빅데이터 기반 서비스 구현 프로젝트

1팀 김종욱, 박수인, 전병주, 김성혜, 전병조

목차

개요

미세먼지량 예측

장르별 도서 분류

소프트웨어 직군 임금 예측

리뷰로 알아보는 자연어 처리

GDP 예측

팀 구성원 역할

이름	역할
김종욱(팀장)	문제 정의 및 목표 설정, 데이터 수집 기반작업,데이터 수집
박수인	데이터 수집,데이터 전처리,데이터 모델링,결과 보고서 작성(ppt)
전병주	데이터 전처리,데이터 모델링,모델 평가
김성혜	데이터 전처리,모델 튜닝,모델 해석/분석
전병조	모델 평가,데이터 시각화,데이터 통합,데이터 수집

프로젝트 일정

(230320-230329)

230320 → 230321-230324 → 230327-230329

주제선정
필요 모델 파악
데이터 탐색

데이터 탐색
데이터 크롤링
데이터 전처리

모델 학습
예측 및 분류 시각화
PPT 제작

→ 230320 발표

프로젝트 수행 결과

@머신러닝-랜덤 포레스트

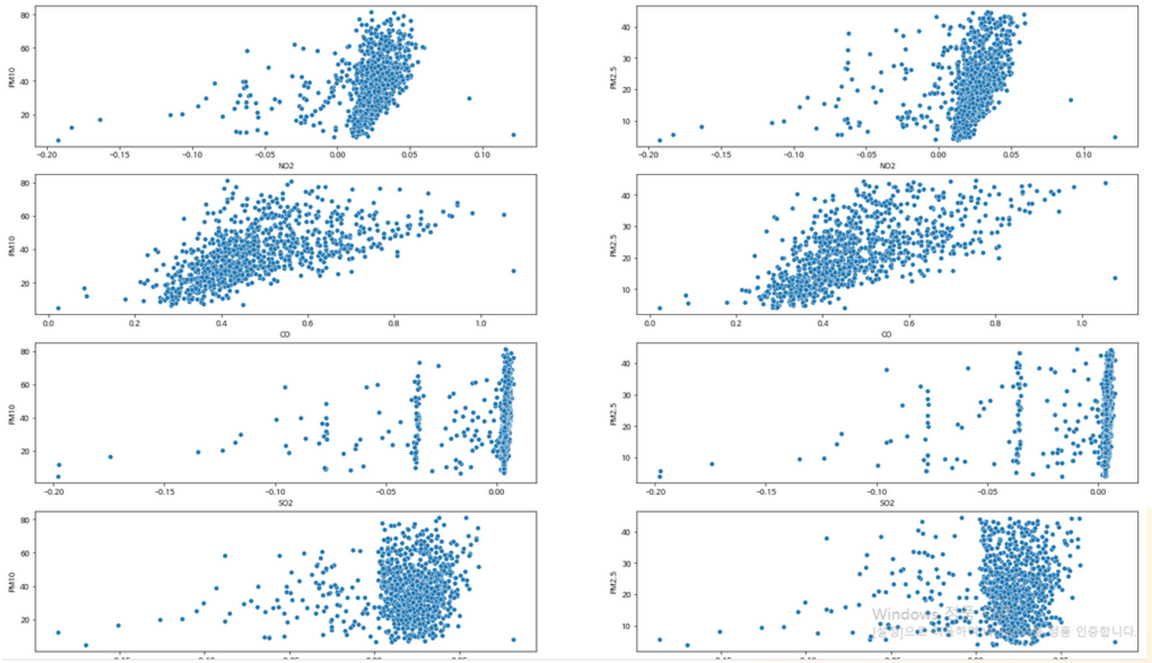
〈미세먼지 데이터 예측〉

-3년치 서울시의 오존(O3), 이산화질소(NO2), 일산화탄소(CO) 아황산가스(SO2)를 이용한 미세먼지(PM10)와 초미세먼지(PM2.5) 회귀

-데이터: 2017~2019년 서울시 공기 오염

-데이터 출처: kaggle, 타입: csv

-PM10과 PM2.5에 대한 오존(O3), 이산화질소(NO2), 일산화탄소(CO), 아황산가스(SO2)의 상관관계 시각화



-데이터의 상관계수

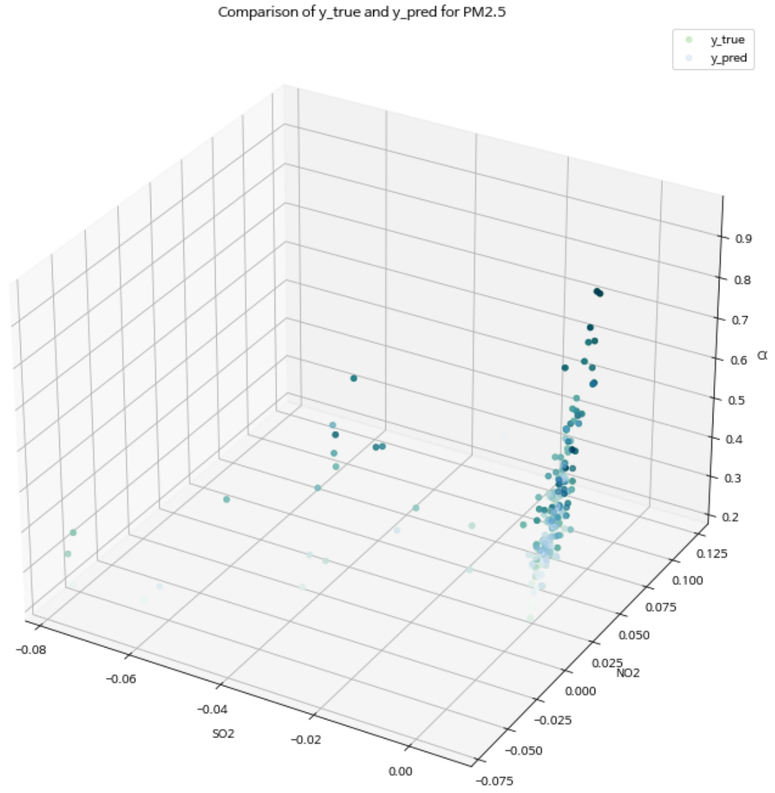
-pm10, pm2.5의 높은 연관성

-O3은 pm10, pm2.5에는 큰 영향이 보이지 않지만 SO2와 NO2에 영향을 미침

-직접적인 영향이 미비한 O3를 제외하고 PM10과 PM 2.5에 대한 테스트 값과 예측 값을 시각화하여 확인

	SO2	NO2	O3	CO	PM10	PM2.5
SO2	1.000000	0.880500	0.855932	0.263964	0.161528	0.125670
NO2	0.880500	1.000000	0.698836	0.558084	0.356869	0.340894
O3	0.855932	0.698836	1.000000	-0.011400	0.153230	0.105833
CO	0.263964	0.558084	-0.011400	1.000000	0.601016	0.646071
PM10	0.161528	0.356869	0.153230	0.601016	1.000000	0.808631
PM2.5	0.125670	0.340894	0.105833	0.646071	0.808631	1.000000

-O3(오존)을 제외한 PM2.5(초미세먼지) 3차원 그래프



@머신러닝-분류 예측

〈줄거리로 책의 장르를 구별할 수 있을까?〉

-약 5,500개의 영문 책의 줄거리

-전문가들이 어느 정도 가공한 자료

-자연어 처리와 라벨화

-단어 빈도에 따른 가중치 부여

-데이터 출처: kaggle, 타입: csv

title	genre	summary
Drowned Wednesday	fantasy	Drowned Wednesday is the first Trustee among ...
The Lost Hero	fantasy	As the book opens, Jason awakens on a school ...
e Eyes of the Overworld	fantasy	Cugel is easily persuaded by the merchant Fia...
Magic's Promise	fantasy	The book opens with Herald-Mage Vanyel return...
Taran Wanderer	fantasy	Taran and Gurgi have returned to Caer Dallben...

-줄거리 자연어 처리

```
lemma = WordNetLemmatizer()
stemmer = PorterStemmer()

def preprocess(text):
    words = word_tokenize(text)
    words = [lemma.lemmatize(word) for word in words if word not in set(stopwords.words('englis
    words = [stemmer.stem(word) for word in words]
    return " ".join(words)
```

-단어 빈도별 가중치

```
tfidf_vectorizer = TfidfVectorizer(max_df = 0.8, max_features= 10000)
xtrain_tfidf = tfidf_vectorizer.fit_transform(X_train.values.astype('U'))
xtest_tfidf = tfidf_vectorizer.transform(X_test.values.astype('U'))

svc = SVC()
svc.fit(xtrain_tfidf, y_train)
```

-영문 도서 csv 파일 오픈

-행: 예측 장르, 열: 실제 장르

	fantasy	science	crime	history	horror	thriller	psychology	romance	sports	travel
fantasy	212	15	0	8	6	28	0	4	0	1
science	11	117	0	11	2	32	2	0	0	0
crime	3	0	72	4	2	85	0	0	0	0
history	11	7	2	125	0	27	1	0	1	1
horror	17	4	1	5	75	76	0	0	0	0
thriller	7	10	3	5	9	276	0	1	0	0
psychology	2	25	0	0	0	10	0	0	0	0
romance	11	0	0	0	0	17	0	0	2	0
sports	2	0	0	1	0	13	0	0	8	0
travel	6	4	0	10	0	8	0	0	0	0

-줄거리에서 단어 빈도수에 따라 단어 벡터화, 가중치 부여

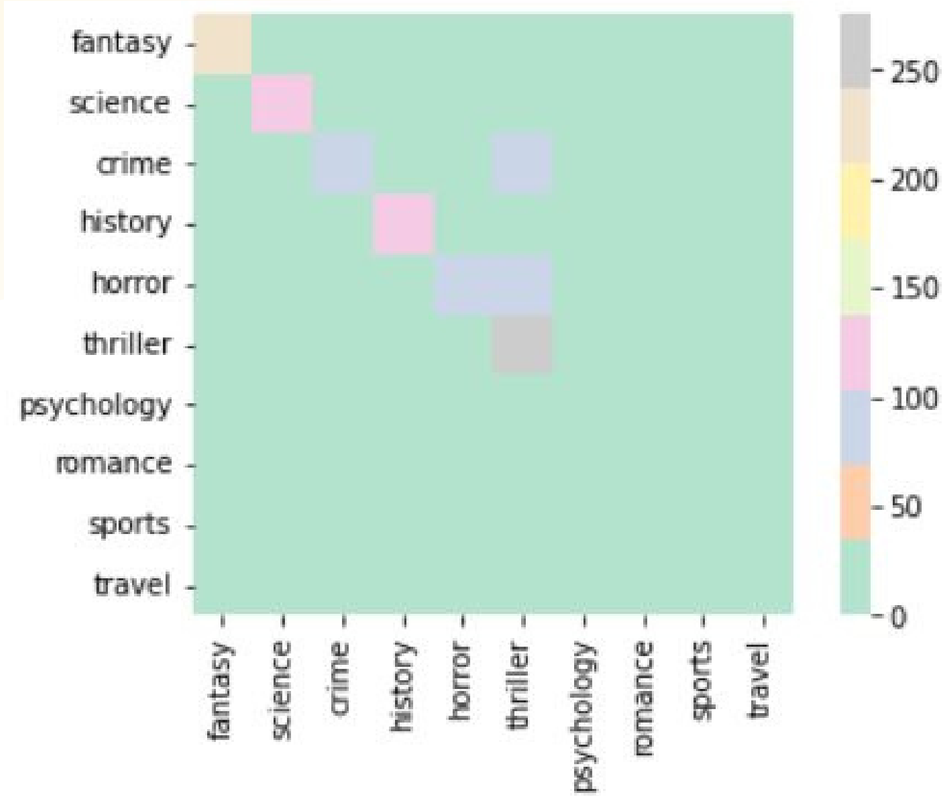
-데이터를 학습용과 테스트용으로 분리, 분류 모델에 학습

-데이터의 실제 장르와 예측장르를 표와 혼동행렬로 출력

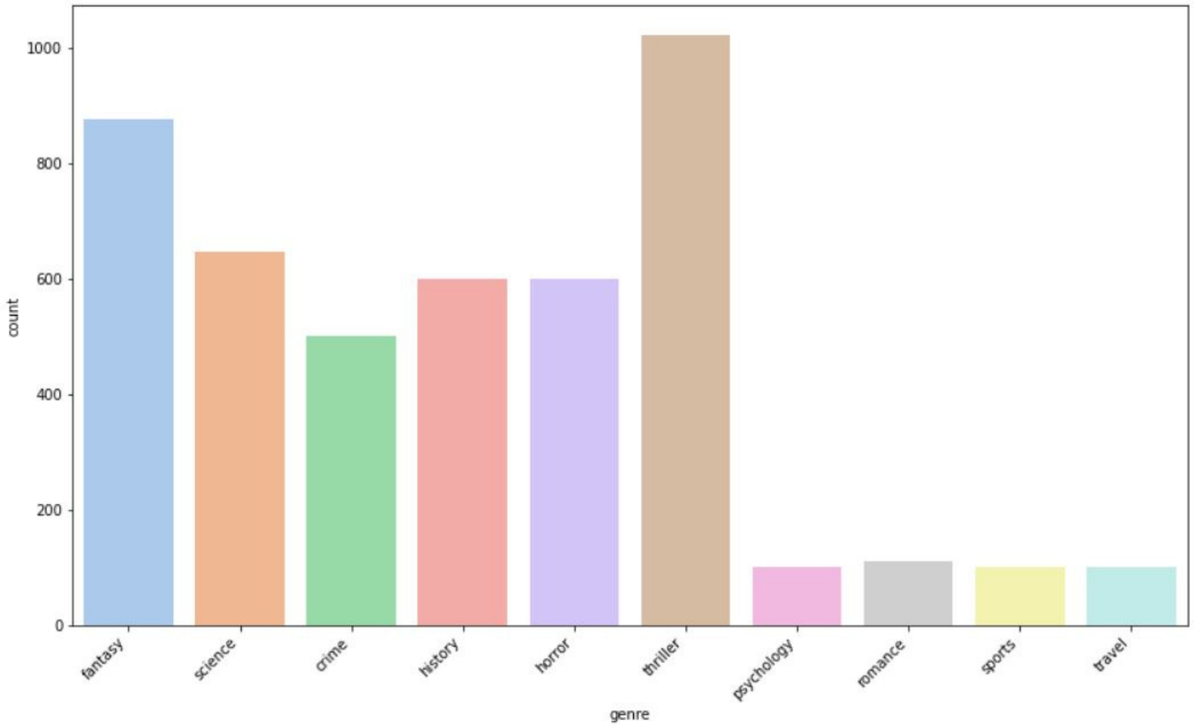
-혼동행렬

	TN	FP	FN	TP
fantasy	1054	70	62	212
science	1158	65	58	117
crime	1226	6	94	72
history	1179	44	50	125
horror	1201	19	103	75
thriller	791	296	35	276
psychology	1358	3	37	0
romance	1363	5	30	0
sports	1371	3	16	8
travel	1368	2	28	0

-모델 테스트 결과 시각화



-사용된 데이터의 장르(y)의 분포



-학습된 모델에 테스트 데이터를 넣은 결과 시각화

-스릴러 장르 이후 철학, 로맨스, 운동, 여행 장르들은 제대로 분류가 되지 않은것 처럼 보임

-몇몇 장르의 적은 데이터 수가 원인으로 추정됨

@머신러닝-선형회귀

〈SW기술자의 하루 임금은 얼마나 될까?〉

-선형회귀: 임금예측과 전망

-전문가들이 어느 정도 가공한 자료

-데이터: 3년치 IT종사자 하루(DAY)임금

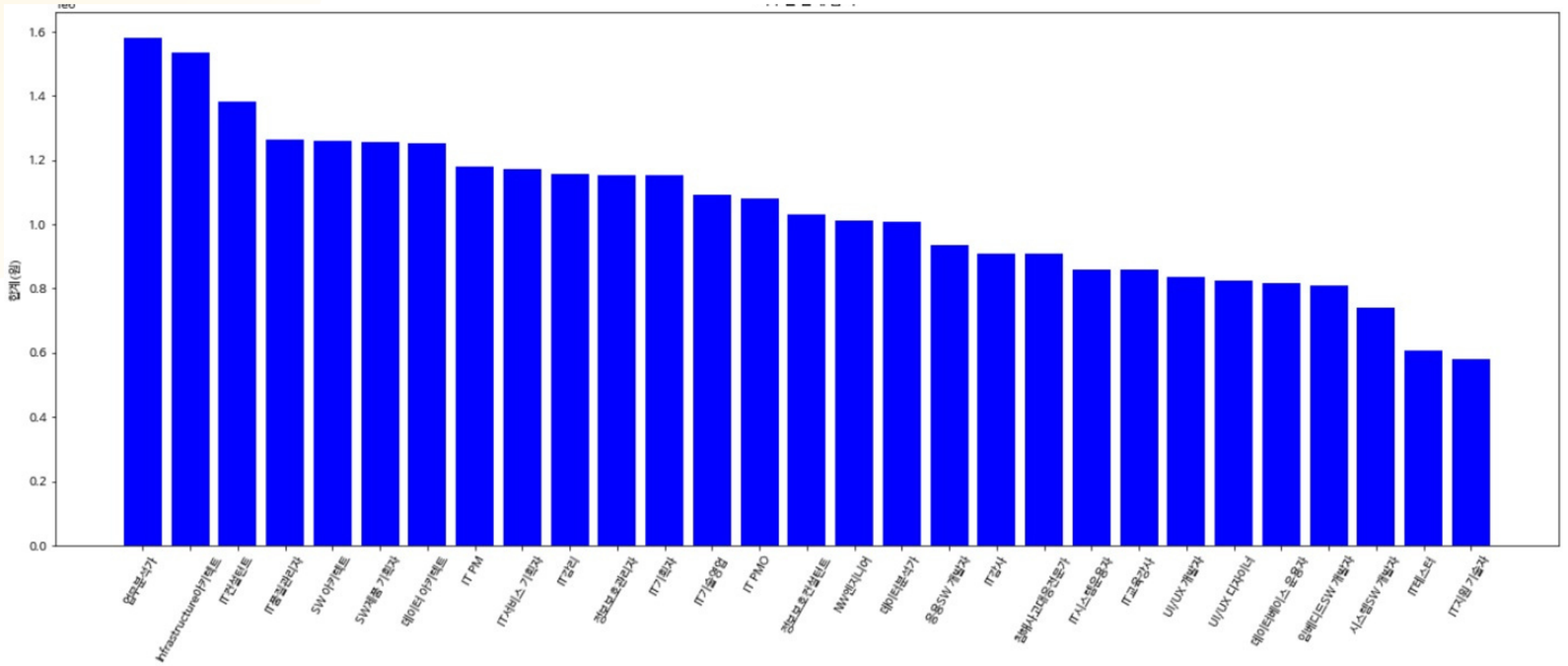
-데이터 출처: kosa, 타입:csv

-참고 자료: 통계 정보 보고서_소프트웨어기술자임금실태 조사 2020

-SW직군 CVS파일 오픈

	직무별	2019	2020	2021
0	IT기획자	403081	388724	360307
1	IT컨설턴트	437900	458818	484732
2	정보보호컨설턴트	340978	342406	347123
3	업무분석가	501090	532243	548550
4	데이터분석가	335799	347670	323184
5	IT PM	362780	411329	406823
6	IT PMO	410270	326211	345428
7	SW 아키텍트	389104	421761	448240
8	Infrastructure아키텍트	461684	517539	556512
9	데이터 아키텍트	399985	437063	414770
10	UI/UX 개발자	258696	302033	274465
11	UI/UX 디자이너	-	250345	228717
12	응용SW 개발자	305985	323174	306034

-SW직군 임금 막대그래프/1day



-상위 3개 직업군: 업무분석가, Infrastructure아키텍트, IT컨설턴트

-하위 3개 직업군은 시스템SW개발자, IT테스터, IT지원 기술자

-소프트웨어 개발자: 중위권에서 하위권 사이 위치

전반적으로 임금은 상승중이고 금액의 분포가 넓어짐

-SW직군 테스트모델 점수 (Score,R-squared)

```
# 회귀분석
# 2019, 2020, 2021을 입력으로
model = LinearRegression()
model.fit(X_train, y_train)
print('테스트 모델 Score:',mo
```

```
테스트 모델 Score: 0.9999999999822013
```

```
# 검증용 데이터를 사용하여 모
y_pred = model.predict(X_test
r2score = r2_score(y_test, y_
print('R-squared:', r2score)
#R-squared(결정계수) 값을 계
#R-squared 값은 0에서 1 사이
```

```
R-squared: 0.9999999999822013
```

```
print('기울기:',model.coef_)
print('y절편:',model.intercep
```

```
기울기: [0.33333256 0.333333   0.33333432]
y절편: -0.3242583299870603
```

@머신러닝-비지도 군집

〈미장원이 생기고 사라지고 또다시 어디에 나타날까?〉

-K-means 군집으로 분포 확인

-데이터: 대한민국 전국 38만개의
미용업 풀 데이터데이터

-데이터 출처: Data.go.kr, 타입: csv

-미장원 CVS파일 오픈

	인허가일자	폐업일자	상세영업상태코드	업태구분명	사용시작지상층	의자수
0	2022-05-17	2022-08-03	2	네일아트업	0.0	3.0
1	2022-05-19	2022-07-13	2	일반미용업	1.0	4.0
2	2022-05-19	2023-02-09	2	일반미용업	0.0	2.0
3	2022-05-23	2022-07-27	2	네일아트업	1.0	1.0
4	2022-05-23	2022-08-04	2	네일아트업	0.0	2.0
...	...	...	...	...	...	...
389664	1996-02-28	NaN	1	일반미용업	1.0	3.0
389665	2010-11-22	NaN	1	일반미용업	0.0	4.0
389666	2018-07-16	NaN	1	피부미용업	NaN	0.0
389667	2018-07-16	NaN	1	네일아트업	NaN	2.0
389668	2018-07-16	NaN	1	네일아트업	0.0	4.0

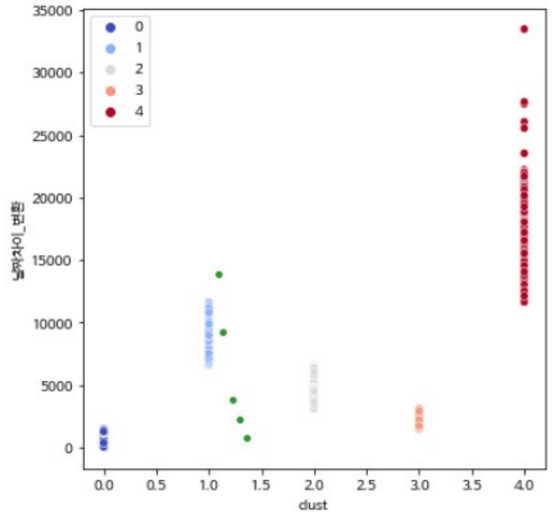
361433 rows × 6 columns

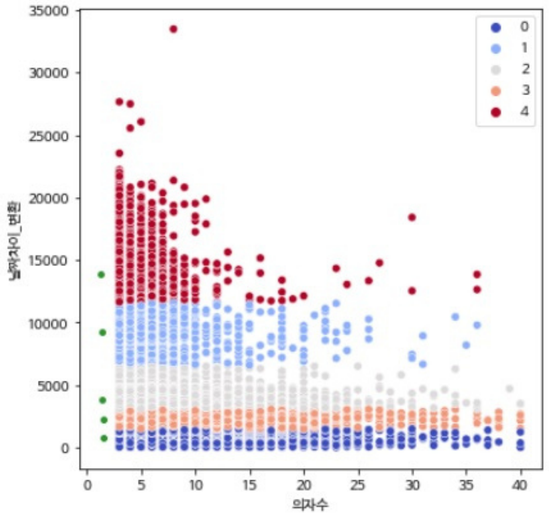
“

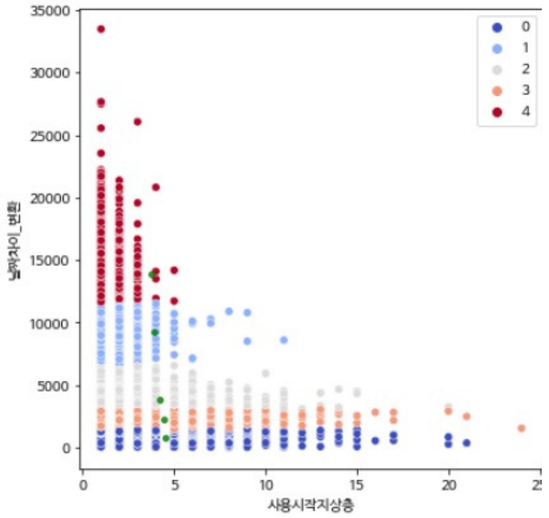
-시간/클러스터,의자 수,사용자 시작 층　-3D)시간/클러스터,의자 수,사용자 시작 층

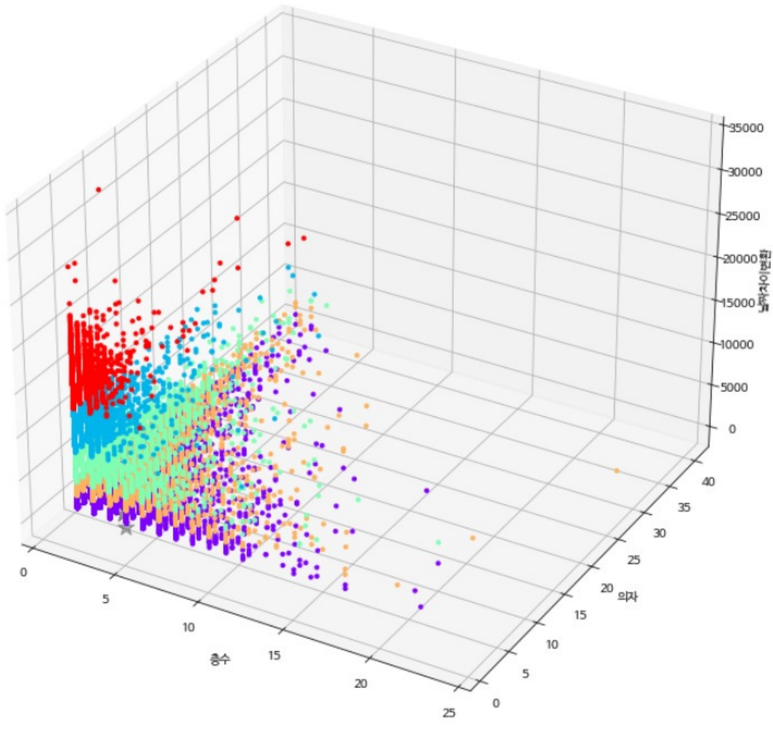
-시간을 날일(day)로 변환하고 이를 수치화

-영업유무, 의자수, 사용자시작층, 업태를 분석요소로 선정하여 K-means 시행









@머신러닝-한글 자연어 처리

〈2022년 12월 25일 네이버 뉴스 제목 시각화〉


```
1 date='20221225'
2
3 news_url = 'https://news.naver.com/main/ranking/popularDay.nhn?date={}'.format(date)
4
5
6 headers = {'User-Agent' : 'Mozilla/5.0'}
7 req = requests.get(news_url, headers = headers)
8
9 soup = BeautifulSoup(req.text, 'html.parser')
0 news_titles = soup.select('.rankingnews_box > ul > li > div > a')
1
2 crowled_title = []
3 for i in range(len(news_titles)):
4     crowled_title.append(news_titles[i].text)
5     print(i+1, news_titles[i].text)
```

@딥러닝-리뷰 감성 분류

〈GRU로 네이버 쇼핑 리뷰 감성 분류〉

-GRU로 네이버 쇼핑 리뷰 감성 분류

```
from tensorflow.keras.layers import Embedding, Dense, GRU
from tensorflow.keras.models import Sequential
from tensorflow.keras.models import load_model
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

embedding_dim = 100
hidden_units = 128

model = Sequential()
model.add(Embedding(vocab_size, embedding_dim))
model.add(GRU(hidden_units))
model.add(Dense(1, activation='sigmoid'))

es = EarlyStopping(monitor='val_loss', mode='min', verbose=1, patience=4)
mc = ModelCheckpoint('best_model.h5', monitor='val_acc', mode='max', verbose=1,
                     save_best_only=True)

model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=['acc'])
history = model.fit(X_train, y_train, epochs=15, callbacks=[es, mc],
                    batch_size=64, validation_split=0.2)
```

-리뷰 예측

```
sentiment_predict('배송 빠르고 제품도 좋아요. 대박나세요~')
```

```
1/1 [==============================] - 1s 821ms/step
97.14% 확률로 긍정 리뷰입니다.
```

```
sentiment_predict('너무 불친절하네요, AS도 너무 오래 걸림. 비추비추.')
```

```
1/1 [==============================] - 0s 33ms/step
99.79% 확률로 부정 리뷰입니다.
```

```
sentiment_predict('강추합니다. 재주문 의사 강력함. ^-^')
```

```
1/1 [==============================] - 0s 39ms/step
97.01% 확률로 긍정 리뷰입니다.
```

```
sentiment_predict('다른 곳에서 사세요. 주문하다가 기분 나빠졌음')
```

```
1/1 [==============================] - 0s 42ms/step
93.22% 확률로 부정 리뷰입니다.
```

```
sentiment_predict('저렴한 가격에 정말 친절하심. 굿굿굿')
```

```
1/1 [==============================] - 0s 47ms/step
92.87% 확률로 긍정 리뷰입니다.
```

```
sentiment_predict('별로예요. ㅏㄴㅇㅁㅗㄱㅈ더ㅏㅠㄴㅁ룰냥ㄹ니아ㅓㅁ륜ㅇㅎㄱ바ㅣㅠ마늘,ㅣ')
```

```
1/1 [==============================] - 0s 31ms/step
99.29% 확률로 부정 리뷰입니다.
```

@딥러닝-시계열 데이터

〈lstm을 통한 gdp 예측〉

-데이터: 국내총생산(당해년가격), 1960-2021.xlsx (총 208 나라, 1990-2021 데이터 사용)

-데이터 출처:KOSIS 국가통계포털

오만	11.7	11.3	12.5
파키스탄	40.0	45.6	48.9
필리핀	50.5	51.8	60.4
카타르	7.4	6.9	7.6
사우디아라비아	117.6	132.2	137.1
싱가포르	36.1	45.5	52.1
스리랑카	8.0	9.0	9.7
시리아	23.9	27.8	33.1
대만	166.4	187.1	222.9

-relu는 일반적으로 lstm에 비적합 sigmoid, tanh 주사용

-relu 함수는 입력값이 0보다 작을 때는 0을 출력하고, 0보다 큰 경우에는 입력값을 그대로 출력

-sigmoid, tanh: 입력값의 범위를 [0, 1] 또는 [-1, 1]로 제한하여, 입력값이 0보다 작아져도 일정한 정보를 전달할 수 있음

또한, tanh 함수와 sigmoid 함수는 미분 가능하고, 비선형성을 가지기 때문에 RNN 모델에서 잘 동작

-lstm 코드 체크v

```python
# X, y 분리
def split_xy(dataset, time_steps, y_column):
    X, y = [], []
    for i in range(len(dataset)-time_steps+1):
        x = dataset[i:(i+time_steps), :]
        X.append(x)
        y.append(dataset[i+time_steps-1, y_column])
    return np.array(X), np.array(y)

time_steps = 3
y_column = 31
X_train, y_train = split_xy (train_scaled, time_steps, y_column)
X_test, y_test = split_xy (test_scaled, time_steps, y_column)
                    # (batch_size, timesteps, input_dim)인 3D 텐서
print(X_train.shape) # (148, 3, 32)
print(y_train.shape) # (148,)
```

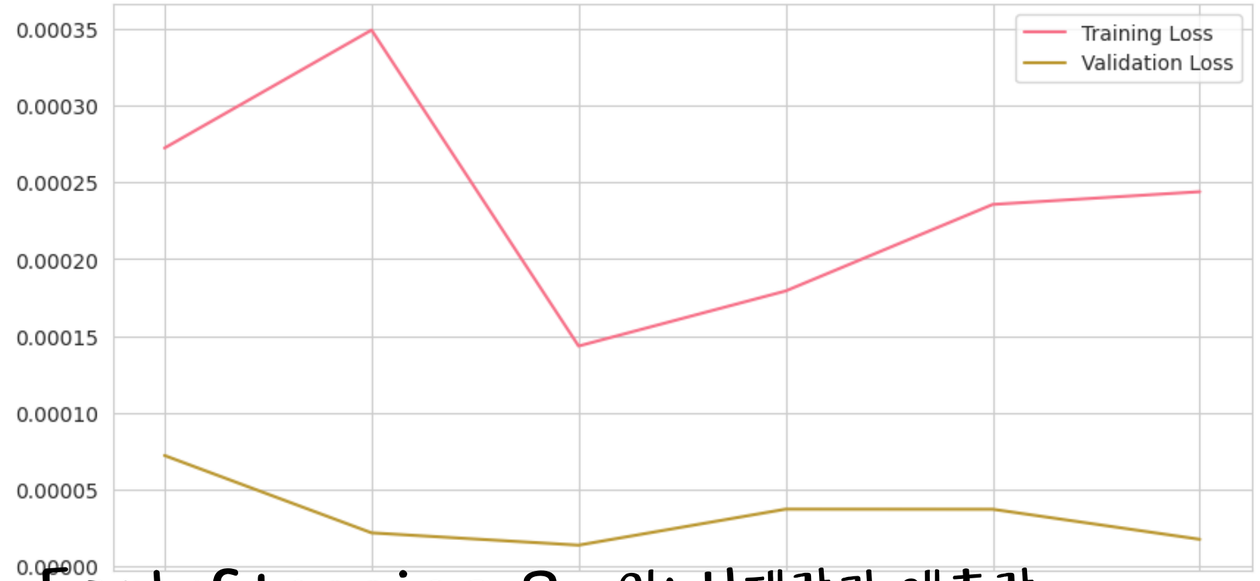
```
48, 3, 32)
48,)
```

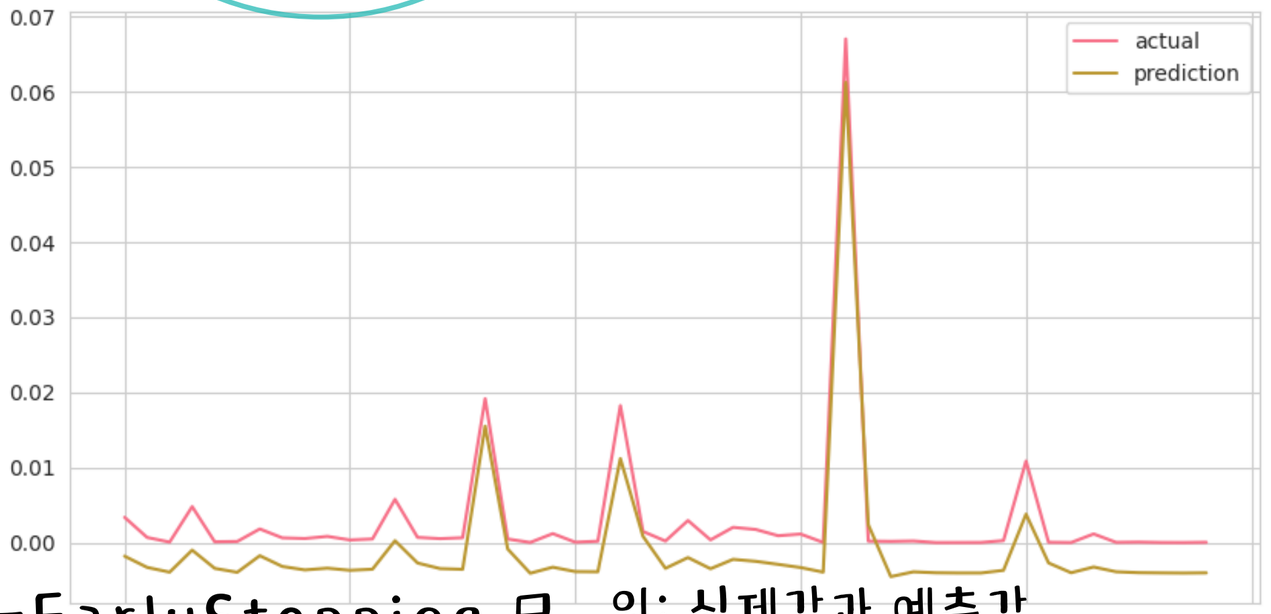
```python
# LSTM 모델 정의
model = Sequential()
model.add(LSTM(64, return_sequences=True, input_shape=(3, 32)))
model.add(Dropout(0.2))
model.add(LSTM(32, return_sequences=True))
model.add(Dropout(0.2))
model.add(LSTM(16))
model.add(Dropout(0.2))
model.add(Dense(1, activation='tanh'))
model.compile(loss='mse', optimizer='adam')
```

```python
# *모델 학습 2
history = model.fit(X_train, y_train, epochs=100, batch_size=1, validation_split=0.2,
                    callbacks=[EarlyStopping(patience=3, monitor='val_loss')])
```

1. 시계열 데이터 전처리
    a. time_steps
        i. 시계열 데이터를 처리할 때 한 번에 처리할 데이터의 개수
        ii. 몇 개의 과거 시점의 데이터를 이용하여 예측할 것인지를 나타내는 변수
        iii. 각각의 데이터 포인트는 time_steps 길이의 X 데이터와 1일치의 y 데이터로 구성
2. lstm 입력층
    a. <데이터 입력 형태 (timesteps, input_dim)>
        i. input_dim: 입력 데이터의 특성의 개수를 나타내는 매개변수
3. EarlyStopping
    a. 딥러닝 모델 학습 시, validation loss가 더 이상 개선되지 않을 때 학습을 조기 종료하는 기법
    b. 모델이 과적합(overfitting)되어 일반화 성능이 떨어지는 것을 방지

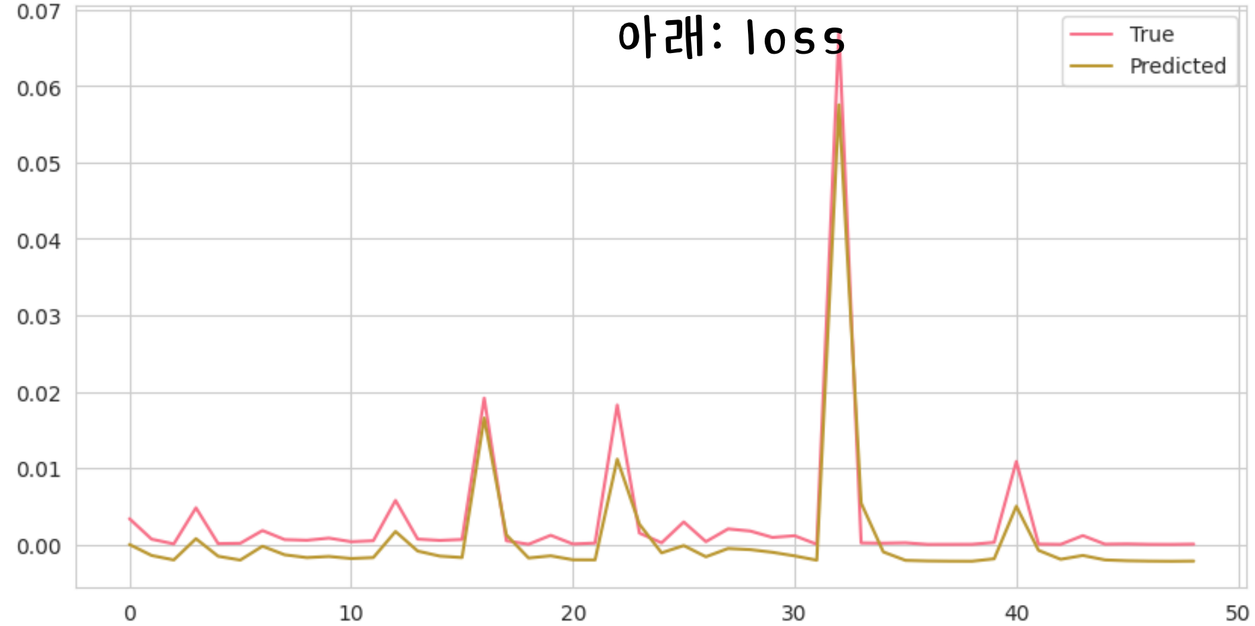
-EarlyStopping 유무





-EarlyStopping 유 위: 실제값과 예측값

아래: loss



-EarlyStopping 무 위: 실제값과 예측값

아래: loss

-EarlyStopping 유무

-EarlyStopping 유

```
Epoch 6/100
40/40 [==============================] - 0s 12ms/step - loss: 2.4390e-04 - val_loss: 1.7615e-05
17/17 [==============================] - 0s 4ms/step - loss: 8.8976e-06
Test set MSE: 0.000009
2/2 [==============================] - 0s 8ms/step
Test set MAE: 0.005987
2/2 [==============================] - 0s 6ms/step
Test set R-squared: 0.911858
2/2 [==============================] - 0s 6ms/step
```

-EarlyStopping 무

```
17/17 [==============================] - 0s 4ms/step - loss: 1.7989e-05
Test set MSE: 0.000018
2/2 [==============================] - 1s 7ms/step
Test set MAE: 0.007542
2/2 [==============================] - 0s 6ms/step
Test set R-squared: 0.821796
2/2 [==============================] - 0s 6ms/step
```

-똑같은 모델, mse, mae, r-squared 차이

-r-squared은 0과 1 사이, 1에 가까울수록 모델이 데이터를 더 잘 설명

-mse, mae 낮을수록 모델이 데이터를 더 잘 설명

팀 구성원 자체 평가

팀원 자체 평가

김종욱

이번에 벌써 5번째이자 마지막 프로젝트입니다.
5번째가되니 어떠한 언어로 접근한다는 생각보다는 컴퓨팅적인 사고로 구체적인 과정을 만들어 나가는 것이 무척이나 재미있었습니다.
모두 화이팅입니다.^^

전병주

프로젝트 초의 목표는 요리 레시피와 재료를 크롤링한 후 재료를 입력하면 보유 중인 재료로 요리 가능한 최적의 레시피를 찾는 머신러닝을 하고자 했습니다.
하지만 비정형 데이터를 정제하는 것이 생각보다 많이 힘들어서 일정상 서울 공기오염도 머신러닝을 하게 되었습니다.
머신러닝을 위한 데이터를 찾고 다듬는 것이 가장 어려웠고 데이터에 적절한 모델을 찾는 과정도 쉽지 않았습니다.

박수인

자연어 처리와 더불어 다양한 전처리 과정의 중요성을 다시한번 느낄 수 있었습니다.
분류 하나만 하여도 다양한 모델이 존재하고 다양한 모델에서의 작업후 모두 다른 정확도를 보임으로 어떤 데이터를 쓰고 어떤 모델을 사용하냐에 따라 전혀 다른 결과가 나옴을 몸소 체험할 수 있었습니다.

전병조

띄어쓰기로 구분할 수 있는 영어보다 형태소 구분이 복잡한 한글의 특수성으로 인해 딥러닝으로 자연어 처리하는 것이 쉽지만은 않았고, 코랩 런타임 유형에서 CPU에 비해 GPU의처리 속도가 훨씬 빠르다는 것을 경험하였습니다.

김성혜

직접 데이터를 모아보니 양이 많으면서 쓸만한 데이터를 찾는 것과, 외국 데이터도 있어서 해석에도 어려움을 겪었습니다. 또 딥러닝은 다른 언어나 분야에 비해 인터넷에 자료가 적다는 것을 느꼈습니다.
딥러닝에 대해서 예전보다 많이 알게 되어서 좋았지만 차원에 대해 공부해서 입력 변수와 출력 변수의 수를 늘려보고 싶습니다.

감사합니다