‘AI 이용해서 영작문 자신감 길러보자'

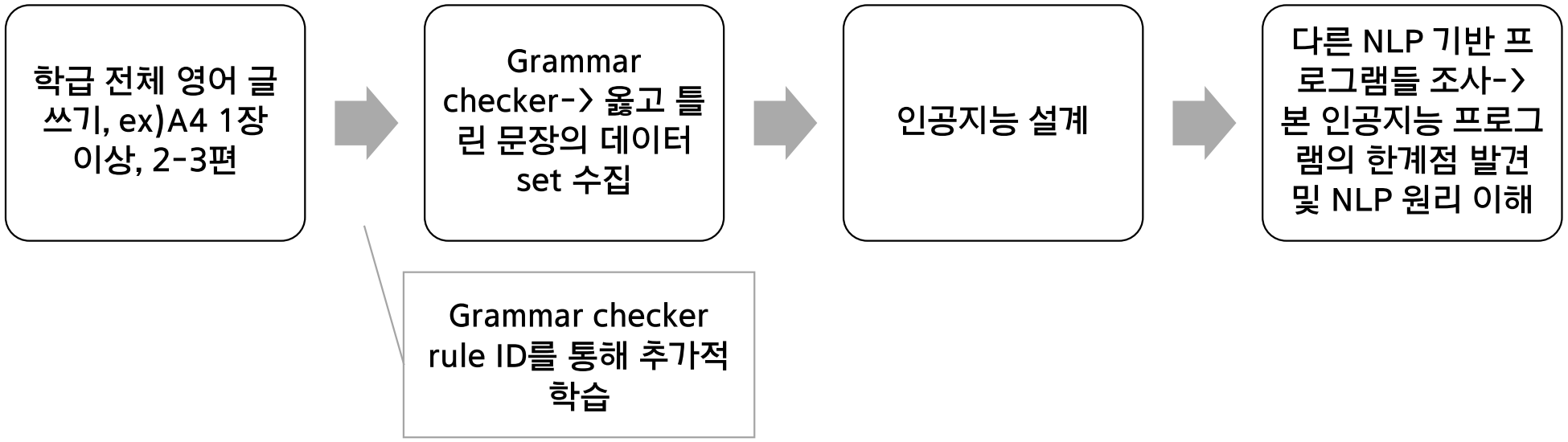
컴퓨터교육과 202100337 제갈민

프로젝트 설계의 계기

[이혜정의 교육과 세상] 말하기·쓰기 평가 못하는 수능 영어

➔ '못해도 해보면 어떨까'

수업 전반 흐름


학급 전체 영어 글 쓰기, ex)A4 1장 이상, 2-3편
Grammar checker-> 옳고 틀린 문장의 데이터 set 수집
Grammar checker rule ID를 통해 추가적 학습
인공지능 설계
다른 NLP 기반 프로그램들 조사-> 본 인공지능 프로그램의 한계점 발견 및 NLP 원리 이해

융합 단계	차시	학습과정	교수학습 시나리오	AI(Tech) 및 유의점	역량 (핵심) (교과)
감성적 체험	1	도입	학생들에게 자유 주제/ 수업 연계 주제를 주고 A4 2-3장 분량의 영어 작문 과제를 수행할 수 있도록 지도한다. 학생들끼리 자신이 작성한 바를 발표하며 자신의 생각을 나눌 수 있도록 지도한다.		영어
	2	도입	Machine Learning 과 NLP (Natural Language Processing)의 개념과 기본적인 원리를 학습해본다. 이때 실제 사례들은 조별 활동을 통해 찾도록 한다.	Machine learning, NLP 개념	정보, 수학
창의적 설계	3	준비	파이썬 라이브러리와 문법에 대해 학습하고 grammar checker를 만들 수 있는 라이브러리를 찾아본다.	파이썬 기본 문법	정보, 영어
	4	준비	다른 grammar checker를 찾아보고 파이썬으로 만드는 grammar checker를 설계해본다.		정보, 영어
	5	시행	grammar checker를 위한 코드를 작성해보고 학생이 작문한 글을 grammar checker를 이용해서 검사해본다.		정보, 영어
	6	준비	grammar checker를 통해 수집한 정/오 문장을 기반으로 data set을 마련한다.	Data Set의 개념	정보, 영어
	7	시행	rule ID을 기반으로 자신이 한 실수를 확인해보고 추가 오류 문장 또한 data set으로 마련한다.		정보, 영어
	8	준비	이전에 찾았던 grammar checker에서 모티브를 얻어 조별로 머신러닝을 기반으로 한 grammar checker를 설계해본다.		정보, 영어
	9	시행	머신러닝 grammar checker 코드를 짜는데 필요한 라이브러리, 인공지능 모델에 대한 이해를 하고 조별로 교사의 도움을 통해 직접 코드를 짜본다.	1. Natural Language Toolkit (NLTK) 2. TextBlob 3. CoreNLP 4. Gensim 5. spaCy 6. polyglot 7. scikit-learn 8. Pattern	정보, 수학
감성적 체험	10	시행	조별로 인공지능 모델의 정확성을 확인하고 현 인공지능 모델의 한계는 무엇인지 향후 어떻게 발전시키면 좋을지 고민해보고 직접 생각을 나눠본다.		정보, 수학
진로 연계 활동	11	시행	자신의 진로에 NLP 기반 머신러닝을 융합시켰을 때 어떤 이점이 있을지 예상되는 모습은 무엇인지 그림으로 표현해본다.	머신러닝 활용	정보, 진로, 미술

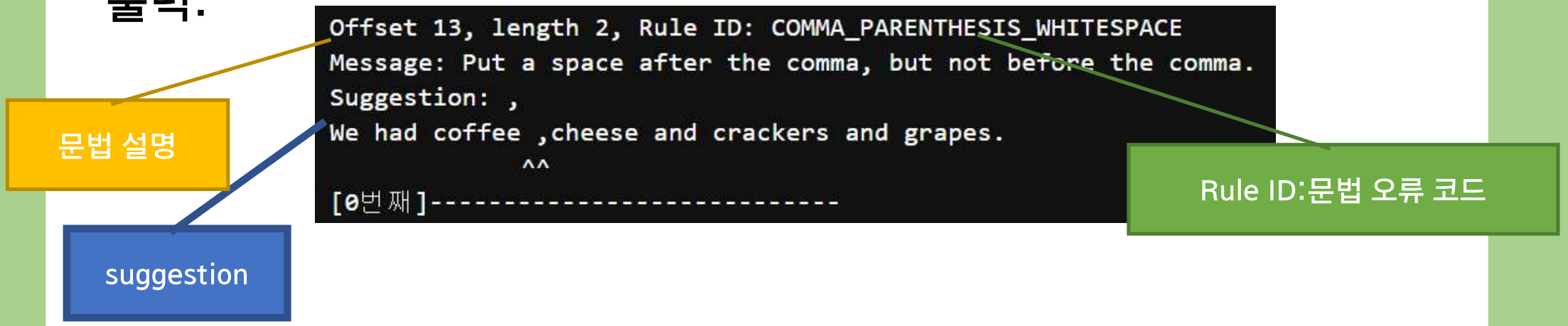
교과별 내용 요소 및 성취 기준

교과명	내용 요소	성취기준
정보교과	● 인공지능과 마이닝을 학습하고 이를 기반으로 자연어 분석을 어떤 원리로 어떻게 시행할 수 있는지 학습한다. ● 자신만의 grammar checker를 설계해보고 시행시켜본다. ● 웹크롤링을 통해 data set을 컴퓨터를 통해 효율적으로 모을 수 있음을 학습하고 시행해본다.	● 다양한 학문 분야의 문제 해결을 위해 설계한 알고리즘을 프로그램으로 구현하고 효율성을 비교·분석한다. ● 다양한 학문 분야의 문제 해결을 위한 알고리즘을 협력하여 설계한다. ● 함수의 개념을 이해하고 함수를 활용한 프로그램을 작성한다. ● 텍스트 기반 프로그래밍 언어의 개발 환경 및 특성을 이해한다. ● 인터넷, 응용 소프트웨어 등 컴퓨팅 도구를 활용하여 문제 해결을 위한 자료를 수집하고 분석한다.
영어교과	영어로 주어진 주제에 대한 작문을 해본다	● 비교적 다양한 주제에 관해 자신의 의견이나 감정을 쓸 수 있다.
수학교과	프로그램에서 발생한 오류를 통계 내보고 어떤 것이 가장 빈번하게 발생하는지 분석해본다.	● 이항분포의 뜻을 알고, 평균과 표준편차를 구할 수 있다. ● 모평균을 추정하고, 그 결과를 해석할 수 있다 ● 표본평균과 모평균의 관계를 이해하고 설명할 수 있다. ● 모집단과 표본의 뜻을 알고 표본추출의 원리를 이해한다. ● 이항분포의 뜻을 알고, 평균과 표준편차를 구할 수 있다.

GRAMMAR CHECKER 활용법

1. 학생이 Grammar checker에 자신이 영작한 문장 입력

출력:



GRAMMAR CHECKER 활용법

2. 학생이 Rule ID 입력-> 추가 영어 학습

```
Rule ID: COMMA_PARENTHESIS_WHITESPACE
```

```
from bs4 import BeautifulSoup
import requests
url = 'https://community.languagetool.or      le/list?offset=0&max=10&lang=en&filter={}&categoryFilter=&_action_list=Filte
query_txt= input ('Rule ID:')
query = query_txt

response = requests.get(url.format(query
soup = BeautifulSoup(response.content,       .parser')

article_lst = soup.select('tbody > tr >
for a_lst in article_lst[:2]:
    text = a_lst.text.strip()
    print(text)
Rule ID:
```

Rule ID가UPPERCASE_SENTENCE_START 일 때

해당 Rule ID가 의미하는 문법오류 설명

```
Rule ID: UPPERCASE_SENTENCE_START
Checks that a sentence starts with an uppercase letter
This house is old. it was built in 1950.
```

해당 Rule ID와 같은 오류가 있는 예시 문장

Web Crawling (web crawler)

- 자동화된 스크립트나 프로그램으로 웹페이지에 있는 데이터를 인덱싱 하는 과정

→자동화된 스크립트 (automated scripts): web crawler, spider, spider bot, and often shortened to crawler.

- 크롤러의 목적: 웹페이지가 무엇인지 아는 것: 유저들이 필요한 정보들을 가져올 수 있게 하는 것
- 데이터의 양이 점차 방대해짐 :2년 간에 90%이상 증가 (IBM,2013)

Web Crawling (web crawler)

- 방식

웹크롤러: 웹사이트의 로봇.txt file(검색엔진이 크롤할 수 있는 url 리스트 해둔 사이트 맵 포함)을 다운→ 링크를 통해 새로운 페이지 탐색→ 새로 찾아진 url을 크롤 큐 (crawl queue)에 저장

Web Crawling (web crawler)

- 파이썬 라이브러리

1. Requests : HTML을 웹사이트 서버에 요청해서 페이지의 데이터를 받을 수 있도록
2. Lxml: 방대한 데이터를 크롤 할 때 편리
3. Beautifulsoup: 가장 흔함, 자동적으로 인코딩 해줌, 요청하기가 쉬움
4. Selenium: 자동화된 웹 스크랩핑, 웹브라우저 자동화, 인간이 하는 형식과 비슷하게 수행 가능
5. Scrapy: 한번에 여러 개 웹사이트 크롤

실제 시행

[73]:
```
import pandas as pd
data_web = pd.read_html('https://www.engvid.com/english-resource/50-common-grammar-mistakes-in-english/')
data_web
```

...

[7]:
```
data_web [0] #table 속에 table -> 이어서 붙여야함, how--> 0 번 필드---, sentence는 sentence단위로
#right가 오른쪽으로 붙고, 0-50번까지 wrong, right column 으로 만들어서
# 문장들에 대해 학습
#ginger it--> 어디가 틀렸는지 말해주는 파트를 ㅎ한 칼럼 더 만들어서 ,sentece 단위로 , 계속 이어 붙이기
# 현재 웹데이터 -->
```

[7]:

	0	1
0	Wrong	I have visited Niagara Falls last weekend.
1	Right	I visited Niagara Falls last weekend.

[12]:
```
for i in range (len(data_web)):
    data_web [i].to_csv(f'./data_web{i}.csv', index= False) #각자 csv저장 #excel file
```

[74]:
```
#이어 붙이기
data_final = data_web [0].copy() #초기화 되어서 계속 concat가능
for i in range (len(data_web)):
    data_final = pd.concat([data_final,data_web[i]]) #data final  초기화 필요
```

[75]:
```
data_final.columns = ['정오','문장']
```

[76]:
```
#reset index
data_final.reset_index(drop=True,inplace = True)
```

[16]:
```
data_final.to_csv('./data_set.csv',index= False)
```

실제 시행

[78]:
```
import pandas as pd
import os

filePath = './'
fileAll = os.listdir(filePath)

for file in fileAll:
    rowsize = sum(1 for row in (open(filePath + file, encoding='UTF-8')))
    newsize = 2  # 쪼개고 싶은 행수 수준으로 입력. 이정도 행수는 200mb 이하임.
    times = 0
    for i in range(1, rowsize, newsize):
        times += 1   # 폴더 내 파일을 하나씩 점검하면서, 입력한 newsize보다 넘는 행을 쪼개줌
        df = pd.read_csv(filePath + file, header=None, nrows = newsize, skiprows=i)
        csv_output = file[:-4] + '_' + str(times) + '.csv'   # 쪼갠 수만큼 _1, _2... _n으로 꼬리를 달아서 파일명이 저장됨
        df.to_csv(filePath + csv_output, index=False, header=False, mode='a', chunksize=rowsize)
```

•••

[82]:
```
df= pd.read_csv('./data_set.csv')
df
```

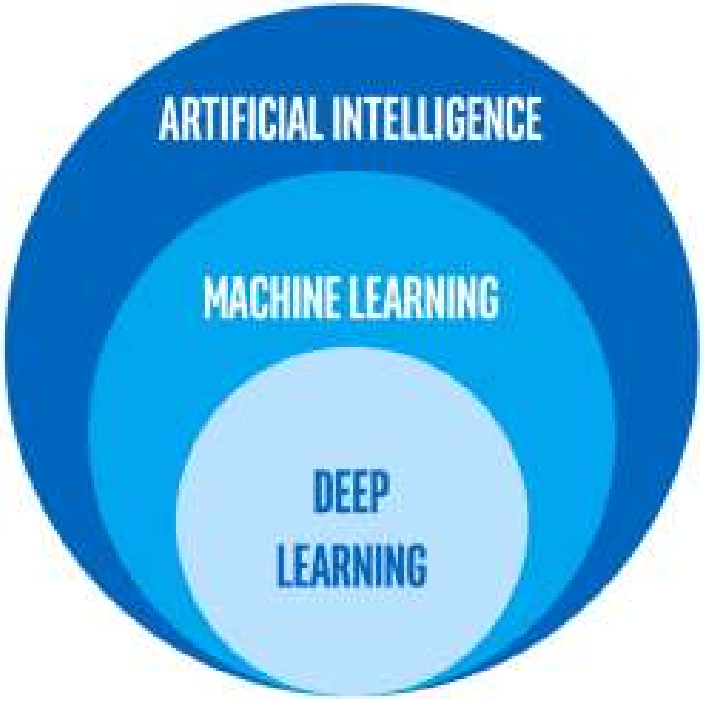
•••

[83]:
```
df1 = pd.read_csv('./extra_data.csv')
df1
```

[83]:

	0	1
0	Incorrect	I have a good news for you.
1	Correct	I have good news for you
2	Incorrect	Worry kills more the men than work.
3	Correct	Worry kills more men than work.
4	Incorrect	We learn not at the school, but in life.
...	...	...
337	Correct	My father will be home this afternoon.
338	Incorrect	I very love drawing.
339	Correct	I love drawing very much.
340	Incorrect	What of season do you like best ?

AI, machine learning, deep learning



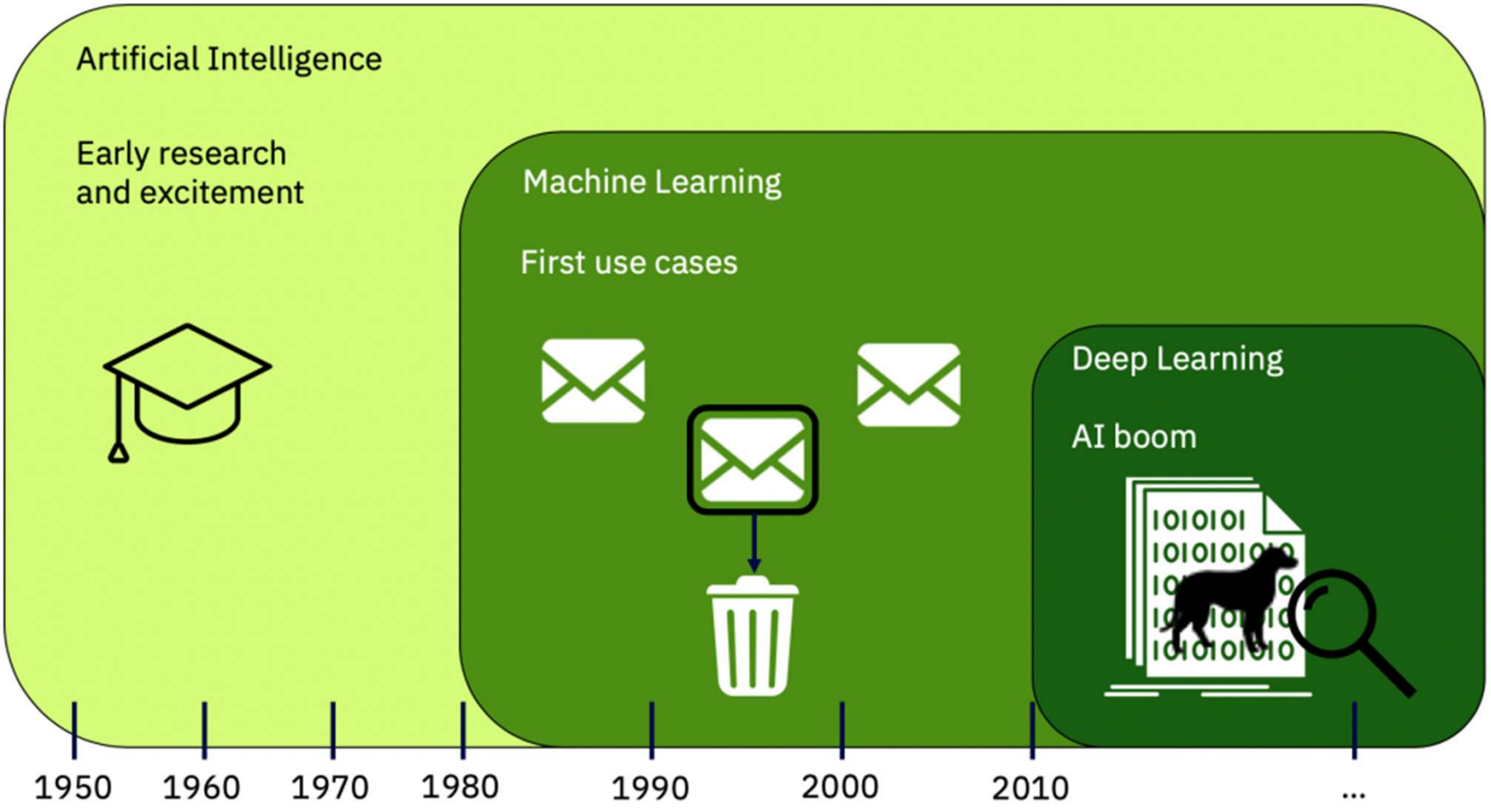
[AI]
스스로 배울 수 있는 기술을 통칭하는 말, 간단하게 말해서 인간의 행동을 흉내낼 수 있는 것

[Machine learning]
AI의 부분 집합
머신러닝 알고리즘- 통계학적 방법론을 사용-> 과거의 인간 행동 참고, 패턴 인식과 결정을 내리기 위해

➔예측의 능력을 발전시킬 수 있으나, 기존에 프로그램 되어 있는 데이터 사이에서만 탐색 가능

[deep learning]
스스로 학습할 수 있는 알고리즘, 다른 상황이나 다른 패턴의 데이터에 노출되어도 알고리즘은 적응할 수 있음

- 인공신경망 네트워크를 기반
- 더 많은 데이터를 필요로 함


Artificial Intelligence
Early research
and excitement
Machine Learning
First use cases
Deep Learning
AI boom
1950
1960
1970
1980
1990
2000
2010
...

NLP

- 컴퓨터에게 사람이 하는 동일한 방식으로 구어나 텍스트를 이해할 수 있는 능력을 주는 것
- 컴퓨터 언어학←통계학+머신러닝+딥러닝 모델
- 인간의 텍스트/음성 데이터를 작성 + 전체 의미를 이해 가능 + 작성자/발화자의 목적이나 감성을 인지
- Ex) 음성 비서(시리), 서비스 챗봇, 직원 생산성 평가
- Python and the Natural Language Toolkit (NLTK):NLP 작업+ 부수적 수행(parsing, word segmentation, stemming and lemmatization, tokenization)에 대한 라이브러리

실제시행

AI buliding

====

[34]:
```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier #ensemble: booasting 데이터 양 들릴때 , randomforest 알고리즘, decision tree
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.pipeline import Pipeline
```

[27]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
```

[32]:
```
x_train, x_test, y_train, y_test = train_test_split(df['sentence'],df['label'],test_size=0.3, shuffle=True)
```

[33]:
```
x_train.value_counts()
```

[33]:
```
 Thanks for giving me useful advices.              2
    Is it the rainy season in China?               1
    My watch isn't running.                        1
  Don't go in the sun.                             1
  Those who are absent, I shall punish them.       1
                                                  ..
 I forget my hat in the house.                     1
 Don't step in the grass.                          1
  I'll date her out this Friday.                   1
    He asked me why I had not gone to the party.   1
  The girl wants to get herself married.           1
Name: sentence, Length: 238, dtype: int64
```

[39]:
```
clf = Pipeline([('tfidf', TfidfVectorizer()), ('rf', RandomForestClassifier(n_estimators=100, n_jobs=1))])
```

[41]:
```
clf.fit(x_train, y_train)
```

[41]:
```
Pipeline(steps=[('tfidf', TfidfVectorizer()),
                ('rf', RandomForestClassifier(n_jobs=1))])
```

[42]:
```
y_pred = clf.predict(x_test)
```

[43]:
```
confusion_matrix(y_test, y_pred)
```

실제시행

```
[43]: array([[10, 45],
             [32, 16]])

[44]: print (classification_report(y_test, y_pred))

                  precision    recall  f1-score   support

         Correct       0.24      0.18      0.21        55
       Incorrect       0.26      0.33      0.29        48

        accuracy                           0.25       103
       macro avg       0.25      0.26      0.25       103
    weighted avg       0.25      0.25      0.25       103


[46]: accuracy_score(y_test, y_pred)

[46]: 0.2524271844660194

[47]: clf.predict(['I have a good news for you'])

[47]: array(['Incorrect'], dtype=object)

[48]: clf.predict(['Worry kills more the men than work'])

[48]: array(['Incorrect'], dtype=object)

[50]: clf2= Pipeline([('tfidf',TfidfVectorizer()), ('rf', SVC(C= 1000, gamma = 'auto'))])

[56]: clf2.fit(x_train,y_train)

[56]: Pipeline(steps=[('tfidf', TfidfVectorizer()),
                      ('rf', SVC(C=1000, gamma='auto'))])

[57]: y_pred1 = clf2.predict(x_test)

[58]: confusion_matrix(y_test,y_pred1)

[58]: array([[13, 42],
             [36, 12]])
```

실제시행

```
[60]: print (classification_report(y_test,y_pred1))
```

```
              precision    recall  f1-score   support

     Correct       0.27      0.24      0.25        55
   Incorrect       0.22      0.25      0.24        48

    accuracy                           0.24       103
   macro avg       0.24      0.24      0.24       103
weighted avg       0.25      0.24      0.24       103
```

```
[61]: clf2.predict(['I very love drawing.'])
```

```
[61]: array(['Incorrect'], dtype=object)
```

```
[64]: clf2.predict(['I love my boyfriend very much.'])
```

```
[64]: array(['Incorrect'], dtype=object)
```

```
[68]: import joblib
      joblib.dump(clf, './abc.pk1')
```

```
[68]: ['./abc.pk1']
```

```
[70]: load_model = joblib.load('./최종학습프로그램.pk1')
```

```
[71]: load_model.predict(['I have a good news for you.'])
```

```
[71]: array(['Incorrect'], dtype=object)
```

문제점

- 정확도가 매우 낮음 <== 데이터셋의 양이 부족하기 때문 (1000문장 내외)
- 다른 예시들

1. BERT (Bidirectional Encoder Representations from Transformers) is a big neural network architecture, with a huge number of parameters, that can range from 100 million to over 300 million.
2. The Corpus of Linguistic Acceptability (CoLA) in its full form consists of 10657 sentences from 23 linguistics publications

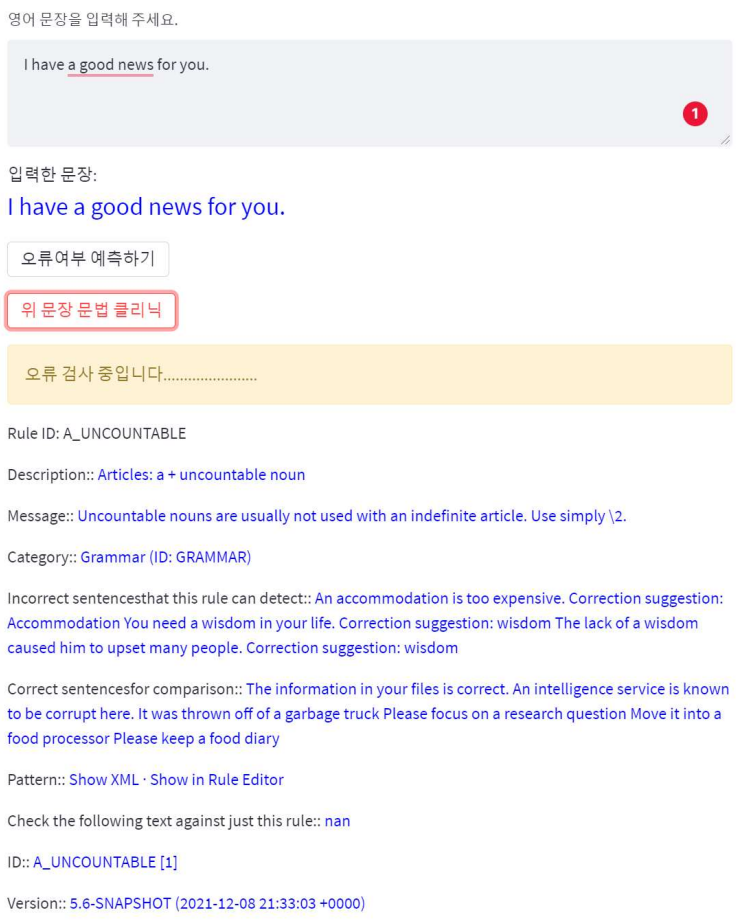
웹 앱

[92]:
```
from bs4 import BeautifulSoup
import requests
url = 'https://community.languagetool.org/rule/list?offset=0&max=10&lang=en&filter={}&categoryFilter=&_action_list=Filter'
query_txt= input ('Rule ID:')
query = query_txt

response = requests.get(url.format(query))
soup = BeautifulSoup(response.content, 'html.parser')

article_lst = soup.select('tbody > tr > td')
for a_lst in article_lst[:2]:
    text = a_lst.text.strip()
    print(text)
```

```
Rule ID: UPPERCASE_SENTENCE_START
Checks that a sentence starts with an uppercase letter
This house is old. it was built in 1950.
```

[94]:
```
import pandas as pd
tmp=[]
for i in range (10):
    url = f"https://community.languagetool.org/rule/list?lanf=en&offset={i}&max=10"
    tmp.append(pd.read_html(url))
```

웹 앱



1. 오류여부 예측하기 -> 인공지능을 기반으로 오류를 예측할 수 있음

2. AI활용을 위해 사용했던 DATA SET을 기반으로 grammar checker tool 인 language_tool_python을 이용하여 사용자가 문법의 정오 확인

2. 사용자가 오류의 RULE ID를 알 수 있고 RULE ID에 대한 설명을 description과 message 란에서 확인 할 수 있음
→ 영어 학습 가능

느낀점

- 컴퓨터교육과, 정보과 교사 → 학생의 흥미, 응용할 수 있는 가능성을 열어주어야 함
- 개인적 발전 방향→ 직접 코드를 짜서 인공지능을 모델링할 수 있는 실력 양성, 인공지능 모델을 수학적으로 이해하고 향후 데이터에 따라 직접 모델링을 해보고 싶다는 생각
    - 이미지 마이닝→ 현미경 사진 속 세포 소기관 인식 (과학과 융합)
    - 텍스트 마이닝→트위터 등 소셜미디어 api활용, 현상에 대한 사람들의 반응 및 인식 인지 해보기 (사회과 융합)