**W205 Final Project: Cricket Fielding Statistics**

Chula Watugala

**Introduction**

Description

Cricket is a sport rich with statistics – probably one of the most numbers-focused games around. However, there is a glaring lack of fielding statistics in the game – something that a similar sport like Baseball has handled by the use of video analysis after the game and error statistics.

No such setup exists in cricket – mostly because there is the perception that fielding is too subjective to evaluate. Compared to batting - where it's about getting runs and you can get a simple batting average measure on how many runs you get every time you bat, or bowling -where you can measure how many runs you give away per out, for fielding there's no clear metric to evaluate players. Because of this there are a lot of myths in the game when it comes to fielding - commentators say "he would've caught that 99 times out of a 100" when some fielder drops a catch. Even though it's just an expression, there is no data on what the actual drop rates of fielders are - or what is an actual acceptable drop rate.

Objectives

The objective of this project is to generate cricket fielding statistics using live text commentary feeds. Fielding events and the players involved will be identified and parsed from the text and the data will be used to create useful fielding statistics. If this allows for a method to value fielding performance accurately, franchises would be able to evaluate player value more accurately.

Fielding events considered:

* Dropped Catches
* Misfields
* Missed Stumpings
* Direct Hits (Run-outs)
* Great Catches
* Catches
* Runs Saved

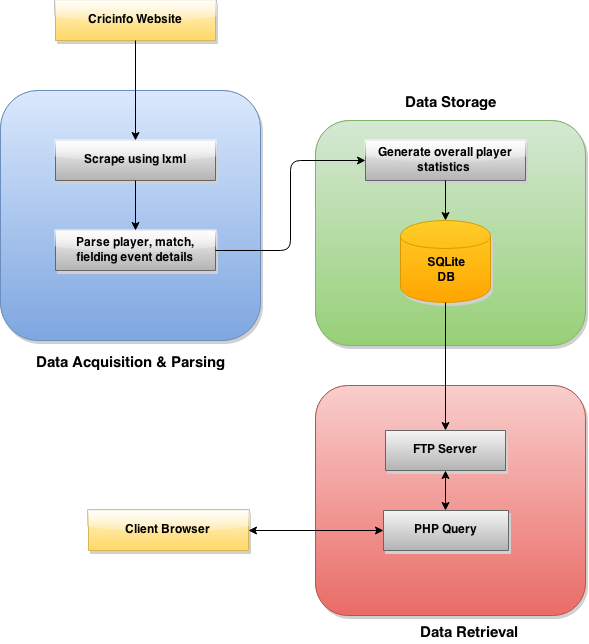
Data Sources

There are multiple sites like [Cricinfo](http://www.espncricinfo.com/magazine/content/story/745379.html), [Cricbuzz](http://www.cricbuzz.com/) and [BBC Cricket](http://www.bbc.com/sport/0/cricket/) that offer live text commentary during games, but only Cricinfo has done it for a long period of time (10+ years) and has had a consistent structure that enables systematic parsing.

Related Work

There have been multiple calls for the necessity of fielding statistics ([here](http://www.espncricinfo.com/blogs/content/story/620188.html), [here](http://www.espncricinfo.com/magazine/content/story/745379.html) and [here](http://www.theroar.com.au/2015/01/12/time-cricket-catch-value-fielding/)), but no concerted effort has been made to do this using an automated and systematic method till now. Therefore this project has the potential to revolutionize how cricket values the fielding aspect of the game.

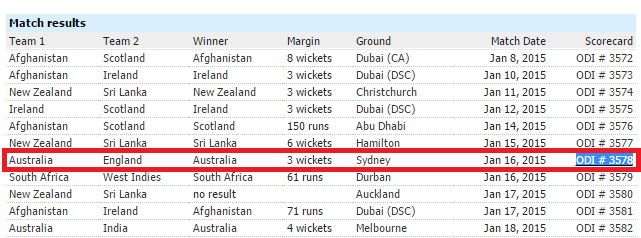
**Solution Architecture**

****

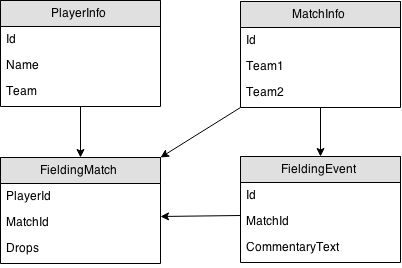
* *Data Acquisition & Parsing:* The lxml package in python is used to scrape each historical match with text commentary and parse the data in xml form.
* *Data Storage:* The generated data is stored in a SQLite database which python has built-in support for.
* *Data Retrieval:* Once the data is generated, it will be hosted on a website using PHP to query the FTP server.

Data Pipelines

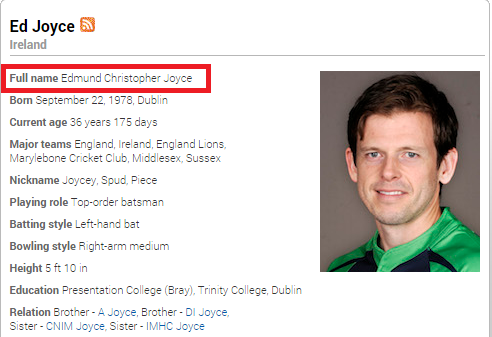
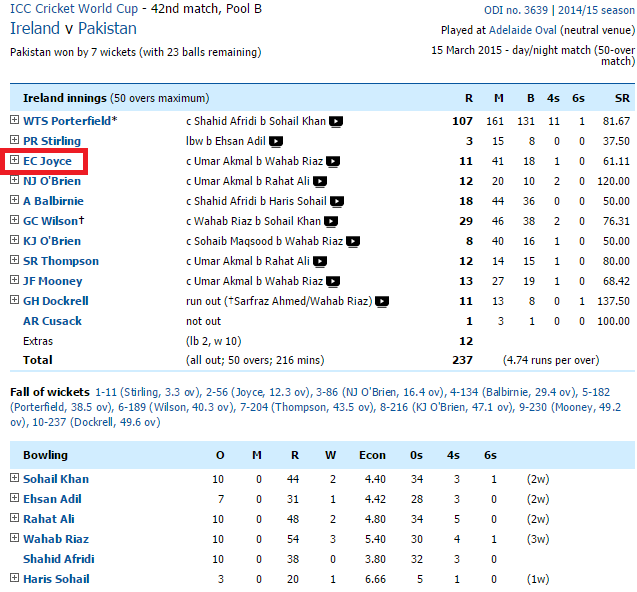
The match results list has links to each match scorecard directly on the page. These links are scraped for match details and stored in a *MatchInfo* table dedicated for match details with a unique ID with the teams involved.

****

Match List

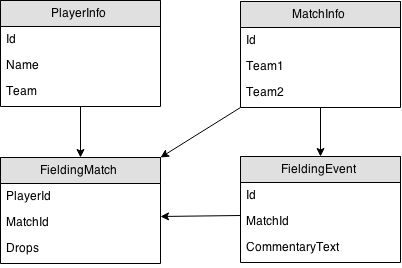


Each match scorecard looks like the example shown below where you get a list of the players involved and their performances with bat and ball. Each player name is linked to a profile page with his unique details, so these player names are scraped to populate a unique player information table.

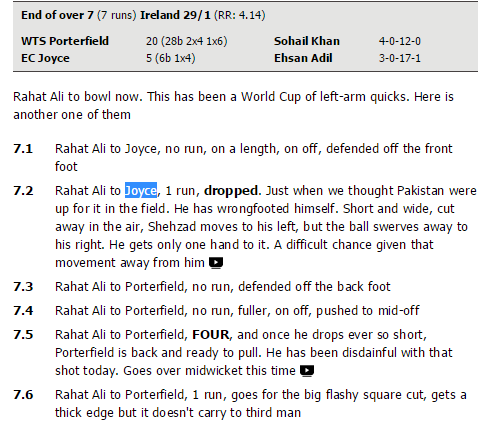
****

Player Profile

Match Scorecard



The scorecard also has the link to the text commentary which is what is used to parse fielding events using keywords. In the highlighted case here Joyce is the batter involved, and using the player information table he can be uniquely identified by matching it with his full name. Each fielding event is stored and then a separate table with cumulative player fielding events is generated for each match which identifies the match and players involved uniquely.

****

Match Text Commentary

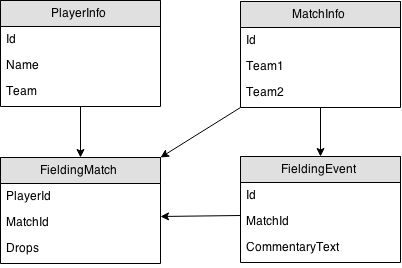
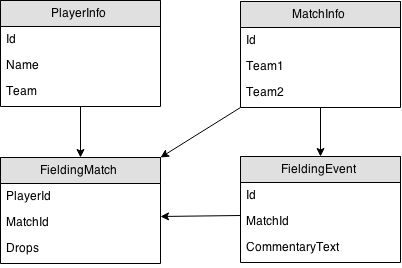
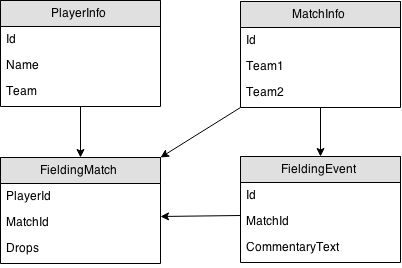
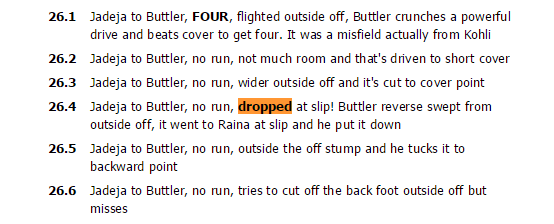
****

Table Schema

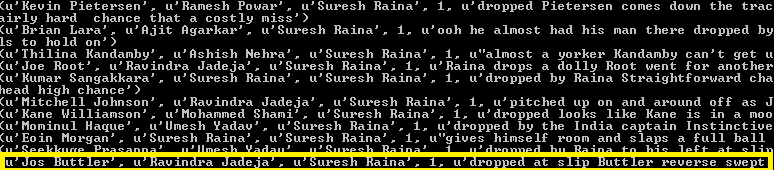
****

Data Acquisition & Storage

In the highlighted dropped event below, the capitalized words are checked in the player table to identify the players involved. Here Buttler and Raina are both in the text, but from the initial part of the text where it says "Jadeja to Buttler" it is known that Buttler is the batter so therefore not the fielder. This way Raina is identified as the fielder who dropped the catch. This is stored in the *fieldingEvent* table, and then an aggregated version with only the fielding event counts grouped by player for the match is stored in the *fieldingMatch* table.



**select batsman, bowler, fielder, droppedCatch, commentary FROM fieldingEventODI where droppedCatch=1 and fielder=?',("Suresh Raina",)**



Data Cleaning

To accurately parse the text commentary data, packages like NLTK were considered but because cricket-specific terms are not recognized and difficult to evaluate it was decided that searching for specific keywords in the text to identify events was a better way forward.

*"bobbles the chance", "has made a meal of it", "sitter", "dolly", "spills", "put down", "dropping the ball", "gets both hands to it but drops it", "drops an easy catch", "fails to take the catch", "dropped", "shelled", "grassed"*

Dropped Catch Key Words

*"dolly on the toes", "dropped right", "dropped with", "dropped just", "dropped it short" , "dropped short", "dropped well in front" , "drops the wrist", "dropped from" , "earlier he was dropped", "dropped his" , "dropped a touch short", "dropped catches" , "dropped behind", "dropped at his feet" , "dropped in", "dropped a bit" , "dropped into", "dropped softly" , "dropped his bat", "dropped catch and" , "dropped it into", "dropped to" , "dropped him earlier", "dropped far too short" , "dropped over", "tough chance", "hard chance", "hard to call it dropped", "hard to call that dropped", "like a football goalkeeper", "desperate effort", "difficult chance", "superb attempt", "good effort", "screaming past", "would have been a very good", "terrific effort", "harsh to blame", "great attempt", "what an effort", "would have been a terrific catch", "would have been a wundercatch", "tough one", "fabulous attempt", "tremendous effort", "difficult one", "would have been a stunner", "would have been a superb", "would have been a mind-blowing", "valiant effort", "harsh to call it", "would have been a classic catch", "would have been a cracker"*

Dropped Catch Invalidating Key Words

The dropped catch fielding event was difficult to accurately evaluate because the keyword "dropped" is used in different ways in cricket. Sometimes it constitutes a dropped catch, but sometimes it might be the batter just "dropping" the ball to the ground or the pitcher "dropping" the ball short (which is a term used to signify that the pitcher has thrown the ball midway through the bases). To distinguish between different types of "dropped" keywords, an additional list of keywords that invalidated the keyword match was utilized - as an example if the text contained "dropped" but the next word was "dropped short", that case is ignored as a fielding non-event. For dropped catches there is the additional complication in that some catches are not expected to be taken because of high difficulty. Specific keywords such as "great effort" or "difficult chance" are used to ignore dropped catches for those cases.

*"direct hit", "accurate with the throw", "throw has beaten him", "hits the stumps direct"*

Direct Hit Key Words

This complication in keywords is not as much an issue for direct hits or great catches - this is reflected in the keywords list length being smaller for those events.

**Implementation**

Technologies Used

*SQL vs NoSQL:*

SQLite was chosen over potential NoSQL solutions like MongoDB or MarkLogic because there were no compelling reasons not to go with a relational database structure. Data is gathered from completed matches in the past and since this is an evaluation of a player’s fielding ability, real-time data is not necessary. Fielding event types do not change so it is possible to start with exhaustive table schema that does not require modification in the future. All the scraped data size amounts to around 150MB which is sizeable but not massive - it also grows steadily since the number of matches played in a year is consistent and does not change significantly.

*Local vs EC2:*

In terms of running the parsing scripts – running locally made most sense instead of using EC2 for more servers since the whole scraping running time to parse scorecard details and text commentary is around 10 hours. After a complete run is completed only newer matches would need be scraped – this would not take more than 5 minutes at a time when run locally.

*PHP vs Other:*

PHP, HTML and CSS were used to display the results of the project in accessible web form. PHP has built-in support to connect to SQLite databases, and HTML/CSS is adequate web design tools for the purposes of this project.

Description of Scripts

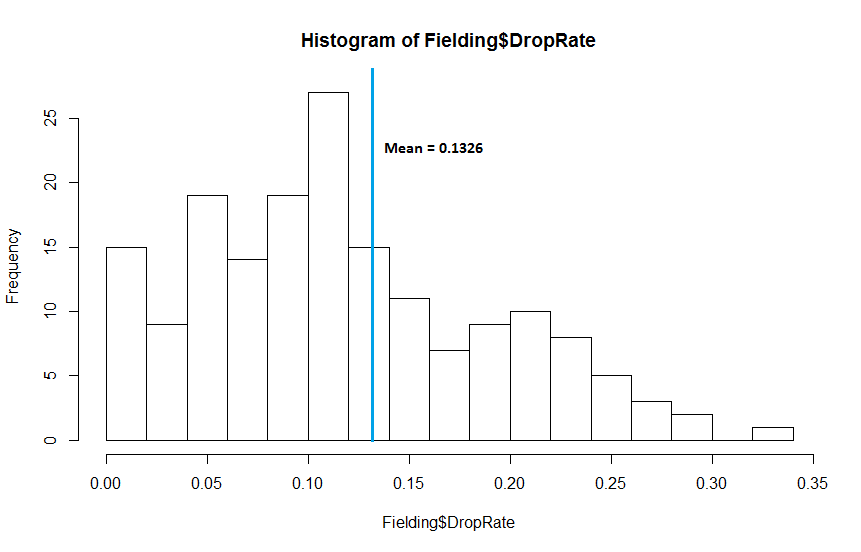
* *createTables.py*: Creates required tables in SQLite which are used in all scripts of project
* *dumpMatchInfo.py*: Scrapes match results lists for match details on teams involved and scorecard links
* *scrapeScorecard.py*: Scrapes match scorecard for batters and bowlers involved in relevant match
* *scrapeFielding.py*: Scrapes text commentary feed of match and parses fielding events and players involved – matching them with existing player table
* *dumpFieldingCareer.py*: Aggregates fielding event data by match over player careers for meaningful statistics
* *cricStats.db*: SQLite database of generated data
* *cricStats.php*: PHP webpage for appropriate display of results
* *style.css*: Styling for webpage

**Results**

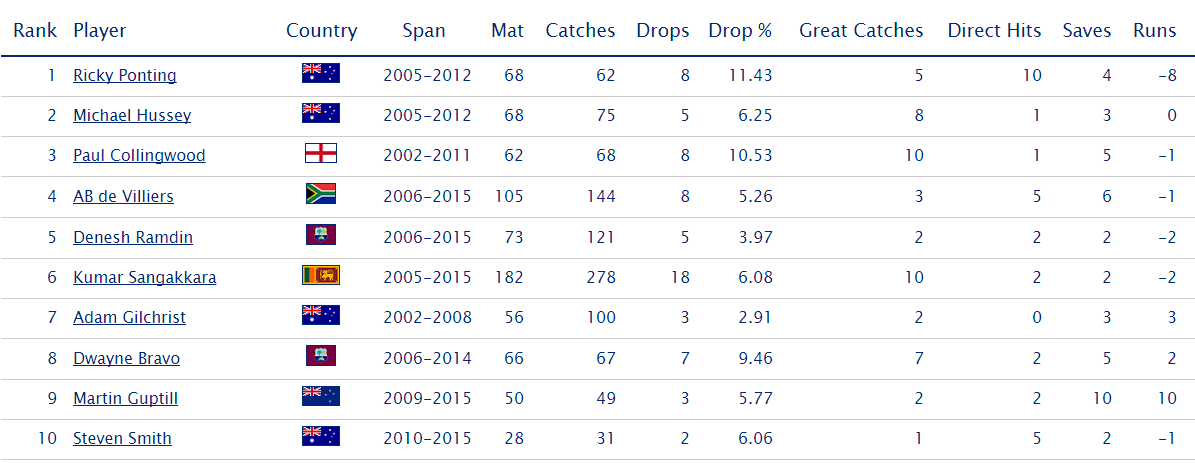
Metrics

Once fielding events are parsed for all matches with the fielders identified, it is possible to generate meaningful statistics to enable comparison between players. One revealing metric is drop rate - the percentage chance that a fielder would drop a catch that he was supposed to have taken.

*Drop % =*



The histogram above shows that the mean drop rate is 13.26% with the graph showing positive skew – indicating that a majority of players perform around or better than the mean drop rate, with a minority of poor fielders with higher drop rates. A fielder with a drop rate less than 10% can be considered to be above average just based on catches.



The table above displays the top fielders rated based on different fielding attributes. Almost all of their drop rates are below 10%, with those that have higher rates like Ricky Ponting and Paul Collingwood having other positive attributes such as high direct hit rates or high great catch rates. Ricky Ponting, Paul Collingwood and AB de Villiers are all considered great fielders – this provides validation to the data and suggests that it is on the right track.

Challenges

The most obvious challenge with this method of parsing fielding events is that it is impossible to guarantee 100% accuracy. Sometimes there are more than 2 players involved in the text and it would be impossible to identify who the correct fielder is without manual observation. Misattribution becomes an issue that is hard to avoid. Even so, based on manual comparisons of a sample of matches, the parsing method has a 90%+ accuracy.

Another issue is that this setup is relatively inflexible and relies on *cricinfo*. If the site changes its format, the parsing infrastructure would also require modification. This also points to long-term viability issues.

Scalability

The current setup focuses on One-day International (ODI) cricket which is one of three forms of the game. The infrastructure used here can be mimicked for Test and Twenty20 cricket with minor modifications – allowing for a three-fold increase in generated data rapidly.

**Conclusion**

Analysis of Result

The project was successful in achieving the stated goal of generating cricket fielding statistics. Although perfect accuracy proved impossible to guarantee, the results were still useful to a great degree and could be used to evaluate player ability as it is – especially considering there are no other current measures in the sport.

Improvements or Alternatives

There are no obvious possible improvements that would solve misattribution issues without manual intervention – so the only plausible better alternative would be to implement a video analysis infrastructure in the game.

Future Iterations

Even though this project successfully generated fielding statistics such as drop rate, it is still difficult to quantify how much value fielding brings to the game. If a player has a 50% drop rate while averaging 40 runs a game with the bat is he better for the team than someone with a 10% drop rate but averages just 30?

To value batting, bowling and fielding impact more accurately, a future iteration could use a "Win Percentage Added” (**WPA**) measure.

For example - in the table to the left a particular team's batting performance is displayed. An over is 6 pitches - and this match has 20 overs. The score column shows that at the 11th over the team was 82/3 - which means they had 82 runs with 3 outs. After the 12th over, the score is 87/4, which means the batting team got 5 runs, but also lost a batter.

It is possible to parse these over-score tables for all historical matches and look for similar scores at the 11th and 12th overs historically. This allows for win odds for the batting team to be generated at each stage of the match.

A possible scenario is that the batting team has 55% win odds after the 11th over, but after the loss of a wicket it goes down to 50% after the 12th. Ignoring the runs scored, if that out was caused by a direct hit from a fielder, he is attributed 5% in win percentage added. This method allows for more accurate measurement of player impact – in fielding as well as batting and bowling.