**프로야구 Application**

-Data Base Team Project-

**공팔포에버팀**

팀장 2008-11705 임채경

2008-11594 류길현

2008-11662 유재성

**1.Description**

1982년 시작된 프로야구는 2014년 현재 33번째 시즌을 준비하고 있다. 2008 베이징올림픽에서 대한민국 야구 대표팀이 금메달을 딴 이후로 야구의 인기는 매년 높아지고 있으며, 그에 따라 야구 팬들의 수준 역시 높아지고 있다. 과거에는 단순히 경기를 관람하고 좋아하는 선수의 멋진 플레이를 감상하는데 그쳤다면 이제는 선수들의 세세한 기록을 찾아보고 그에 따라 승부를 예측하는 것까지 팬들의 요구 사항이 확장되고 있다. 또한 야구는 다른 어떤 sports와 비교해 보아도 기록하는 stat들이 특히 더 많다. 전통적인 stat인 타율, 안타, 홈런, 승, 방어율, 탈삼진 등에서부터 war, ops+ 등의 현대적인 sabermetrics (실증적인 야구 분석을 위해 야구 통계학자들이 개발한 stat)에 이르기 까지 계속해서 야구 stat은 개발되고 발전되어 오고 있다. 그러한 추세에 발맞추어 우리 조에서는야구의 기록적인 면에 집중하여, KBO에서 제공하는 야구 관련 data 를 가지고 선수들의 stat 을 보관하고 user 들로 하여금 언제든 찾아볼 수 있는 야구 기록 application 을 만들고자 한다.

**2. Uniqueness & Usefulness**

실제 야구팬의 입장에서 그동안 야구기록을 찾아볼 수 있는 사이트라고는 KBO홈페이지나 Naver Sports, Daum Sports등이 전부인데, 여기서 제공하는 선수 기록정보는 야구팬의 입맛에 맞지 않는 부분이 많다. 각각 선수 별로 일자 별 stat을 볼 수 있고 누적 stat을 볼 수 있지만, 날짜의 범위가 제한적이고 또한 제공하는 stat의 종류도 많지 않다. 거기에다 프로야구에서 공식적으로 시상하는 부문인 타자 8개 stat (타율,타점,홈런,안타,출루율,장타율,도루,득점)과 투수 6개 stat (승,탈삼진,방어율,승률,세이브,홀드)을 제외한 다른 stat은 랭킹 정보를 제공하지 않으며, 그나마 제공하는 랭킹정보에도 20위 이하의 선수들에 대해서는 랭킹 정보를 제공하지 않는다. 따라서 타율 20위 안에 들지 않는 선수에 대해서 그 선수의 타율이 리그 전체에서 몇 위정도 되는지를 알 수 있는 방법이 없는 것이다. 또한 아쉬운 것은 야구에는 수많은 stat들이 있지만 그 정보들의 랭킹 만으로는 원론적인 질문인 “그래서 프로야구 선수 중에 제일 잘 치는 타자가 누구야?” 와 같은 질문에 대답할 수 없다는 것이다. 예를 들어 타율 1위인데 홈런은 꼴찌인 타자와 홈런은 1위지만 타율은 꼴찌인 타자 중에 누가 더 잘치는 타자인지 언뜻 파악하기가 쉽지 않다는 것이다. 따라서 우리 조의 application에서는 이런 점들을 보완하여 전문적인 야구팬들을 위한 프로야구 application을 만들고자 한다.

**3. Functionality**

우리조의 application 이 제공하는 기능은 크게 4가지 이다. 첫째로 유저가 원하는 타자 및 투수의 기록을 보여준다. 유저로부터 원하는 선수의 이름을 받아서 DB 안에서검색을 하고 특정 선수의 모든 기록 stat을 가져와 table의 형태로 보여준다. 이 때 동명이인이 있을 경우 두 선수의 정보를 한 페이지에 따로따로 table의 형태로 보여준다. 여기서 제공하는 정보에는 웹에서 crawling해온 타율, 타점, 홈런, 방어율, 승수, 탈삼진 등의 기본 stat 뿐 아니라좀 더 전문적인 OPS, WHIP, 피삼진, 피안타율, 피출루율, 순수장타율 등의 stat들도 포함하고 또한 야구 경기 기록 이외에 선수의 경력, 입단년도, 연봉 등의 경기 외적 정보 역시 포함한다. 또한 그날까지의 누적 stat만 제공하는 일반적인 다른 기록 application과는 달리 우리조의 application은 최근 5경기에 대해서 그 경기까지의 누적 stat도 제공하고, 최근 10경기에 대해서 그날 경기에 대한 daily stat 역시 제공하며모든 stat들을 한번에 볼 수 있는 유저 친화적인 interface를 제공한다. 두 번째로 부문별 순위에서의 선수 검색 기능을 제공한다. 순위를 보기 원하는 숫자를 입력한 뒤 순위보기를 클릭하면 타자는 타자 table에 투수는 투수 table에 입력한 숫자 값만큼 순위에 해당하는 선수들의 이름이 보여진다.예를 들어 10을 입력하면 투수의 모든 stat별로 10위까지 출력되고, 타자 역시 모든 stat별로 10위까지 출력이 된다. 숫자를 입력하지 않고 순위보기를 클릭하였을 경우 default값인 10위까지 stat별로 출력이 된다. 그리고 우리조의 application은 특정 선수에 대한 순위 정보도 제공한다. 순위를 알아보고 싶은 선수의 이름을 입력한 뒤 순위보기를 클릭하면 각각 stat별로 10위까지의 선수들이 출력된 후 그 아래에 입력한 특정선수의 순위정보를 같이 출력해준다. 세 번째로 팀 정보를 보여준다. 팀 순위 보기 버튼을 클릭하면 프로야구 9개의 팀 별로 모든 팀 stat과 순위 정보가 출력된다. 원하는 팀 stat 별로 내림차순으로 볼 수 있다. 네 번째로 타석에서 타자와 투수의 대결을 예측하여 유저에게 알려준다.유저가 가상 대결 결과를 알고자 하는 투수와 타자의 이름을 입력하면 우리가 보유한 data를 통해입력한 투수와 타자의 대결을 예측하여 그 결과를 확률로서 유저에게 보여준다. 그 외 추가적으로 선수의 daily info를 삭제할 수 있는 deletion기능과 언제든 웹 crawling을 할 수 있는 기능을 제공하는데 이는 추후에 관리자만이 실행할 수 있는 기능으로 변경할 예정이다. 이상으로 우리조의 application은 위의 4가지 기능을 제공하며 야구 팬들중 특히 전문적인 자료를 원하는 팬들에게 유용한 application 이 될 것이다.

**4. Data/Development Environment**

Data: 우리조의 data는 전부 KBO 프로야구 2014 시즌 실제 data이다. 현재까지 진행되는 모든 경기의 투수, 타자 누적 stat과 경기 별 stat을 data로 사용한다. 이를 위하여 KBO 홈페이지에서 실제 data들을 web crawling해온다.

Development Environment: 웹페이지와 관련된 부분은 전부 HTML과 CSS, PHP를 통해 작업하였고, data base는<http://jsryu21.bacchus.snucse.org:8080/> 서버의 mysql database를 사용하였다. 이외에 DB 쿼리문을 이용하거나 web crawling 작업, DB에 접속하는 부분은 전부 Python을 사용하여 작업하였다. 또한 우리의 코드는 전부 putty를 통하여 서버에서 코딩하였다.

기본적인 프로젝트 환경은 apache, php, mysql, python, crontab 이 돌아가야 합니다.

서버는 Ubuntu 12.04.4 LTS 입니다.

apache 의 config 와 php config 수정이 필요할 수도 있어 첨부합니다.

apache config 인 my.conf 는 /etc/apache 에 php config 인 php.ini 는 /etc/php5/cli 에 넣어주시고 각 service restart 해주시면 됩니다.

첨부한 dbproject 파일을 열면 crawler 폴더와 www 폴더가 있습니다. crawler 폴더에 있는 스크립트는 서버에 로그인한 유저의 home 폴더에 넣어주세요.

additional\_infos\_calculator.py, game\_crawler.py, match\_predictor.py, match\_result\_creator.py, player\_crawler.py, clean\_daily.py, team\_crawler.py 에서 db 를 사용하기 때문에 소스의 다음과 같은 부분들을 설정하신 세팅에 맞게 수정해주셔야 합니다.

db = connect(db="dbproject", user="root", passwd="asdf1234", use\_unicode=True, charset="utf8")

서버에 로그인한 유저의 home 폴더에서 crontab -e 명령을 치셔서 문서 마지막에 다음과 같이 추가해주세요(단 경로는 알맞게 바꿔주셔야 합니다.)

0 2,10,18 \* \* \* /home/jsryu21/crawler.py

www 폴더는 apache 홈폴더에 넣어주세요(저희 서버 환경에서는 /var/www)

**5. DB Relation Design**

**6. Relation Schema Of the Database**

**7. Data Crawling / Management**

1. **team\_crawler**
   1. **각각의 team 은 숫자로 구별되는데, 이 숫자는 2014년 프로야구 개막이후 최초로 순위가 1위에서 9위까지 나뉘었을 때를 기준으로 산정했습니다.(2014년 4월 15일)**
      1. **team table 에 team 정보가 없을 때 Insert 쿼리를 이용해서 추가해줍니다.**
   2. **team 의 매 경기 기록이 kbo 에서는 두군데에서 나눠서 제공하는데 이 두 부분에서 각각 보여주지 않는 기록들이 있습니다. 따라서 우리 Application 에서는 한번에 보여주기 위해서 두 부분을 모두 crawling 하여 teaminfo table 에 넣었습니다.**
      1. **http://www.koreabaseball.com/Record/TeamRecord.aspx**
      2. **http://www.koreabaseball.com/TeamRank/TeamRank.aspx?searchDate=2014-04-15 // 경기 날짜**
      3. **teaminfo table 에 team 의 경기 기록이 없을 때 Insert 쿼리를 이용해서 추가해주고, 경기 기록이 있을 때는 Update 쿼리를 이용해서 두 기록을 합쳐줍니다.**
2. **player\_crawler**
   1. **모든 선수들을 모아둔 페이지 parsing 을 통해서 투수와 타자 url 을 알냅니다.**
      1. **http://www.koreabaseball.com/Record/PlayerSearch.aspx**
   2. **각 선수들의 url 에서는 선수의 기본정보와 누적 Stat, 매경기 Stat 들이 있습니다.**
      1. **선수 기본정보중 변하지 않는 정보는 player table 에 변할 수 있는 정보는 playerinfo table 에 넣어줍니다.**
      2. **player table 에 없는 player 는 insert 해줍니다.**
      3. **playerinfo table 은 변경된 정보가 있으면 record 를 update 하지 않고 변경된 정보를 새롭게 insert 하여 과거 정보를 잃어버리지 않도록 합니다.**
   3. **각 선수들은 최근 경기까지의 누적 Stat 과 매경기 Stat 들을 따로 가지고 있습니다.**
      1. **투수의 경우 누적 Stat 은 pitcher table 에 매경기 Stat 들은 daily\_pitcher table 에 보관합니다.**
      2. **pitcher table 에 선수의 최근 경기까지의 누적 Stat 이 없을 때 insert 를 해줍니다. daily\_pitcher 는 선수의 경기에 대한 Stat 이 있는 지 모든 경기를 체크하여 없을 때 insert 해줍니다.**
      3. **타자의 경우 누적 Stat 은 batter table 에 매경기 Stat 들은 daily\_batter table 에 보관합니다.**
      4. **batter table 에 선수의 최근 경기까지의 누적 Stat 이 없을 때 insert 를 해줍니다. daily\_batter 는 선수의 경기에 대한 Stat 이 있는 지 모든 경기를 체크하여 없을 때 insert 해줍니다.**
3. **game\_crawler**
   1. **BoxScore 에서 타자 타격 기록과 투수 피칭 기록을 크롤링합니다.**
      1. **해당 경기의 타자 타격 기록이 없을 때 Insert 합니다.**
      2. **해당 경기의 투수 피칭 기록이 없을 때 Insert 합니다.**
4. **clean\_daily**
   1. **프로젝트를 진행하는 도중 match\_result 를 만들던 도중 daily\_pitcher 의 기록이 홈페이지와 다른 경우가 생기는 것을 발견했습니다. 크롤러가 잘못 동작했다기보다는 최근 경기의 기록에 대해 수정이 가끔 일어난다는 것을 발견했고, 크롤러가 오동작했을 때 daily\_pitcher 와 daily\_batter table 의 기록을 날리고 다시 크롤링하기 위해서 만들어졌습니다.**
   2. **실행하면 daily\_pitcher table, daily\_batter table 의 모든 record 를 delete 합니다.**
5. **Management**
   1. **Linux 서버어서 프로젝트를 진행했기 때문에 crontab 이라는 서비스를 이용해서 주기적으로 crawling 을 돌렸습니다.**
   2. **crontabl -l**
      1. **# m h dom mon dow command**
      2. **0 2,10,18 \* \* \* /home/jsryu21/crawler.py**
      3. **매일 오전 2시, 오전 10시, 오후 6시에 크롤링합니다.**
   3. **매번 크롤링할 때 game, player, game 에 대한 정보도 crawling 하지만 이후 설명할 additional\_infos\_calculator(추가 stat 계산), match\_result\_creator(Simulation 을 위한 대결 데이터 생성), classifier\_creator(대결 데이터로 기계 학습 모델 학습) 까지 한꺼번에 수행합니다.**

**8. Additional Information Extraction**

**additional\_info\_calculator**

**채경아 너가 써줘야 할 부분인것 같아**

**9. Player Information Retrieve**

**이 부분 거의 길현이가 다 작업했는데 내가 한 부분은 거의 소스 정리 수준이라**

**10. Batter VS Pitcher Simulation**

1. **타자와 투수의 이름을 입력하면 승부를 예측해주는 시스템입니다.**
   1. **시즌 개막 이후 모든 게임 정보로부터 타자와 투수의 승부를 추출하여 데이터로 사용합니다.**
   2. **5월 29일 기준으로 16505개의 데이터를 구하였습니다.**
   3. **6개의 기계학습 모델을 학습시켰습니다.**
   4. **타자와 투수의 최근까지의 누적 Stat 을 계산 후 이를 모델에 넣어 데이터를 구성합니다.**
   5. **각 모델의 예측 값에 가중치를 두어 결과를 해석하도록 합니다.**
2. **match\_result\_creator**
   1. **game\_crawler 에서 크롤링한 경기의 타자 자격 기록과 투수 피칭 기록을 적절하게 조합하면 어떤 타자와 어떤 투수가 대결을 펼쳐 어떤 결과가 나왔는지를 얻을 수 있습니다.**
   2. **game\_crawler 에서 크롤링한 대결의 결과는 한글로 제공되는데 이를 숫자로 변환하기 위해 다음과 같은 mapping 을 사용했습니다. 좀 더 예측의 정확도를 높이기 위해 같은 종류의 결과는 한데 묶었습니다.**
      1. **번 : 1, 희번 : 2, 희비 : 3, 희실 : 4, 희선 : 5, 비 : 6, 안 : 7, 2 : 8, 3 : 9, 홈 : 10, 땅 : 11, 병 : 12, 직 : 13, 파 : 14, 실 : 15, 삼진 : 16, 야선 17, 사사구 : 18**
   3. **이 때 game\_crawler 에서 크롤링한 타자와 투수는 player\_id 가 아닌 \_name 으로만 제공되었기 때문에 동명이인 처리 때문에 그 날 타격 또는 피칭 기록과 daily\_batter, daily\_pitcher 기록을 비교하여 name 을 player\_id 로 변환할 수 있었습니다.**
   4. **batter\_player\_id 와 pitcher\_player\_id 를 구하면 각 선수의 오늘까지의 daily\_batter, daily\_pitcher 기록을 누적합니다.**
   5. **위에서 누적한 타자 기록, 누적한 투수 기록, 대결 결과로 구성된 데이터를 파일 이름을 날짜로 구분하여 match\_results 라는 폴더 아래 보관합니다.**
3. **classifier\_creator**
   1. **대결 결과들이 Discrete 하기 때문에 Classification 학습 모델을 사용합니다.**
   2. **match\_result\_creator 에서 만든 match\_results 폴더 아래의 데이터를 읽어 학습시킵니다.**
   3. **기계 학습 모델은 다음과 같이 사용했습니다.**
      1. **K-nearnest neighbors**
      2. **Support vector machine**
      3. **Stochastic gradient descent**
      4. **Decision Tree**
      5. **Random Forest Ensemble**
      6. **Naïve Bayes**
4. **match\_predictor**
   1. **타자 이름과 투수 이름을 입력하면 동명이인에 대한 처리를 웹에서 해줍니다.**
   2. **선택한 타자와 투수에 대해서 대결을 학습했던 모델에서 예측 결과를 얻어냅니다.**
   3. **예측 결과값을 다시 사람이 해석할 수 있도록 remapping 해줘야 합니다.**
      1. **1:"번트", 2:"희생번트", 3:"희생플라이", 4:"희생실책", 5:"희생선택"** **, 6:"플라이", 7:"안타", 8:"2루타", 9:"3루타", 10:"홈런", 11:"땅볼", 12:"병살"** **, 13:"직선타", 14:"파울아웃", 15:"실책", 16:"삼진", 17:"야수선택", 18:"사사구"**
   4. **각 모델의 예측 값에 가중치를 두어 결과 해석합니다. 각 모델의 정확도가 서로 다르기 때문입니다. 각 모델의 Precision 은 다음과 같습니다.**
      1. **K-nearnest neighbors : 0.7406**
      2. **Support vector machine : 0.2766**
      3. **Stochastic gradient descent : 0.2939**
      4. **Decision Tree : 0.5391**
      5. **Random Forest Ensemble : 0.2641**
      6. **Naïve Bayes : 0.5667**

**11. Work Division**

**12. Final Thoughts**

**유재성 : 최근에 학교 수업 숙제 이외의 코딩을 할 기회가 적었는데, 내가 만들어보고 싶었던 Application 을 직접 proposal 하고, 구현할 수 있는 기회여서 더욱 재미있었고 능동적으로 참여할 수 있었습니다. 구현하면서 python 이라는 언어에도 많이 친숙해진 것 같고, 이정도 크기의 Relation Design DB 를 설계해볼 수 있어서 좋은 경험했습니다. 그리고 같이 프로젝트를 진행하면서 수고해준 길현이와 채경이와도 손발이 잘 맞아서 다행이고 이번 기회로 친해질 수 있어서 좋았습니다. Final Demo 준비하면서 Web 부분 좀 더 다듬고 기능을 추가해서 직접 도메인에 올려서 사람들에게 직접 서비스를 제공해보자고 했는데 많은 기대가 됩니다.**