준히 증가하다가 2021년과 2022년 급격히 감소함.
휴업한 여행사는 2019년까지는 10곳 이내였으나 2020년과 2021년 급격히 증가하였음.

3. 탐색적 데이터 분석 – 영업현황

3. 연도별 영업 상태 변화

4) 취소/말소/만료/정지/중지한 여행사

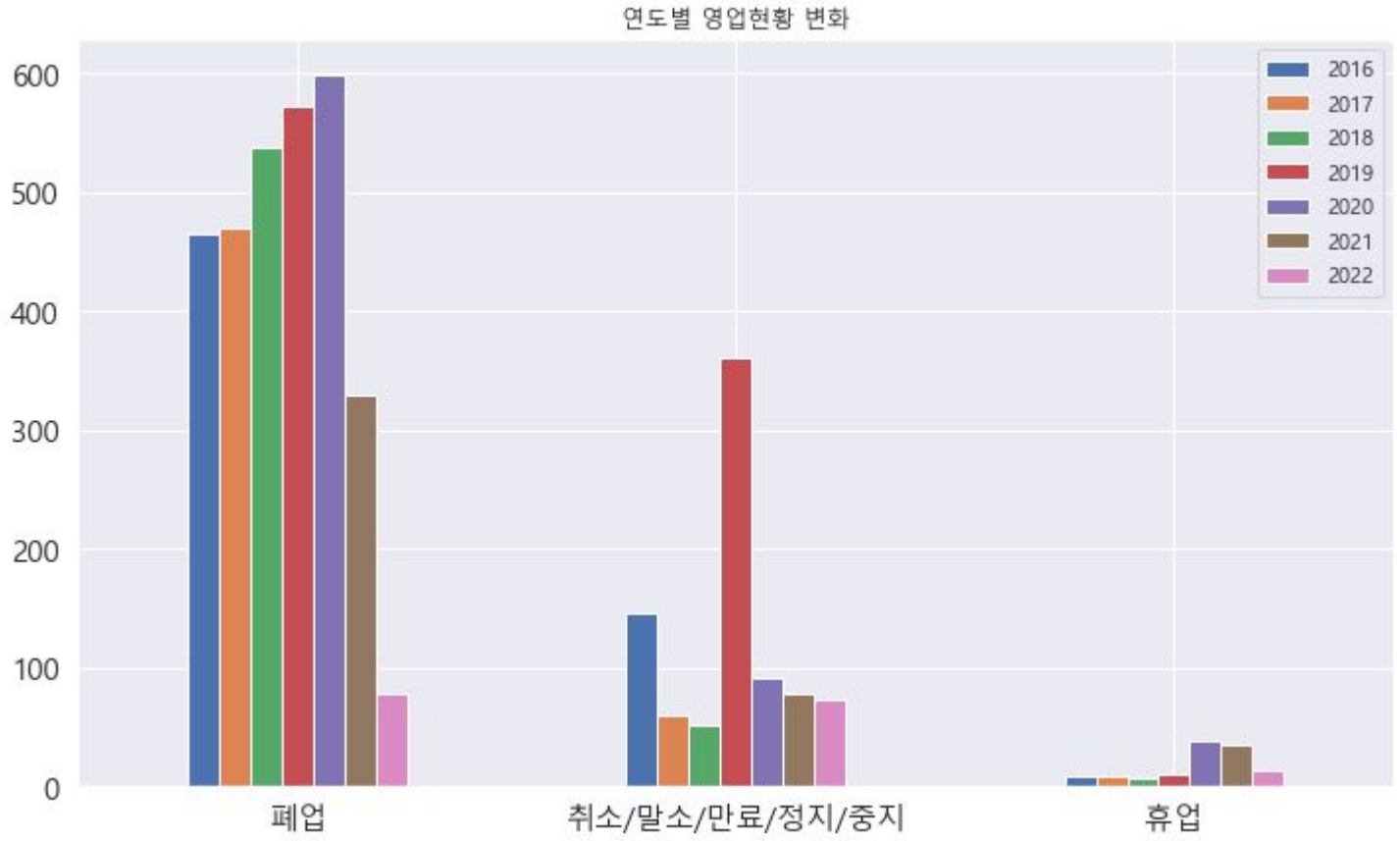
취소/말소/만료/정지/중지한 여행사는 2019년 갑자기 급격히 증가하였다가 2020년 급격히 감소함.
폐업, 휴업 여행사가 2020년 최고치를 기록한 것에 영향을 받은것으로 보여짐.

3. 탐색적 데이터 분석 – 영업현황

3. 연도별 영업 상태 변화

4) 연도별 영업상태별 변화 비교

(코로나19 팬데믹이었던 2020년을 기점으로 전후 3년간 데이터)



인허가 건수는 항상 폐업 상태가 압도적으로 많았으나,

2022년 상반기는
폐업(78건)과
취소/말소/만료/정지/중지(74건)
이
비슷함

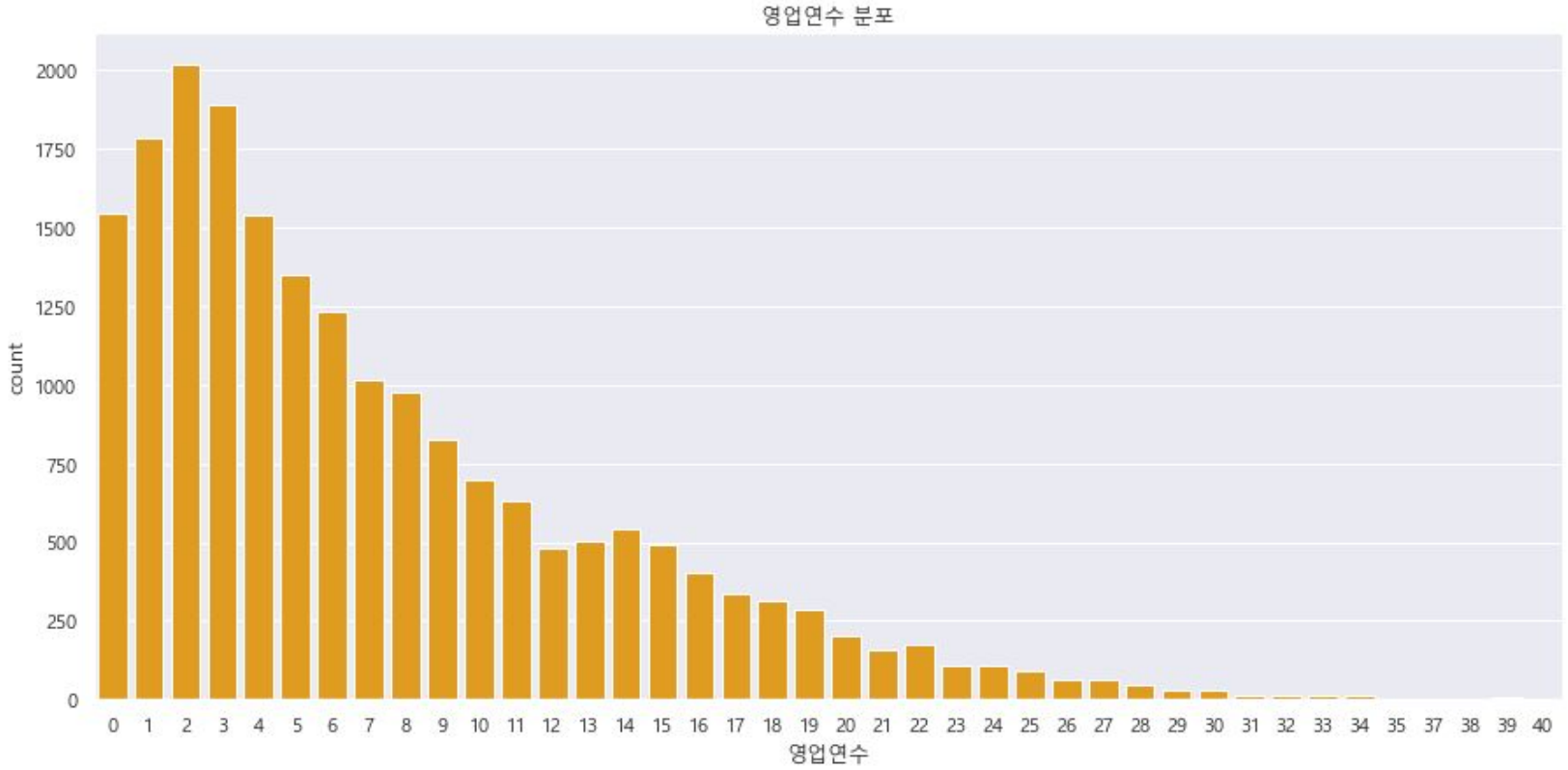
3. 탐색적 데이터 분석 – 영업현황

4. 영업연수 비교

1) 영업연수 평균 : 7년
df['영업연수'].mean()
= 7.285192778916837

2) 영업연수 최대값 : 40년
df['영업연수'].max()

3) 영업연수 최빈값 : 2년
df['영업연수'].mode()



3. 탐색적 데이터 분석 – 위치 현황

1.좌표정보 데이터의 결측값 처리

1) ’좌표정보(x)', '좌표정보(y)‘ 의 결측값 조사

df['좌표정보(x)'].isnull().value_counts()

- False 19400
- True 597

2) 결측값이 있는 597 행 제거

df.dropna(subset=['좌표정보(x)'])

df.shape

(19400, 11)

3. 탐색적 데이터 분석 – 위치 현황

2. 좌표 변환

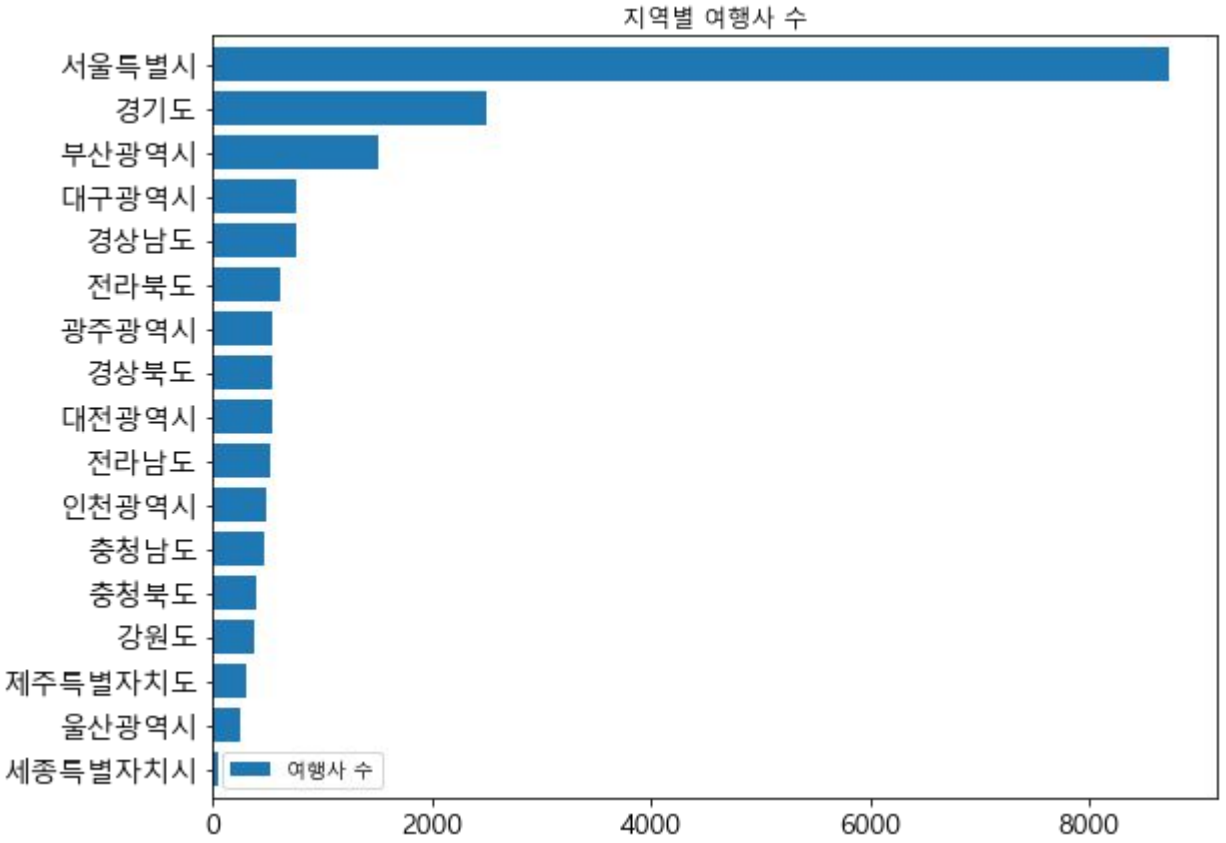
UTM-K 좌표를 WGS1984좌표로 전환하기 (folium 사용을 위해)

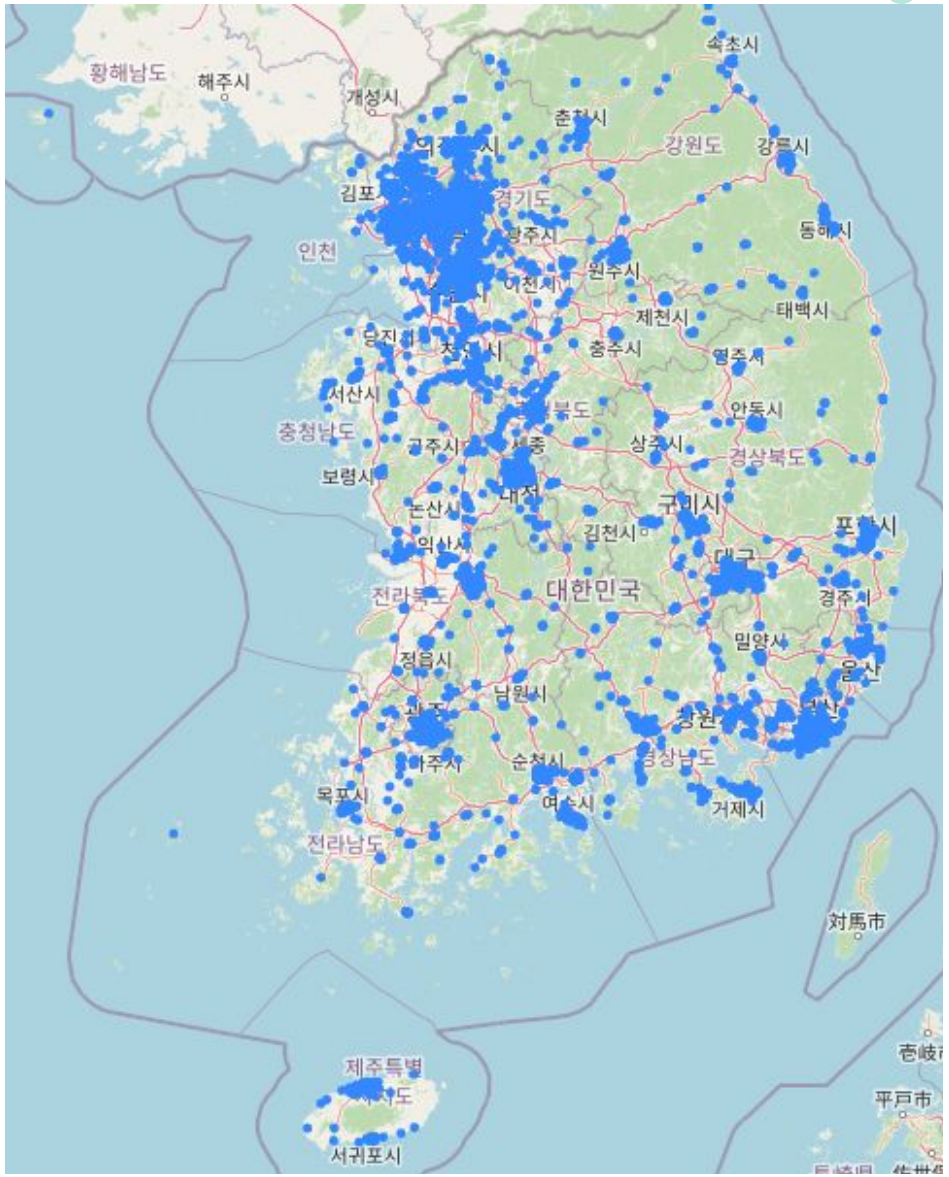
```
from pyproj import Proj, transform
import pandas   as pd

# Projection 정의
# UTM-K
proj_UTMK = Proj(init='epsg:2097') # UTM-K(Bassel) 도로명주소 지도 사용 중

# WGS1984
proj_WGS84 = Proj(init='epsg:4326') # Wgs84 경도/위도, GPS사용 전지구 좌표

# x, y 컬럼을 이용하여 UTM-K좌표를 WGS84로 변환한 Series데이터 반환
def transform_utmk_to_w84(df):
    return pd.Series(transform(proj_UTMK, proj_WGS84, df['point_x'], df['point_y']), index=['x',
'y'])



# 좌표 전환 함수 적용
df_area[['x_w84', 'y_w84']] = df_area.apply(transform_utmk_to_w84, axis=1)
```

3. 탐색적 데이터 분석 – 위치 현황

3. 여행사 위치 분석

서울에 위치한 여행사 수(8734개)가 압도적으로 많다.
경기도에 위치한 여행사 수(2498개)의 3.49배





4. 탐색적 데이터 분석 – 결론

- 신규 여행사 인허가 신청은 경제위기 상황에서 급격히 감소한다.
- 2020년까지 폐업하는 여행사 수가 꾸준히 증가하고 있다.
- 코로나 19의 영향으로 2020년, 2021년에는 휴업한 여행사 수가 급격히 증가했다.
- 여행사의 평균 영업 연수는 7년 정도이지만, 2년 정도 영업하는 여행사 수가 가장 많았다.
- 여행사의 위치는 서울이 8734개로 경기도 2498개의 3.49배로 압도적으로 많다.

Ⅲ. 기계학습 분석

1. 데이터 개요
2. 데이터 전처리
3. 기계학습
    1) 로지스틱회귀분석
    2) 의사결정나무
    3) 랜덤 포레스트
    4) KNN
    5) SVM
    6) ANN
    7) keras 신경망
4. 결론

1. 데이터 개요

1. 전국 지자체의 '국내외여행업' 인허가 데이터

2. 코로나19 시도별발생동향 – 확진환자 정보
   - 출처 : 보건복지부 코로나바이러스감염증-19
     http://ncov.mohw.go.kr/bdBoardList_Real.do?brdId=1&brdGubun=13&ncvContSeq=&contSeq=&board_id=&gubun=
   ※ 발생률 : 인구 10만 명당 (지역별 인구 출처 : 행정안전부, 주민등록인구현황 ('21.12월 기준))

3. 시도별 인구수
   - 출처 : 행정안전부-주민등록 인구통계
     https://jumin.mois.go.kr/index.jsp

2. 데이터 전처리

1. ‘국내외여행업 인허가’ 데이터의 소재지 정보를 기준으로 ‘코로나19 시도별발생동향’, ‘시도별 인구수’ 조인
2. 자본금이 0인 필드는 평균값으로 대체
3. 분석에 필요한 컬럼과 영업상태가 ‘영업’ , ‘폐업’ 인 데이터만 추출
4. 파생변수 result를 추가하고 영업=0 , 폐업=1 의 분류 데이터 추가
5. 언더샘플링

	좌표정보(x)	좌표정보(y)	국외영업	확진	사망률	발생률	인구수	capital	result
0	212436.8069	455486.1215	1	5285963	0.001202	38966	13577808	115227483.2	0
1	211505.6400	455454.0636	1	5285963	0.001202	38966	13577808	115234378.9	0
2	212357.0935	455436.2616	1	5285963	0.001202	38966	13577808	115241275.5	0

3. 기계학습 – 1)로지스틱 회귀분석

1. 모델 요약

1) 모델 설명력 : 15.6%

2) 유의하지 않은 변수

:좌표정보(x), capital

3) 상관관계가 있는 변수:

: 국외영업(–2.8) , 사망률(+243.5)

Model:	Logit	Pseudo R-squared:	0.156
Dependent Variable:	result	AIC:	19455.5951
Date:	2022-07-29 18:25	BIC:	19517.3401
No. Observations:	16616	Log-Likelihood:	-9719.8
Df Model:	7	LL-Null:	-11517.
Df Residuals:	16608	LLR p-value:	0.0000
Converged:	1.0000	Scale:	1.0000
No. Iterations:	8.0000		

	Coef.	Std.Err.	z	P>|z|	[0.025	0.975]
좌표정보(x)	-0.0000	0.0000	-1.4030	0.1606	-0.0000	0.0000
좌표정보(y)	-0.0000	0.0000	-6.3827	0.0000	-0.0000	-0.0000
국외영업	-2.8727	0.2041	-14.0772	0.0000	-3.2726	-2.4727
확진	-0.0000	0.0000	-25.6450	0.0000	-0.0000	-0.0000
사망률	243.5771	90.4599	2.6927	0.0071	66.2790	420.8752
발생률	0.0000	0.0000	8.5035	0.0000	0.0000	0.0000
인구수	0.0000	0.0000	25.0892	0.0000	0.0000	0.0000
capital	0.0000	0.0000	0.9159	0.3597	-0.0000	0.0000

3. 기계학습 – 1)로지스틱 회귀분석

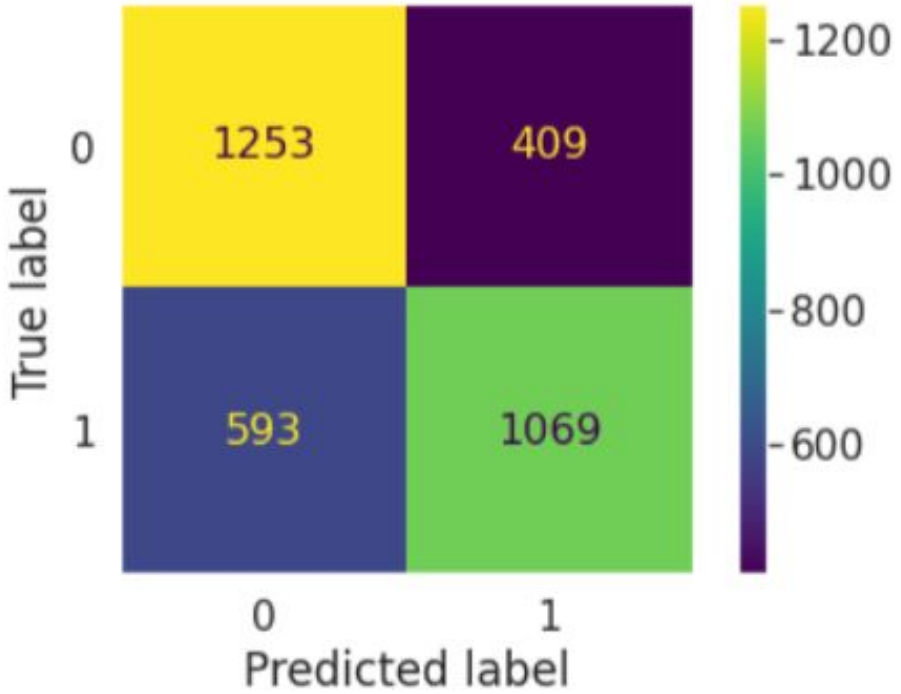
2. 모델 정확도

- 학습용 : 0.6914685525127896
- 검증용 : 0.6985559566787004

```
11 print("학습용 :",model.score(X_train, y_train))
12 print("검증용 :",model.score(X_test, y_test))
```

```
학습용 : 0.6914685525127896
검증용 : 0.6985559566787004
```

3.confusion matrix 출력



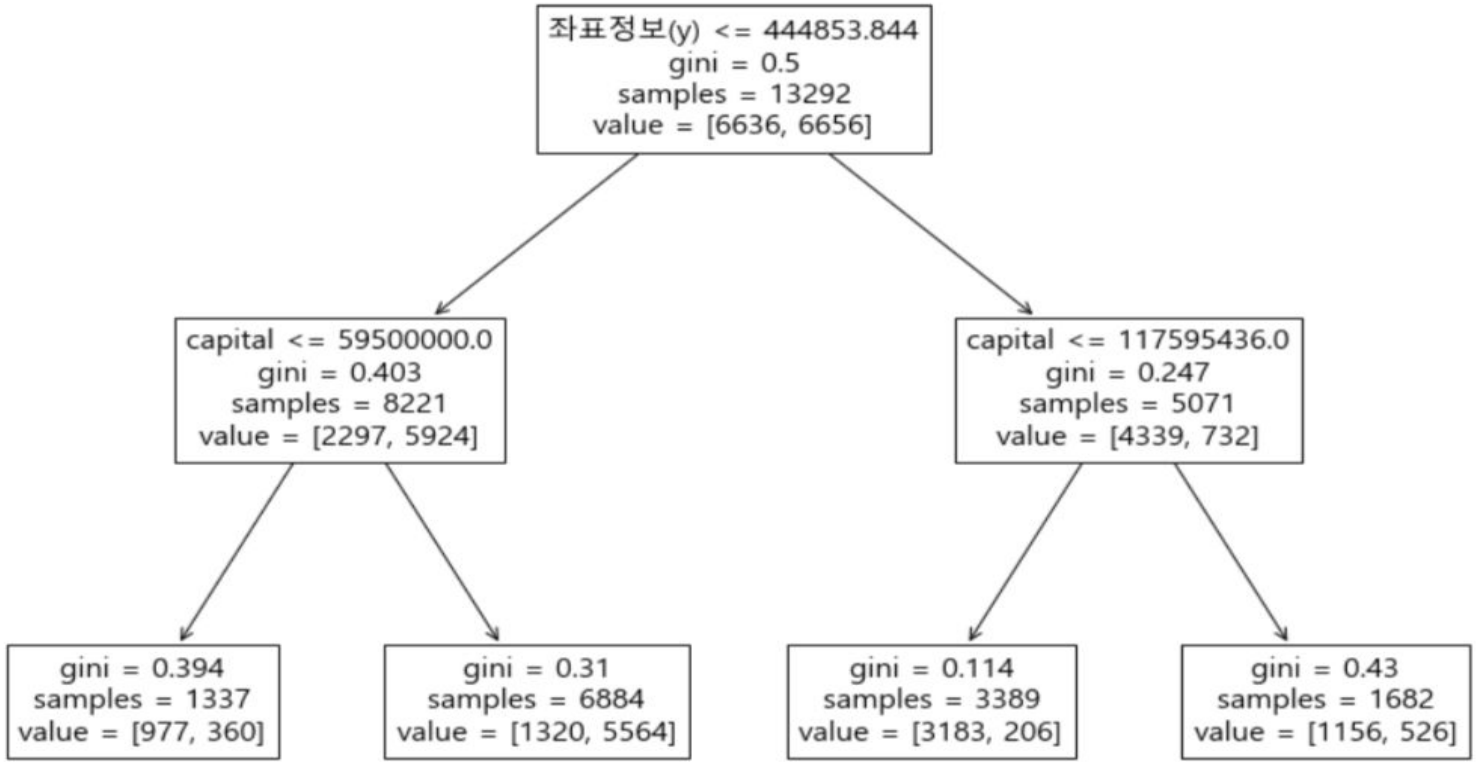
3. 기계학습 – 2)의사결정나무

1.GridSearchCV 를 사용해서 최적의 파라미터 찾기

– 최적의 파라미터 (gcv.best_params_)
criterion : '**gini**',
max_depth : **3**,
max_leaf_nodes : **4**,
min_samples_split : **2**

3. 기계학습 – 2)의사결정나무

2. 최적의 파라미터로 학습한 모델 정확도
   - 학습용: 0.8185374661450496
   - 검증용: 0.8249097472924187



3. 기계학습 – 3)랜덤포레스트

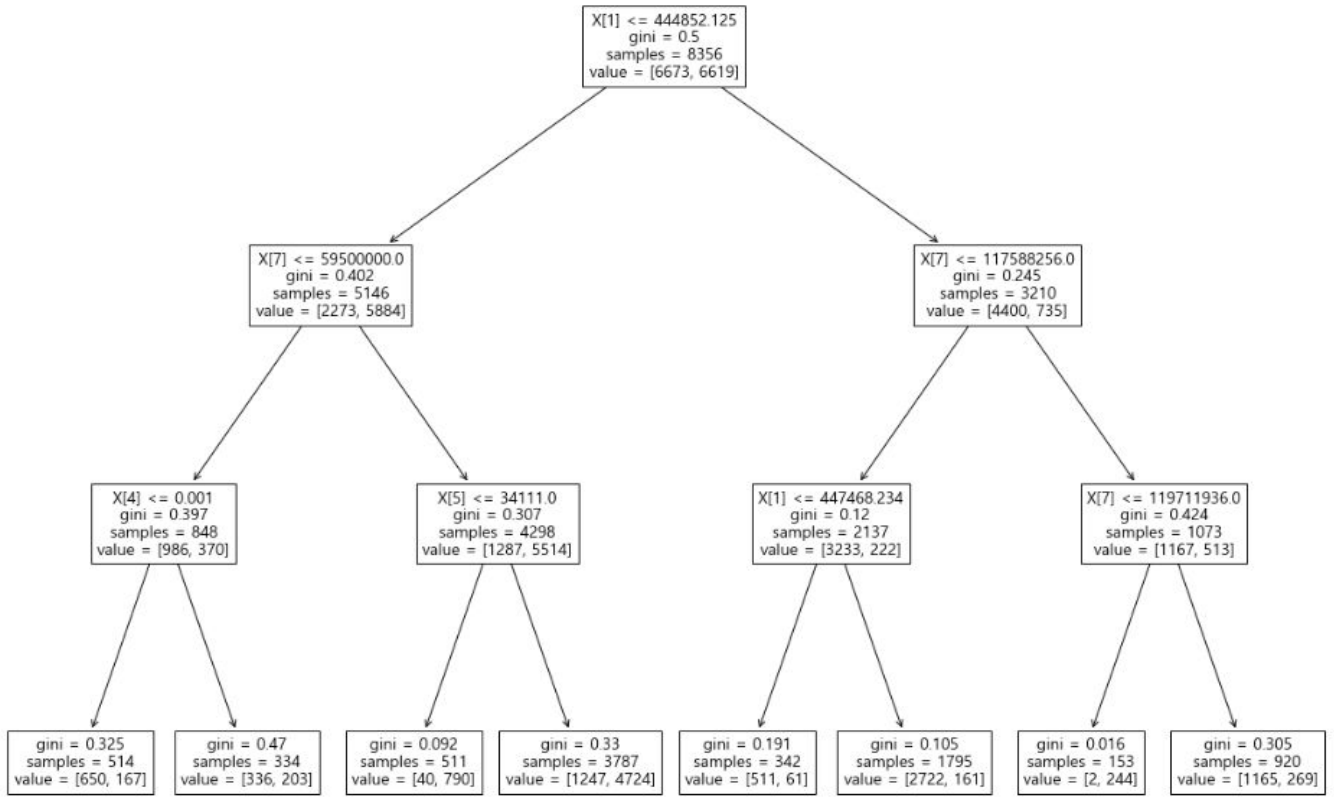
1.GridSearchCV 를 사용해서 최적의 파라미터 찾기

– 최적의 파라미터 (gcv.best_params_)
criterion : '**gini**',
max_depth : **3**,
n_estimators : 50

3. 기계학습 – 3)랜덤포레스트

2. 최적의 파라미터로 학습한 모델 정확도

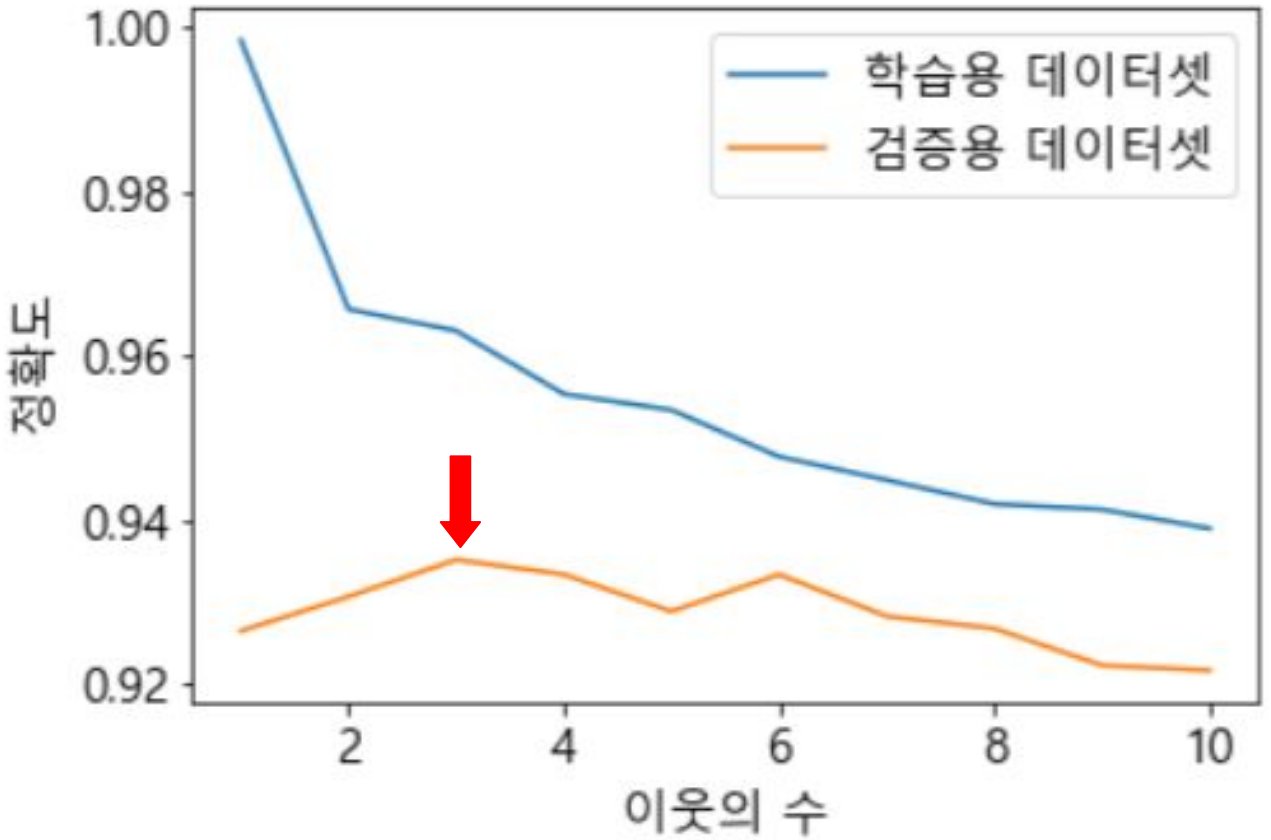
- 학습용: 0.7923563045440867
- 검증용: 0.796028880866426



3. 기계학습 – 4)KNN

1. 최적의 이웃 수 구하기

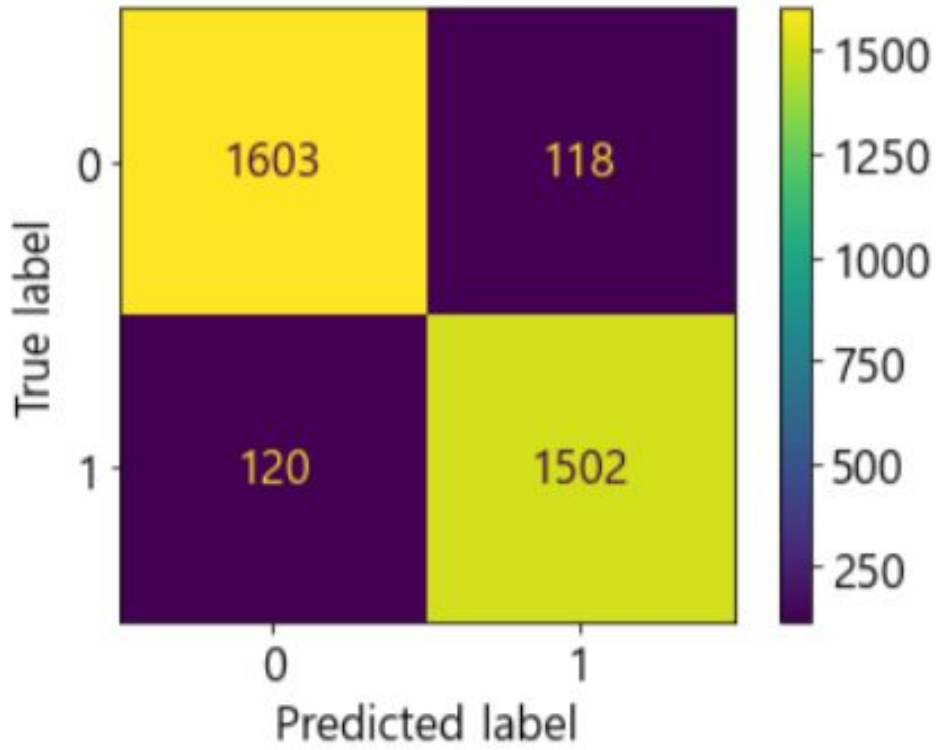
- 최적의 이웃 수 : **3**



3. 기계학습 – 4)KNN

2. 최적의 이웃 수 = **3** 으로 학습한 모델 정확도
   - 학습용 : 0.9627468581687613
   - 검증용 : 0.9347891115764284

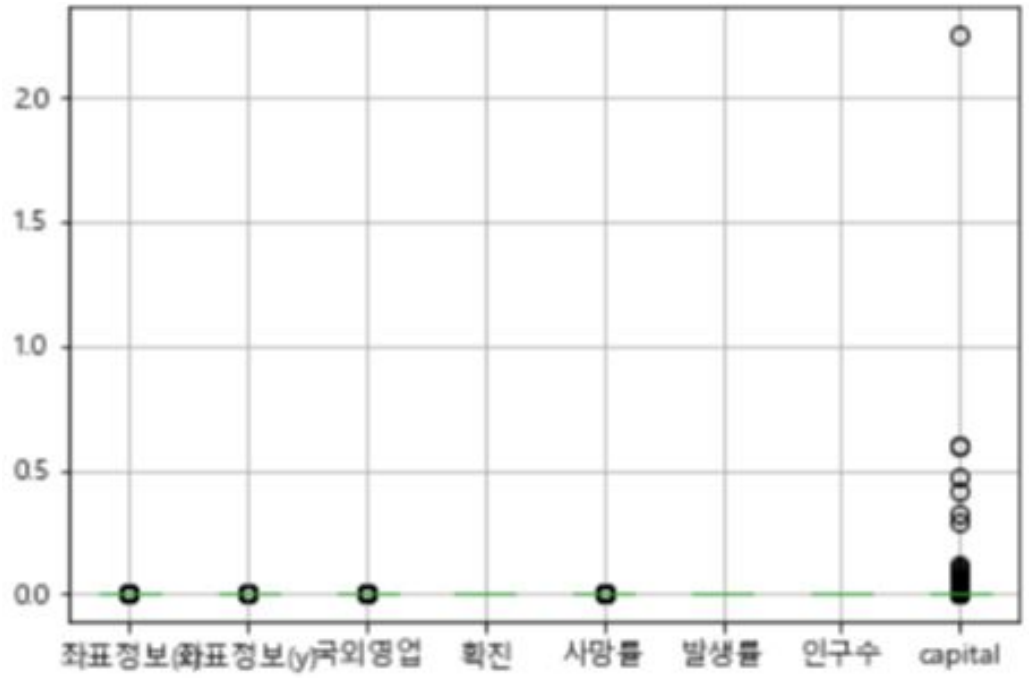
3.confusion matrix 출력



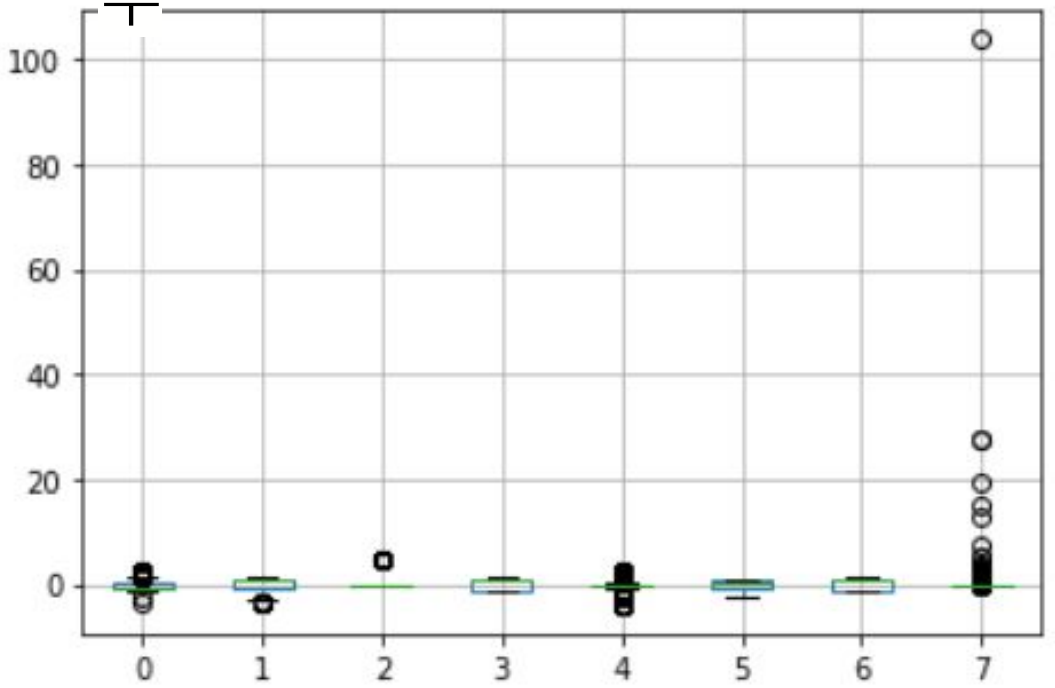
3. 기계학습 – 5)SVM

1. 스케일링

1) 스케일링 전



2) StandardScaler 적용 후



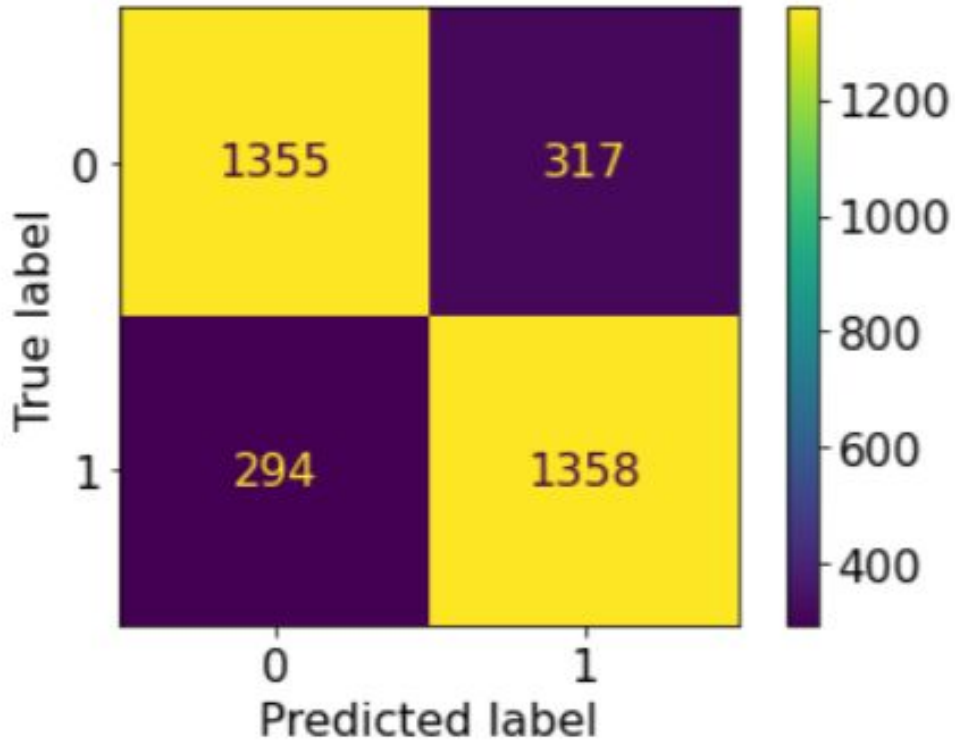
3. 기계학습 – 5)SVM

2. 스케일링한 데이터로 학습한 모델 정확도
   - 학습용 : 0.7536864279265724
   - 검증용 : 0.7409747292418772

3. 기계학습 – 6)ANN

1. 스케일링
2. 스케일링한 데이터로 학습한 모델 정확도
   - 학습용 : 0.8192897983749624
   - 검증용: 0.8161853188929001

3.confusion matrix 출력



3. 기계학습 – 7)keras 신경망

1. StandardScaler 적용

2. 모델

– 손실함수 'mse', 'binary_crossentropy' 의 결과값은 동일

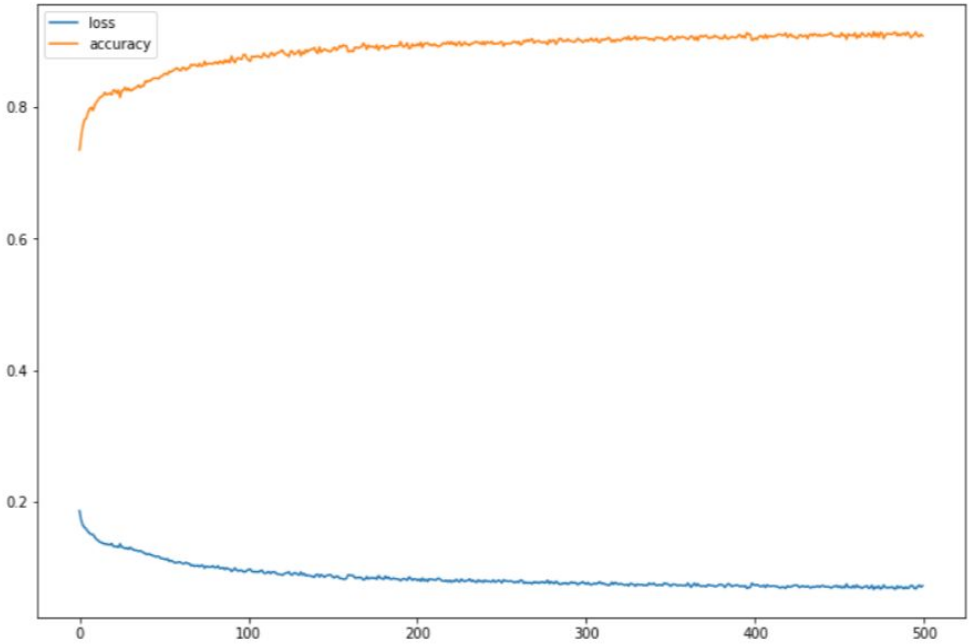
```
model = Sequential()
model.add(Dense(128, input_shape=(len(X_train.columns),),activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss='mse', optimizer='adam', metrics=['accuracy'])
#model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

3. 기계학습 – 7)keras 신경망

3. 모델 적용

hist = model.fit(X_train_scaled, y_train, epochs=500

```
Epoch 497/500
416/416 [==============================] - 1s 2ms/step - loss: 0.0713 - accuracy: 0.9089
Epoch 498/500
416/416 [==============================] - 1s 2ms/step - loss: 0.0734 - accuracy: 0.9072
Epoch 499/500
416/416 [==============================] - 1s 2ms/step - loss: 0.0705 - accuracy: 0.9108
Epoch 500/500
416/416 [==============================] - 1s 2ms/step - loss: 0.0723 - accuracy: 0.9084
```



4. 모델 정확도

loss, accuracy = model.evaluate(X_test**_scaled**, y_test, verbose=0)

- 정확도 = 0.88

4. 기계학습 분석 – 결론

	로지스틱 회귀분석	의사결정 나무	랜덤 포레스트	KNN	SVM	ANN	keras
학습용 정확도	69.14%	81.85%	79.23%	96.27%	75.36%	81.92%	
검증용 정확도	69.85%	82.49%	79.60%	**93.47%**	74.09%	81.61%	88%

<모델 정확도 순서>

KNN > keras > 의사결정나무 > ANN > 랜덤포레스트 > SVM > 로지스틱회귀분석

Ⅳ. 향후 계획

향후 계획

◈ 향후 계획

– 비정형 데이터 분석 추가

: 여행상품 리뷰 텍스트 데이터를 수집하여 텍스트마이닝으로 이진분류

감사합니다 ☺