**[인공지능 고급 과정]**

**언어지능 프로젝트**

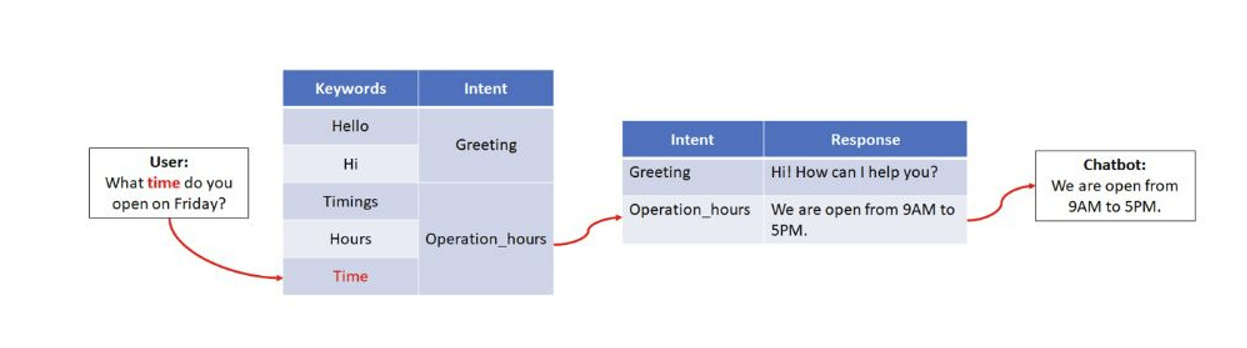
|  |  |
| --- | --- |
| 조 명 | ChatBot Maker |
| 프로젝트 주제 및 개요 | ChatBot Project |
| 프로젝트 수행 방향 | ▶수행방향  1. 챗봇 시스템 데이터 선택  2. 챗봇 시스템 기반이 되는 Rule 설정  3. 챗봇 시스템 구현  ▶수행도구  jupyter notebook, google colab  ▶데이터  KETI\_대화데이터\_일상\_오피스.txt |
| 프로젝트 조직  (구성원 및 역할) | ▶ 조장: 유지연  ▶ 조원1: 최원호  ▶ 조원2: 김현호 |
| 프로젝트 목적 | 자연어 처리 라이브러리를 학습에 따른 결과물의 하나로 챗봇을 선택하여 그 체계의 흐름을 파악하고, 다양한 챗봇 모델 중에서도 rule based model을 구현한다. |

**1. 프로젝트 내용**

* 1. **데이터**

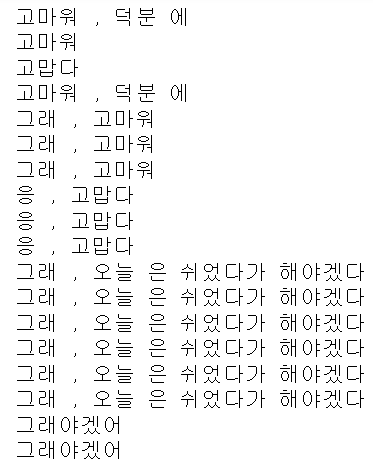
KETI\_대화데이터\_일상\_오피스.txt

**1.2 모델**



**1.3 실행**

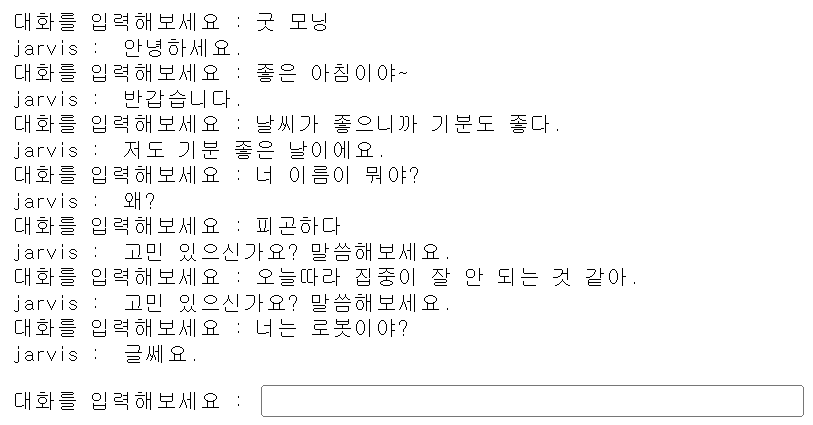
1 step: Konlpy의 Okt의 형태소 분석기를 이용해서 분류된 형태소를 data\_list에 저장했다.



2 step: 룰을 규정한 DataSet을 request데이터셋과 response데이터셋 결합하여 새로운 데이터셋(data0417) 생성



3 step: rule-based 모델링을 이용하여 일상 오피스 대화를 챗봇으로 구현했다.



1. **주요 기술**

* 언어 : Python
* 모델 : rule-based 모델
* 프레임워크 : Anaconda

1. **주요 소스코드**
   1. **데이터**

|  |
| --- |
| import pandas as pd  data = pd.read\_table('KETI\_대화데이터\_일상\_오피스.txt', sep='\t', header=None)  hello = data.loc[data[0]==1]  res = data.loc[data[0]==2]  from konlpy.tag import Okt  tokenizer = Okt()  data\_list = []  i = 0  for word in hello[1]:  test = tokenizer.morphs(word)  t = test[0]  for i in range(1, len(test)):  if test[i] == '.' :  continue  t += ' ' + test[i]  data\_list.append(t)  data\_list = pd.DataFrame(data\_list)  hello.reset\_index(inplace=True)  res.reset\_index(inplace=True)  data\_list.reset\_index(inplace=True)  data0417 = pd.concat([hello[1], data\_list, res[1]], axis=1)  data0417 = data0417.drop(['index'], axis=1)  data0417.columns = ['request', 'rule', 'response']  data0417.to\_csv('data0417.csv') |

**3.2 모델**

|  |
| --- |
| import pandas as pd  import random  chatbot\_data = pd.read\_csv('data0417.csv')  chat\_dic = {}  row = 0  for rule in chatbot\_data['rule']:  chat\_dic[row] = rule.split(' ')  row += 1 |

**3.3 실행**

|  |
| --- |
| def chat(request):  # 문장으로 찾기============================  for k, v in chat\_dic.items():  index = -1  for word in v:  try:  if index == -1:  index = request.index(word)  else:  if index < request.index(word, index):  index = request.index(word, index)  else:  index = -1  break  except ValueError:  index = -1  break  if index > -1:  return chatbot\_data['response'][k]    # 단어로 찾기=============================  number = []  for k, v in chat\_dic.items():  if request in v:  number.append(k)    if len(number) != 0:  num = random.choice(number)  return chatbot\_data['response'][num]  #==========================================    return '무슨 말인지 모르겠어요'  while True:  req = input('대화를 입력해보세요 : ')  if req == 'exit':  break  else:  print("jarvis : ", chat(req)) |