

목차

1. 의사결정나무와 로지스틱 회귀분석을 활용한 타이타닉 생존자 예측

(2021.05~2021.06 / Python)

2. Naive bayes 를 활용한 네이버 댓글 긍부정 분석기

(2019.12~2020.05 / Python)

3. 위키피디아 API 를 활용한 마인드맵 사이트

(2019.11 ~2019.12 / Python, HTML, CSS, JAVAScript)

4. 블로그 검색 API 를 활용한 언급 빈도 시각화

(2019.09 / R)

5. 상권분석 API 를 활용한 매물 클라우드 펀딩 플랫폼

(2019.06 / Oracle , Java, Spring, Html, CSS)

6. 트위터 API 를 활용한 관광지 검색 데이터 수집

(2017.06 / Python)

7. 야구 선수 데이터의 점수화를 통한 스카우팅 분석

(2015.12 / R, Oracle)

@ Git 주소 : <https://github.com/Q3333>

1. 의사결정나무와 로지스틱 회귀분석을 활용한 타이타닉 생존자 예측

[개인, Kaggle Competition에서 도전한 프로젝트]

<https://www.kaggle.com/c/titanic/submissions>

위 링크에서 도전해본 타이타닉 생존자 데이터를 이용한 결과 값 예측 프로젝트이다.

상세 내용은 아래 Git Link에 정리

https://github.com/Q3333/Study_Project/tree/master/Project%20-%20Titanic%20prediction

1-1. 첫 시도는 Kaggle 튜토리얼을 통해 배운 Pandas 조작으로 그룹핑을 해서 의미 있는 column을 찾아 보았다.

fare를 평균보다 더 낸 사람들의 집단을 정리해서 따로 DF를 생성하였다.

```
▶ rate_fare = sum(af)/len(af)

print("% of fare high who survived:", rate_fare)

% of fare high who survived: 0.676829268292683
```

결과는 약 67%가 생존

참고 : 전체 생존율은 약 38%이다.

```
▶ total = sum(train.Survived)/len(train.Survived)

print("% of survived:", total)

% of survived: 0.3838383838383838
```

1-2. 두 번째는 의사결정나무를 활용해서 생존 여부를 예측해 보았다.

의사결정나무 - target 변수와 feature 변수 나누기

Pclass~Embraked 항목을 feature로

```
feature_names = ["Pclass", "Sex", "Age",  
                 "SibSp", "Parch", "Fare", "Embarked"]
```

```
X_train = train[feature_names]  
  
print(X_train.shape)  
X_train.head()
```

Survived를 target으로 설정

```
label_name = "Survived"  
  
y_train = train[label_name]  
  
print(y_train.shape)  
y_train.head()
```

1-3. 마지막으로 target과 train을 나누는 과정에서 영감을 얻어서 로지스틱 회귀 분석을 시도했다.

로지스틱 회귀분석 - 변수선택(유의확률을 기반으로)

train 데이터에서 종속변수를 Survived로 두고, Pclass부터 Embarked까지의 모든 변수를 넣은 모형을 확인

```
import statsmodels.api as sm  
  
reg = sm.OLS.from_formula("Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked", train).fit()  
reg.summary()
```


2. Naive bayes 를 활용한 네이버 댓글 긍부정 분석기

[개인, 책을 통해 공부를 하면서 만들어 본 프로젝트]

전체 과정은 https://github.com/Q3333/Study_Project

2-1 유저에게 기사 URL과 학습 데이터의 개수를 입력 받음

```
In [2]: URL = input("네이버 뉴스 기사 댓글창의 URL을 복사해서 입력해주세요")

#https://sports.news.naver.com/news.nhn?oid=477&aid=0000232510&m_view=1&sort=LIKE - 손흥민, 공격이 많음

네이버 뉴스 기사 댓글창의 URL을 복사해서 입력해주세요https://news.naver.com/main/ranking/read.nhn?sid1=&rankingType=popular_memo&oid=001&id=0011552671&date=20200416&type=1&rankingSectionId=100&rankingSeq=1&m_view=1&includeAllCount=true&m_url=%2Fcomment%2Ffall.nhn%3FserviceId%3Dnews%26gno%3Dnews001%2C00011552671%26sort%3Dlikability

In [3]: train_number = input("원하는 학습 데이터의 개수를 입력해 주세요")

원하는 학습 데이터의 개수를 입력해 주세요20
```

2-2 유저가 학습데이터의 긍,부정 여부를 판단

```

else :
    print("pass")
    train_count = train_count + 1
    continue

train_list.append((text,NB))
train_count = train_count + 1
print("")

print(train_list)

```

0번째 train data 입니다. 남은 학습 수 : 30
 손흥민 지켰다
 학습 데이터의 긍정 여부 입력해 주세요
 긍정 : 1 , 부정 : 2, 중립 : 3 : 1
 긍정입니다

1번째 train data 입니다. 남은 학습 수 : 29
 레이나 야신모드에 놀라고 손흥민 극장골에 넣었다
 학습 데이터의 긍정 여부 입력해 주세요
 긍정 : 1 , 부정 : 2, 중립 : 3 : 1
 긍정입니다

2번째 train data 입니다. 남은 학습 수 : 28
 손자들이 암만욕해도 결국 메이스는 손흥민이다
 학습 데이터의 긍정 여부 입력해 주세요
 긍정 : 1 , 부정 : 2, 중립 : 3 : 1

2-3 형태소 분석 후 학습된 데이터를 바탕으로 단어의 긍,부정 여부를 판단

```
""")
Most Informative Features
    틀이다/Verb = True          neg : pos = 5.8 : 1.0
    끄다/Verb = True            neg : pos = 5.8 : 1.0
    걸/Noun = True              neg : pos = 5.8 : 1.0
    이/Determiner = True        neg : pos = 5.8 : 1.0
    되다/Verb = True            neg : pos = 5.8 : 1.0
    손/Noun = True              neg : pos = 2.5 : 1.0
    마지막/Noun = True          neg : pos = 2.5 : 1.0
    되다/Verb = False           pos : neg = 2.4 : 1.0
    하다/Verb = True            neg : pos = 1.9 : 1.0
    ./Punctuation = True        neg : pos = 1.9 : 1.0

[10]: # 위에서 만든 모델에 전체 text data를 넣어 분석
      text_count = 0
```

2-4 결과

'그~~~~~렇게 복하고도 더 팔게있다고 능력살이 날려드느 손까늘.. 지갑지 않니??? 오늘 경기에서 설령설령 원다고 복하더니 결국 극상 반쯤었
는데 우짜지??'의 긍부정 판정 결과는 pos입니다.
'ㅋㅋ 이게 왜 세탁이나 ㅋㅋ 평점도 최고점 받았는데 ㅋㅋ 진짜 왜 세탁이란건지 하나도 모르겠다 5경기 연속골인데 ㅋㅋ 진짜 얼마나 까고
싶어서 안달난거냐 ㅋㅋ'의 긍부정 판정 결과는 pos입니다.
'진짜 손흥민 개대박이다 진짜 소름돋는다'의 긍부정 판정 결과는 pos입니다.
'솔직히 손흥민 이왕의 선수 각 아니냐?'의 긍부정 판정 결과는 pos입니다.

```
[11]: print("pos 표현의 갯수는 " + str(pos_count) + "개 이며, 약 " + str(round(pos_count/total_length*100,2)) + "%입니다.")
      print("neg 표현의 갯수는 " + str(neg_count) + "개 이며, 약 " + str(round(neg_count/total_length*100,2)) + "%입니다.")
```

pos 표현의 갯수는 236개 이며, 약 98.74%입니다.
neg 표현의 갯수는 3개 이며, 약 1.26%입니다.

@ 긍,부정의 비율이 어느 한 쪽이 치우쳐진 기사일 경우에는 다른 한 쪽의 학습 데이터가 부족해져서 전체 학습데이터의 양을 늘려야 할 것 같다.

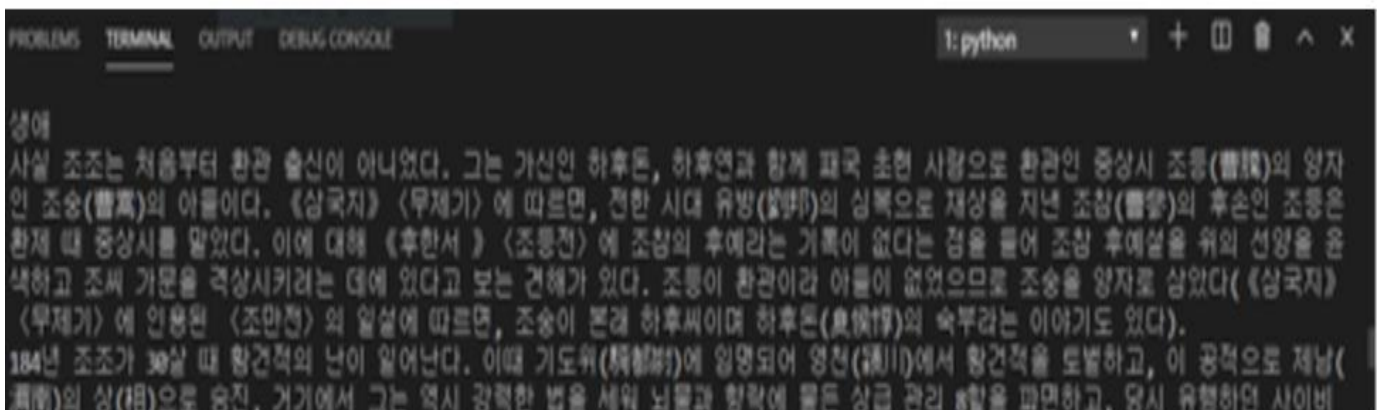
3. 위키피디아 API 를 활용한 마인드맵 사이트

[4인, 멀티캠퍼스 최종 프로젝트])

역할 : 데이터 수집, 전처리, 제공 및 전체적인 기능 개발

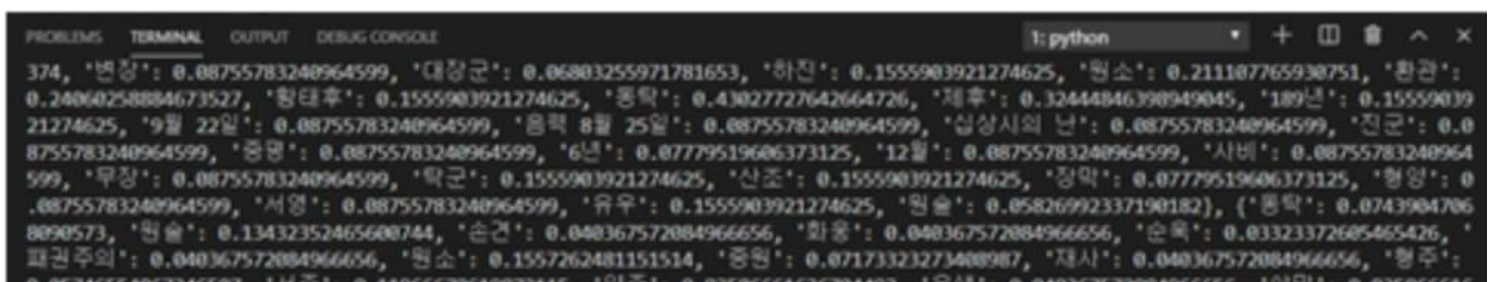
전체 내용 : https://github.com/Q3333/Ask_and_Wiki_2019_12

3-1 위키피디아 API에서 텍스트 데이터 추출

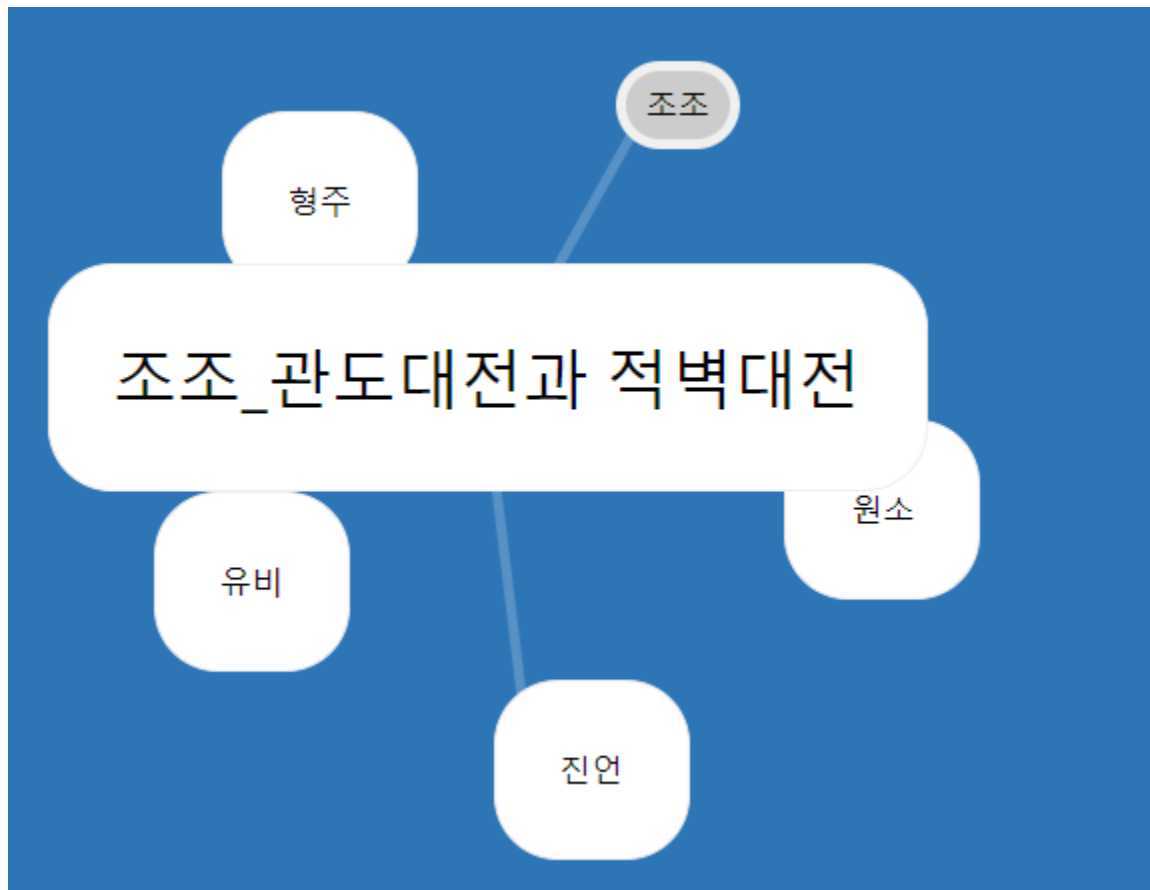


3-2 서브섹션이 있는 경우 TF-IDF를 통해 주요 단어를 추출

- 1 생애
 - 1.1 출생
 - 1.2 황건적의 난
 - 1.3 동탁 제거와 서주 공략까지
 - 1.4 관도대전과 적벽대전
 - 1.5 삼국 정립과 죽음



3-3 TF-IDF 분석된 값 중 가장 연관성이 높은 단어들을 마인드맵으로 뿌려줌



4. 네이버 블로그 검색 API 를 활용한 언급 빈도 시각화

[개인, 멀티캠퍼스에서 배운 R을 이용해 네이버 블로그 검색 API 의 결과를 워드클라우드로 시각화])

결과 :



주요 코드 :

키워드 설정, 조회 개수 설정, 네이버 API 연동

```

3 #기본 URL
4 urlStr <- "https://openapi.naver.com/v1/search/blog.xml?"
5 #검색어 설정 및 UTF-8 URL 인코딩
6 searchString <- "query=코타키나발루"
7 #UTF-8 인코딩
8 searchString <- iconv(searchString, to="UTF-8")
9 #URL 인코딩
10 searchString <- URLencode(searchString)
11 searchString
12
13 #나머지 요청 변수 : 조회 개수 1000개, 시작페이지1, 유사도순 정렬
14 etcString <- "&disply=1000&start=1&sort=sim"
15
16 #URL 조합
17 reqUrl <- paste(urlStr, searchString, etcString, sep="")
18 reqUrl
19

```

데이터 전처리, 워드클라우드 출력

```

refinedStr <- result
#XML 태그를 공란으로 치환
refinedStr <- gsub("<\\/?)(\\w+)*([^<>]*)>", " ", refinedStr)
#단락을 표현하는 불필요한 문자를 공란으로 치환
refinedStr <- gsub("[[:punct:]]", " ", refinedStr)
#영어 소문자를 공란으로 치환
refinedStr <- gsub("[a-z]", " ", refinedStr)
#영어 대문자를 공란으로 치환
refinedStr <- gsub("[A-Z]", " ", refinedStr)
#숫자를 공란으로 치환
refinedStr <- gsub("[0-9]", " ", refinedStr)
#여러 공란은 한 개의 공란으로 변경
refinedStr <- gsub(" +", " ", refinedStr)

#제외할 특정 단어를 정의
excluNouns <- c("부터", "하기", "이번", "오래전", "고고해주세요")

nouns <- nouns [!nouns %in% excluNouns ]
nouns [1:50]

#빈도수 기준으로 상위 50개 단어 추출
wordT <- sort(table(nouns), decreasing=T)[1:50]
wordT
library(wordcloud2)
wordcloud2(wordT, size=3, shape="diamond")

```

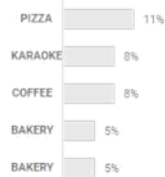
5- 상권분석 API를 이용한 데이터 시각화

[5인, 멀티캠퍼스에서 만난 분들과 참여한 해커톤 프로젝트])

역할 : 수집된 데이터 분석 및 시각화 및 전체적인 웹 개발

전체 프로젝트에 관한 정리는 https://github.com/Q3333/hackathon_2019_06

투표 현황



05 Comments



Q3333

2019년 6월 30일 오전 6시 53분

REPLY

동네에 비어있는 가게가 많았는데 KB FBI덕분인지는 모르겠지만 확실히 요새들어
줄어든 느낌인 것 같아요.



RubinYoon

2019년 6월 30일 오전 6시 43분

REPLY

상권분석 서비스 진짜 신기해요! 내가 생각했던 느낌이란 딱 맞게 나옴!ㅋㅋ

상권 분석



활성도



과밀도



안정성



성장성

업종선택 ▼

리워드선택 ▼

수요조사 참여하기

전체 프로젝트는 웹 플랫폼 개발이었는데, 그 중 상권 분석의 4지표를 보기 쉽게 제공하는 방법이 뭘까 고민을 하다가 전체 동네의 평균과 최대값, 최소값을 구한 뒤 해당 점포의 위치와 비교를 하여 나온 차이만큼 점수화 하여 별점으로 보여 주었다.

6 - 트위터 API를 활용한 관광지 검색 데이터 수집

[개인, 학부 Python 수업에서 배운 내용을 토대로 데이터 크롤링])

결과 화면, 수집 키워드는 ['관광','여행','갯다움','갯는데','다녀옴']) 이었고, 관광지 이름들을 변수로 언급 될 때 마다 카운팅 되는 식으로 구현하였다.

```
C:\windows\py.exe
RT @comet_56baek: 통대분 크레페 만드는거 거의 마약팔급... https://t.co/ZoexlecVUF
RT @cybereater: 홀대 크로와플. 주문하자마자 구워주셔서 따끈하고 바삭하다. 상당히
가 좋을 듯. 누텔라 추가도 가능! 기본 크로와플1800원, 바나나/딸기/키위/파인애플+휘
RT @rapmonster_net: 경북궁 남준은 마치 제 집을 찾은 어린 왕자처럼 편안해보여옴...
https://t.co/N5FrVfSCP7 http://t.co/hclxQwwSlY
명동 : 925
남산 : 17
가로수길 : 40
경복궁 : 18
인사동 : 2525
통대분 : 1500
창덕궁 : 3
홀대 : 704
독촌한옥마을 : 1
이태원 : 222

Total tweet : 5879
```

트위피 라이브러리를 사용한 주요 코드

```
from tweepy import Stream
from tweepy import OAuthHandler
from tweepy.streaming import StreamListener

tweet_stream = open('tour.txt', 'a') #raw data를 저장시키기 위해서 txt파일을 변수에 저장한다.

class Listener(StreamListener):

    def on_data(self, data):
        try :
            tweet_stream.write(data) #트위터에서 가져온 데이터를 저장하는 과정
            tweet_stream.write('\n')
            print('suc')
        except :
            print('Err')

        return True

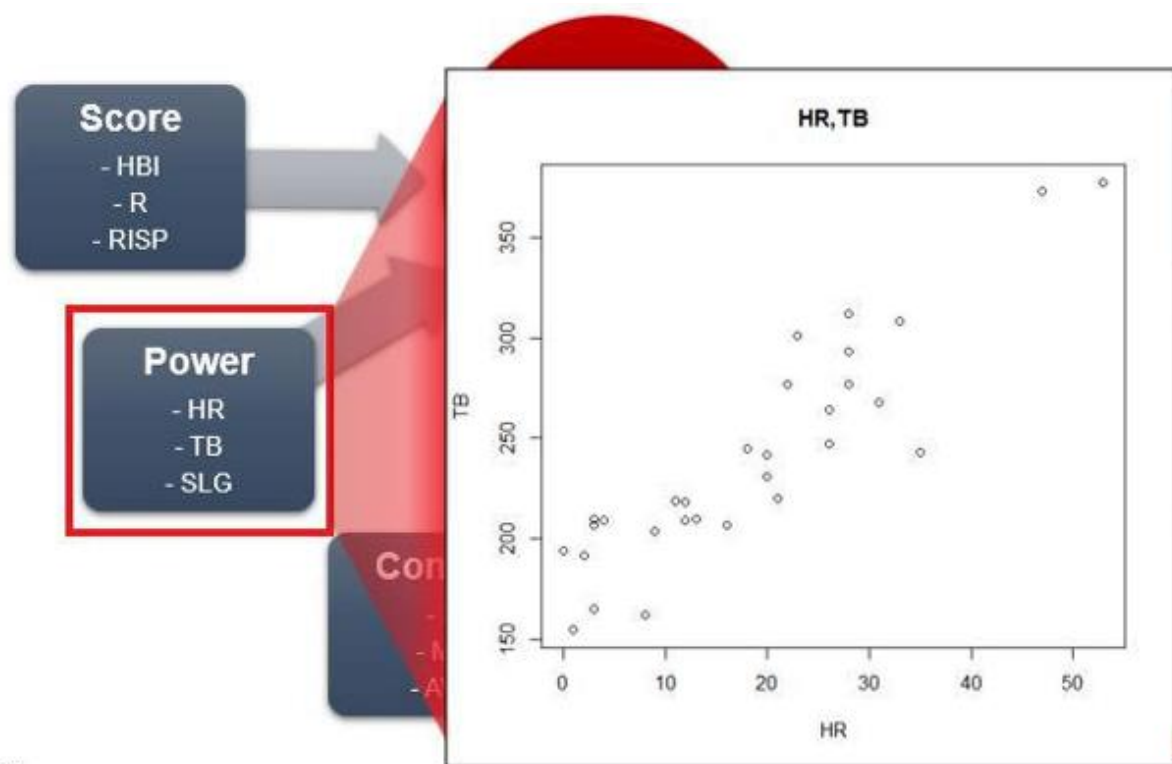
    def on_error(self, status):
        print('error : ', status)
```

7 - 야구 선수 데이터의 점수화를 통한 스카우팅 분석

[2인, 데이터베이스(SQL) 수업 마무리 프로젝트])

역할 : 데이터 수집 및 분석과 아이디어 제안

R을 이용한 상관관계 분석과 데이터베이스 수치들을 이용한 점수화



홈런 , 타점의 상관관계 오른쪽 위 방향으로 정비례 한다는 걸 보여준다.

이러한 분석과 비슷한 상관관계 분석을 수 차례 진행한 내용을 바탕으로 기록들

을 6가지로 분류하고 그룹핑해서 점수화.

점수화 과정

```
insert into main_table(player_name, SCORE_VALUE, POWER_VALUE, CONTACT_VALUE,  
                        BASE_RUNNING_VALUE, ON_BASE_VALUE, DEFENSE_VALUE)  
select PLAYER_NAME,  
       round((((R+RBI)/5)+(RISP*100)),2) as score,  
       round((SLG+(HR_number/TB))*100,1) as power,  
       round((((MH+H)/5)+(AVG*100)),2) as contact,  
       round(((50+SB)-CS-((OOB+PKO)/2)),2) as base_running,  
       round((((BB+HBP)*0.8)/2+(OBP*100)),2) as on_base,  
       round(((FPCT-(E/G))*100),2) as defense
```

점수화를 이용한 선수 검색

Ex) 득점 능력이 70이상이거나 파워능력이 80이상인 선수
들의 이름과 팀명

```
select player_name, team_name, SCORE_VALUE, POWER_VALUE  
from main_table  
where score_value >= 70 or power_value >=80;
```

결과

PLAYER_NAME	TEAM_NAME	SCORE_VALUE	POWER_VALUE
1 Thames	nc	85	91.6
2 yuhanjun	nx	83.3	65.5
3 parkbyunggho	nx	92.5	85.5
4 nasungbum	nc	86.1	64.3
5 kimhyunsoo	ds	78.1	64.2
6 parksukmin	ss	83.2	65.6
7 choihyungwoo	ss	73.6	67
8 aduchi	rd	77.6	65.3
9 choijunsuk	rd	71.4	64.5