INDEX

4. ブログ検索 API を活用した言及頻度の視覚化

(2019.09 / R)

5. 商圏分析 API を活用した売り物クラウドファンディングプラットフォーム

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

6. ツイッターAPI を活用した観光地検索データ収集

(2017.06 / Python)

7. 野球選手データの点数化によるスカウティング分析

(2015.12 / R, Oracle)

1. 意思決定ツリーとロジスティック回帰分析を活用したタイタニック生存者予測

[個人、Kaggle Competition で挑戦したプロジェクト])

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

上のリンクで挑戦してみたタイタニック生存者データを利用した結果値予測プロジェクト。

詳細は下記 GitLink にまとめ

https://github.com/Q3333/Study_Project/tree/master/2021_Titanic_prediction

1-1.　最初の試みは Kaggle チュートリアルで学んだ Pandas 操作でグルーピングをして意味のある column を探す

翻訳）

fare を平均よりもっと払った人たちの集団を整理し、DF を生成した。

結果は訳６７％が生存、　＊全体の生存率は３８％

つまり、高い部屋に泊まったお客たちの生存率は、平均より３０％ぐらい高かった。

1-2. 次は意思決定木を活用して生存可否を予測

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()
```

翻訳）Decision Tree　ー　Target と　Feature でデータ仕分け

Pclass ~ Embarked っていう名前の項目を Feature として設定。

生存率の Survived データを target value として設定。

1-3. 最後に target と train を分ける過程でインスピレーションを得てロジスティック回帰分析を試みる

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()
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.7897	0.070	11.211	0.000	0.651	0.928
Sex[T.1]	0.5031	0.028	17.824	0.000	0.448	0.558
Pclass	-0.1752	0.020	-8.849	0.000	-0.214	-0.136
Age	-0.0060	0.001	-5.546	0.000	-0.008	-0.004
SibSp	-0.0419	0.013	-3.216	0.001	-0.067	-0.016
Parch	-0.0156	0.018	-0.855	0.393	-0.051	0.020
Fare	0.0003	0.000	0.994	0.320	-0.000	0.001
Embarked	0.0423	0.020	2.101	0.036	0.003	0.082

Logistic regression model を使って、最も意味がある VALUE を探索。

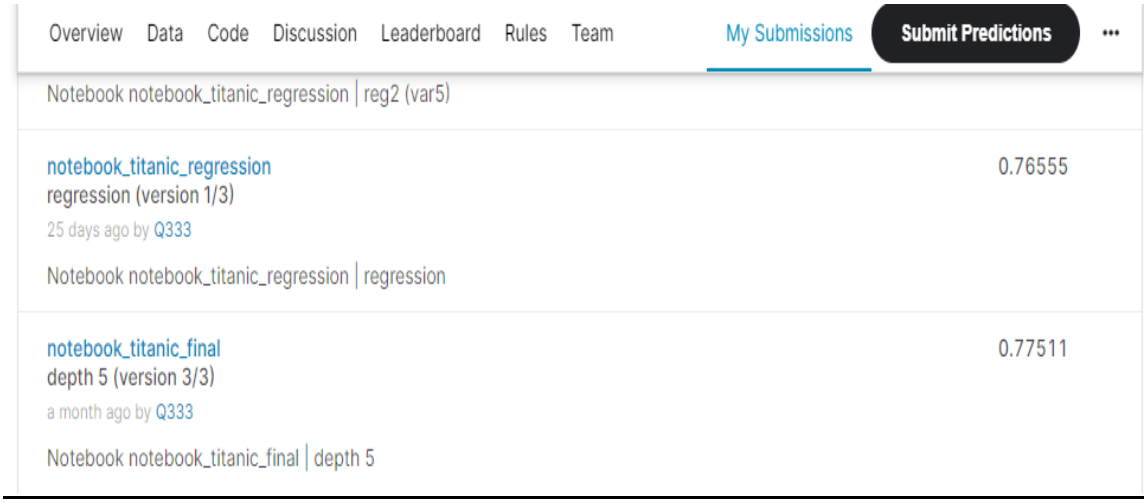
유의확률이 0.05보다 높은 두 변수를 제거하고 다시 확인

```
reg = sm.OLS.from_formula("Survived ~ Pclass + Sex + Age + SibSp + Embarked", train).fit()
reg.summary()
```

로지스틱 모델 학습

로지스틱 회귀모형에 각각 학습 데이터들을 적용시킴

```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(x_train, y_train)
```



それぞれのスコアはロジスティックが 76、意思決定木が 77 点

2. Naive bayes を活用した WEBSITE コメント不正分析機　（NLP)

[個人、本を通じて勉強をしながら 作ってみたプロジェクト])

全体の過程は下の LINK

https://github.com/Q3333/Study_Project/tree/master/2020_Naver_comment

2-1 ユーザーに記事 URL と学習データの個数を入力してもらう

```
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, 긍정이 많을

네이버 뉴스 기사 댓글창의 URL을 복사해서 입력해주세요https://news.naver.com/main/ranking/read.nhn?sid1=&ra
id=001155267 1&date=20200416&type=1&rankingSectionId=100&rankingSeq=1&m_view=1&includeAllCount=true&m_url=%
3Dnews%26gno%3Dnews001%2C001155267 1%26sort%3Dlikability

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

원하는 학습 데이터의 개수를 입력해 주세요20
```

翻訳）IN2：記事の URL を入力してください。

IN3：学習データの数を入力してください。

2-2 ユーザーが学習データの肯定·否定を判断

```
    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 지렸다
학습 데이터의 긍부정 여부를 입력해 주세요
긍정 : 1 , 부정 : 2, 중립 :3 : 1
긍정입니다

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

2번째 train data 입니다. 남은 학습 수 : 28
손까들이 암만욕해도 결국 에이스는 손흥민이다
학습 데이터의 긍부정 여부를 입력해 주세요

긍정 : 1 , 부정 : 2, 중립 :3 :
```

翻訳）

データの POSITIVE,NEGATIVE の判断をしてください。

Positive : 1

Negative : 2

Neutral ： 3

@学習するデータの数によって、判断の精密度が上がる。

＠学習データ２０個、TARGET データ２０００個みたいな動きも一応可能。

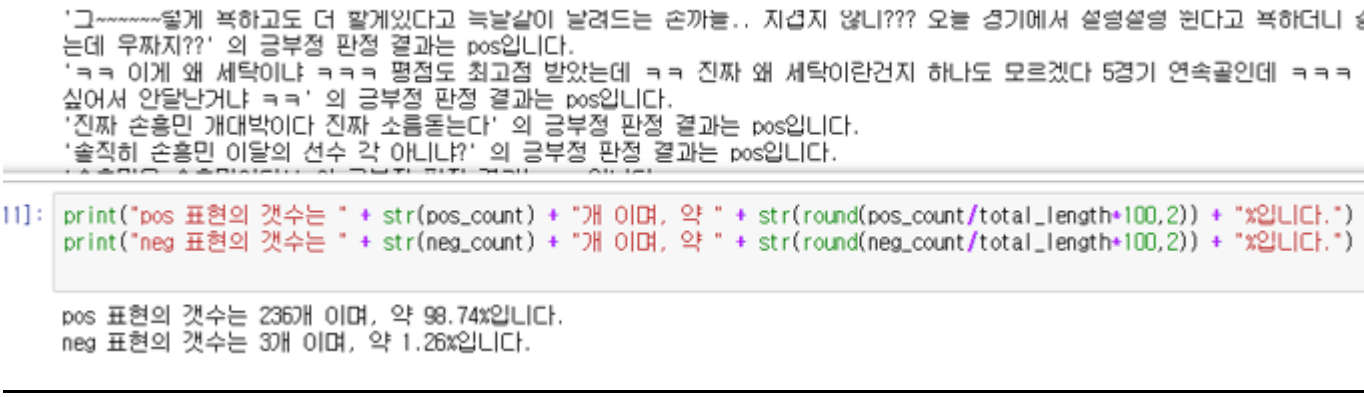
2-3 形態素分析後、学習されたデータを基に単語の肯定·否定可否を判断

```
  """)

Most Informative Features
                들이다/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 結果



```
'그~~~~~~렇게 복하고도 더 할게있다고 늑날같이 날려드는 손까늗.. 지넙지 않니??? 오늘 경기에서 설렁설렁 뛴다고 복하더니 
는데 우짜지??' 의 긍부정 판정 결과는 pos입니다.
'ㅋㅋ 이게 왜 세탁이냐 ㅋㅋㅋ 평점도 최고점 받았는데 ㅋㅋ 진짜 왜 세탁이란건지 하나도 모르겠다 5경기 연속골인데 ㅋㅋㅋ
싶어서 안달난거냐 ㅋㅋ' 의 긍부정 판정 결과는 pos입니다.
'진짜 손흥민 개대박이다 진짜 소름돋는다' 의 긍부정 판정 결과는 pos입니다.
'솔직히 손흥민 이달의 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.
```

@ 改善事項:肯定·否定の割合がどちらか一方に偏った記事の場合には、もう一方の学習データが不足し、全体の学習データの量を増やす必要がありそう

3.ウィキペディア API を活用したマインドマップサイト構築

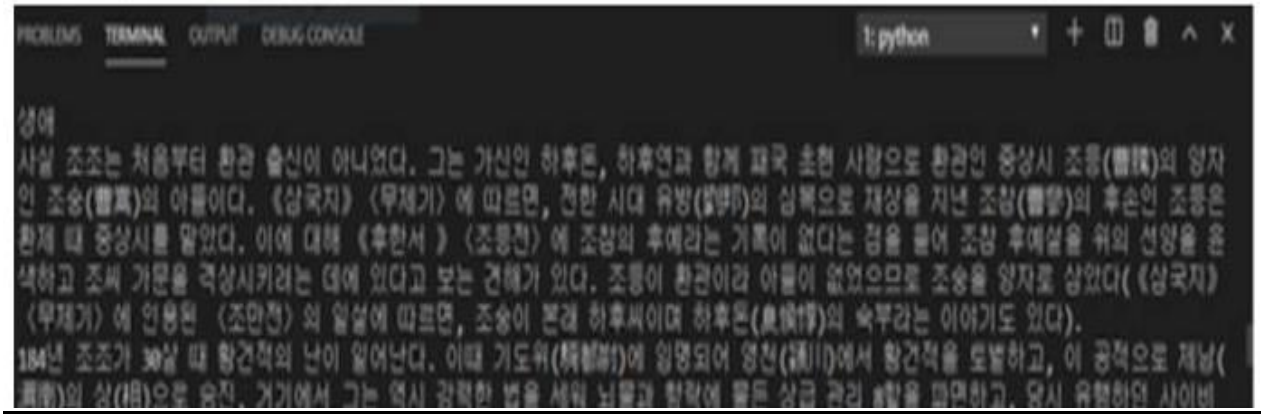
[4 人、*マルチキャンパス最終プロジェクト])

＊Multicampus : SAMSUNG 系企業の教育専門,企業研修専門会社

役割: データ収集、前処理、提供、および全体的な機能開発

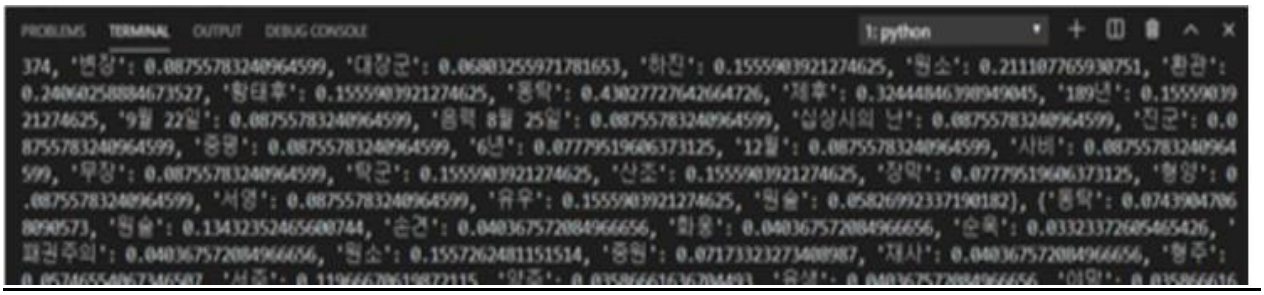
全体内容 ：https://github.com/Q3333/Ask_and_Wiki_2019_12

3-1 ウィキペディア API からテキストデータを抽出



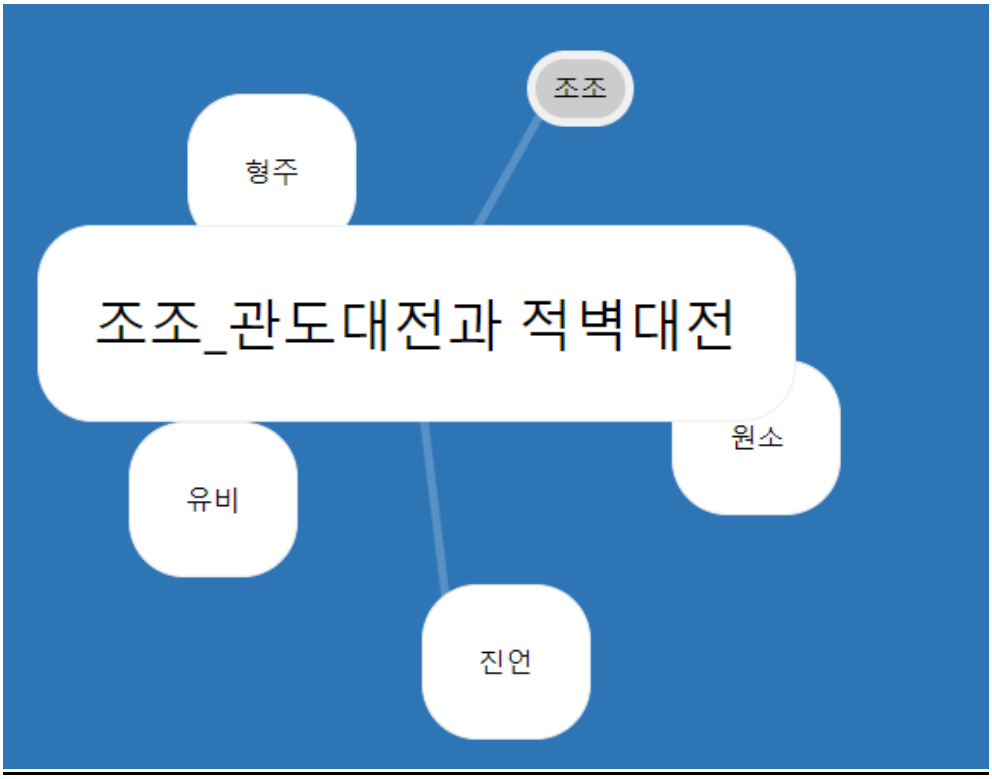
翻訳）三国志　ジョジョの WIKI 情報

3-2 サブセクションがある場合、TF-IDF を通じて主要単語を抽出



翻訳）WIKI の中にある単語を TF－IDF で重要単語選別

3-3TF-IDF 分析された値の中で最も関連性の高い単語をマインドマップで撒く



結果：ジョジョと最も関係がある人物や事件が登場

４．ブログ検索 API を活用した言及頻度の視覚化

[個人、マルチキャンパスで学んだ R を利用してネイバーブログ検索 API の結果をワードクラウドで視覚化]）

結果：

主要コード

キーワード設定、照会数設定、ネイバーAPI 連動

```
#기본 URL
urlStr <- "https://openapi.naver.com/v1/search/blog.xml?"
#검색어 설정 및 UTF-8 URL 인코딩
searchString <- "query=코타키나발루"
#UTF-8 인코딩
searchString <- iconv(searchString, to="UTF-8")
#URL 인코딩
searchString <- URLencode(searchString)
searchString

#나머지 요청 변수 : 조회 개수 1000개, 시작페이지1, 유사도순 정렬
etcString <- "&dispaly=1000&start=1&sort=sim"

#URL 조합
reqUrl <- paste(urlStr, searchString, etcString, sep="")
reqUrl
```

翻訳）

＃基本 URL

＃検索ワード設定、

＃ENCODING

＃照会件数、ページ、SORT など設定

データ前処理、ワードクラウド出力

```
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","고고해주세용")

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")
```

＃データ前処理（EX.空欄削除、特殊文字削除、英語のアルファベット統一など）

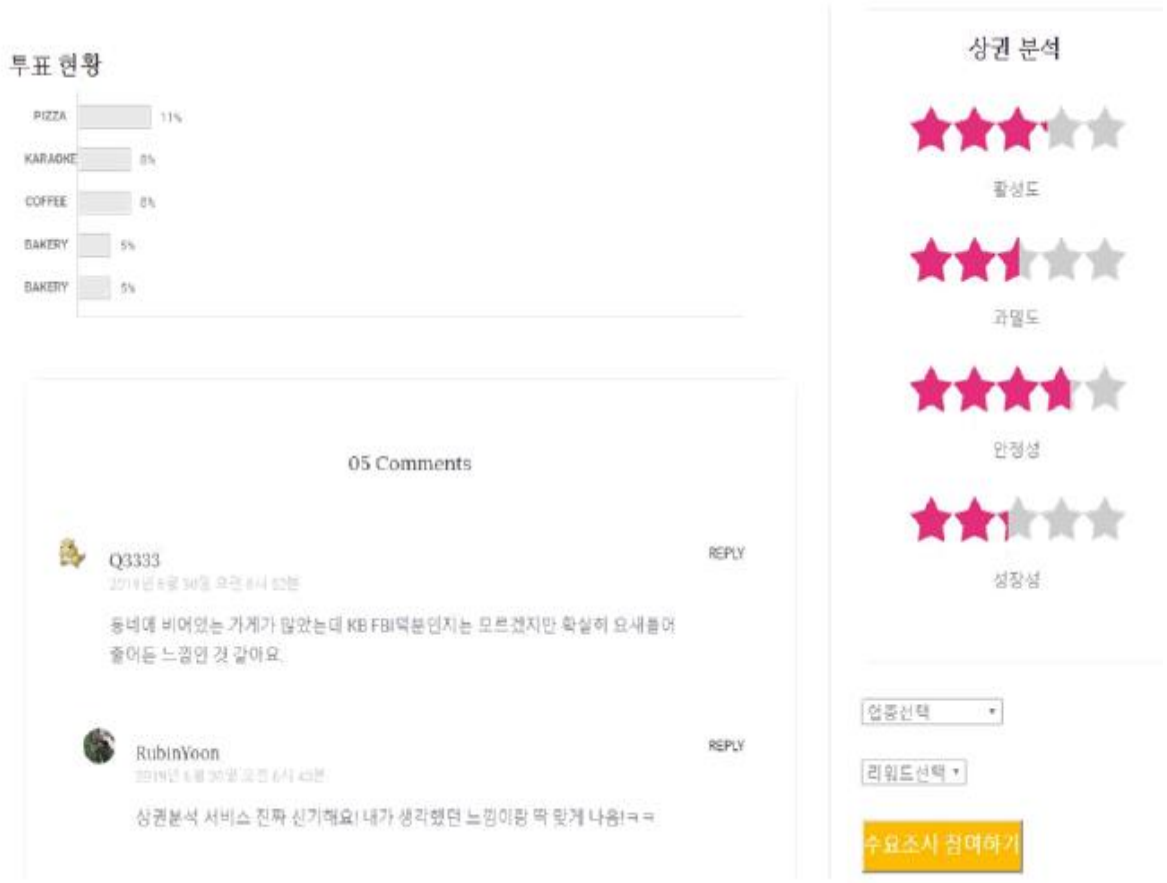
５．商圏分析 API を利用したデータ視覚化

[5 人、マルチキャンパスで会った方々と参加したハッカソンプロジェクト])

役割:収集されたデータの分析と視覚化、および全体的なウェブ開発

全体プロジェクトに関する整は下の LINK

https://github.com/Q3333/hackathon_2019_06



全体のプロジェクトはウェブプラットフォームの開発でしたが、その中で商圏分析の 4 指標を見やすく提供する方法は何かと悩み

町全体の平均と最大値、最小値を求めた後、該当店舗の位置と比較して出た差だけ点数化して星で見せることにした。

６．ツイッターAPI を活用した観光地検索データ収集

[個人、学部 Python 授業で学んだ内容を基にデータクロリング])

トゥイーピーライブラリを使用した主なコード

```
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)
```

結果画面

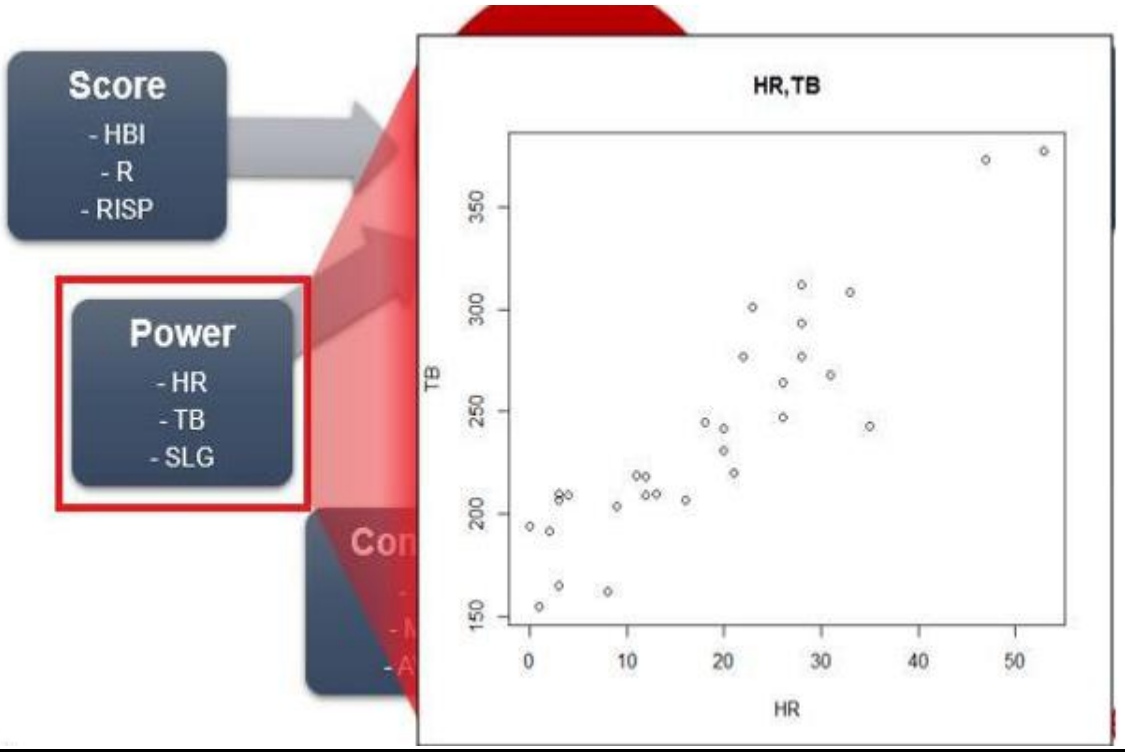
収集キーワードは「観光」、「旅行」、「行ってきた」、「行った」）などであり、観光地の名前を変数として言及されるたびにカウントされるように実装

7. 野球選手データの点数化によるスカウティング分析

[2 人、データベース(SQL)授業終了プロジェクト])

役割:データ収集と分析とチームリーダー

R を利用した相関分析とデータベース数値を利用した点数化



本塁打、打点の相関関係、右上方向に正比例するということを示す

このような分析と似た相関関係の分析を数回行った内容をもとに記録

を 6 つに分類し、グルーピングして点数化。

点数化過程

```
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
```

点数化を利用した選手検索

Ex) 득점능력이 70이상이거나 파워능력이 80이상인 선수들의 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	parkbyungho	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

翻訳）特定能力が７０点以上だったり、POWER VALUE が８０以上の選手たちの名前とチーム名