IndicWiki Summer Internship - Cricket Players

Domain

Team

Data Collection
- Sources/ Sites
- Tools used for Data collection
- Images
- Data Storing

Data Cleaning
- Cases taken care of
- Data Merging

Version control

Sample article
- Sections

Jinja template creation
- Edge cases
- Categories, References
- Infobox

Translation/ Transliteration
- Libraries
- Common Issues while translation, transliteration

XML Generation

Quality Review

Potential Future Enhancements

Github Repository Structure Details

Cricket Players

The aim of this domain is to create a large number of articles (about 10,000) about notable cricket players across the world. This domain has potential because of the interest and passion for cricket and cricketers in our country, and Telugu-speaking states are no exception.
Hence, we are generating these data-rich articles in telugu for about 10,000 notable players, and uploading them to wikipedia, so that people who can read only in their native language (here, telugu) can benefit from this information. We programmed in python throughout this IndicWiki Project for data scraping, merging, cleaning etc. and took help of libraries for obtaining translated and transliterated data in desired fashion.

Team

This team working on the 'Cricket Players' domain of IndicWiki batch-3 comprises 4 members. Their details are given below.

- **Gokul Vamsi Thota** - gokul.vamsi@research.iiit.ac.in
  Worked primarily on sections - professional life (early career, domestic and international career), records (automating and generating translated versions), categories.
- **Hrudai Koda** - hrudai.k18@iiits.in
  Worked primarily on sections - personal life, statistical analysis.
- **Vasireddy Komal Kumar** - komalkumar.v18@iiits.in
  Worked primarily on sections - records, test_records, ODI_records, T20I_records, awards, references (rendering).
- **Sowmya Varakala -** vsvarakalasowmya@gmail.com
  Worked primarily on sections - infobox, awards (transliteration), overview.

Data Collection

Sources/ Sites

- ESPN cricinfo
  - Format of data available - Tables, information for few attributes under corresponding headings (for example - teams names as a list under Teams heading etc..)
  - Tools used - BeautifulSoup, Selenium, pandas library
  - Attributes found - Every attribute except for awards, jersey numbers and images.
- Cricketer Life
  - Format of data available - Player names and their jersey numbers
  - Tools used - BeautifulSoup, pandas library
  - Attributes found - Jersey Number
- Kaggle
  - Format of data available - Column of a dataset
  - Tools used - pandas library

    - Attributes found - Awards
- Wikipedia (with help of Wikidata)
    - Format of data available - As a string value inside infobox of corresponding player's article
    - Tools used - BeautifulSoup, pandas library, wikipedia library
    - Attributes found - Image link

Tools used for Data collection

- BeautifulSoup
    - Webpage content is available merely as strings, and hence not helpful for dynamic actions like selecting options on a drop down etc. → Selenium library was used for such cases.
    - In some cases links were not in the exact same format for every cricket player → Additional handling and conditional checks were needed in code, for such cases.
    - Speed was an issue as web page source code extraction incorporating sleep statements (for avoiding consecutive requests) consumed a lot of time → We divided the execution among the team and ran multiple duplicates based on number of team members in countries
- Selenium
    - It was very slow, even in comparison with BeautifulSoup, for extracting content from dynamic web pages →We divided the execution among the team and ran multiple duplicates based on number of team members in countries
    - It gave errors in very frequent intervals regarding web driver issue, because of which dynamic actions like selecting an option of dropdown didn't work as expected in most cases → Stats for which these issues were faced, were scraped again in json format from a different webpage by sending requests for each player
- Wikipedia Query Service
    - There were no specific issues in using this tool, as SPARQL language was fairly intuitive and helped in easy collection of english wikipedia article links for each player, which in-turn helped to obtain image links from the infobox of that player's article.

Images

- English Wikipedia Article links for corresponding player (with help of Wikipedia Query Service and Wikidata)
    - We tried to use images in wikidata directly, that can be easily extracted with SPARQL (Wikipedia Query Service), but there were multiple images in few cases and it was hard to automate and obtain only formal ones → We automated

extracting english wikipedia article links from Wikipedia Query Service and then scraped infobox image links for these articles.

Data Storing

- Format - xlsx, csv and pkl
- Why - We tried to store the key versions of our datasets in all three formats because we were generally updating the dataset via code in the csv file, and we pickle it for more efficient loading and storing, which saves time. We also stored it in xlsx format because it usually has a smaller size than csv format, and is easier to edit on the go. Nevertheless, manual edits in the dataset were mostly done with the help of Google Sheets on either csv or excel files and everything was updated accordingly.

Data Cleaning

Cases taken care of

- Case
  - There were about 20k players when we scraped data, but a very significant number of them had very sparse rows →We filtered out unhelpful attributes with a lot of sparse data, and also eliminated rows with a strict notability check, and also ensured each cricket player has at least 60 non-nan values overall considering all attributes.
- Case
  - Attributes related to BBI (Best Bowling Innings) and BBM (Best Bowling Match) have data stored in an unwanted format (3/20 is stored as Mar-20 etc..) → We made modifications with code to rectify this issue.
- Case
  - Attributes starting with Batting_ should have all floats replaced by integers (except for average) → Were typecasted accordingly.
  - Attributes starting with Bowling_ should have all floats replaced by integers (except for average) → Were typecasted accordingly.
  - Attributes starting with HOME_, AWAY_ or NEUTRAL_ should not have floating point values (except average and strike rate) → Were typecasted accordingly.
  - Jersey numbers shouldn't be floating point values → Were typecasted accordingly.
- Case
  - Batting style should always be "Left Hand bat" or "Right Hand bat" → Other such values were handled with conditions.
  - Player role should be valid (values like "batter, batter" should be handled) → Other such values were handled with conditions.

    - Some awards had undesirable sentences which seemed biased → Such cases were manually handled.

Data Merging

- Primary key - Cricinfo id
- Process followed, Tool used - 'merge' method in pandas library was used for merging datasets with 'Cricinfo_id' as primary key
- Final KB format - csv, xlsx, pkl
- Final KB rows X columns - Dataset in the below link has 10,050 rows and 389 columns. But effective attributes used in article rendering is 255 (multiple versions of few attributes are stored in the same dataset). Also, there are some rows with the same value for full name (article title). In such cases, only the most populated row with a particular title is considered and the rest are discarded (Explained in XML Generation section below). This reduced the count of rows by 97. Hence, the effective rows AND columns count is 9,953 X 255.
- Final KB link - https://github.com/indicwiki-iiit/Cricket-players/tree/main/data
- For the Kaggle dataset we intended to use for awards, we weren't sure about the credibility → We verified credibility of a dataset we obtained for cricket players, with relevant data related to awards.

Version control

- We used github primarily as our version control tool. As we had many files and versions that got updated very frequently, we created a repository on github and made our changes in that. As many times the repository will look rough and incomplete, we first used this for our changes and intended to shift to the IndicWiki repo when all the work is complete.
  Link to the repo we usually worked on:
  https://github.com/HrudaiKoda/Web-Crawling-Players
  Link to the IndicWiki repo (only with finalized contents):
  https://github.com/indicwiki-iiit/Cricket-players

Sample article

- Link - https://github.com/indicwiki-iiit/Cricket-players/blob/main/Cricket%20Players%20-%20Sample%20Article.pdf

Sections

- We took some reference from existing articles on telugu wikipedia to understand which information would be helpful to provide in such an article, and what would be the most

organized way to do the same. Hence, based on the attributes we could scrape, we divided them into meaningful sections to make the article look complete.

- **Overview** → This section contains a summary about the player, described as paragraphs, containing some insight about the player, teams, few records and awards etc.
- **Personal life** → Information regarding personal relationships, details about the date of birth, place of birth, date and place of death (if dead) etc. are displayed here.
- **Career** → The information of this section has been split into three major subsections.
    - **Early Career** → The information regarding when the career of the player has started, and details regarding his debut performances in different formats, are described here.
    - **Domestic and International Career** → Key information about the player regarding his various stats related to his batting, bowling, fielding/wicket-keeping across all formats of cricket in his/her domestic and international career, is displayed here. Details about major trophies and championships in which he/she played are also displayed here.
    - **Statistical Analysis** → The information regarding the performances of the player in cricket grounds of different nations and conditions (Home, away, neutral) are described here.
- **Records** → Information regarding the various miscellaneous records he/she has achieved throughout his/her career.
- **Test Records** → Information regarding the various test records he/she has achieved throughout his/her career.
- **ODI Records** → Information regarding the various ODI records he/she has achieved throughout his/her career.
- **T20 Records** → Information regarding the various T20 records he/she has achieved throughout his/her career.
- **Awards** → Information regarding different awards won by the player in due course of his/her career (not necessarily all are displayed).
- **References** → Collection of various reference links that have been cited in the article - such as cricinfo profile and stats, awards dataset, jersey number source etc.

Jinja template creation

- Link - https://github.com/indicwiki-iiit/Cricket-players/tree/main/final_templates

Edge cases

- For numeric attributes, -1 was placed for stats that didn't have valid values, and hence was handled in template generation implementation.
- Null values, NoneType objects, empty strings, 'nan' values are all completely handled such that information would be displayed only when the content in that cell didn't correspond to any such above mentioned categories.
- Singular - plural values for numeric attributes were also handled in the template itself.

- Consistent spacing and line-breaks are ensured while generating templates.
- For tables, only those rows and columns are displayed for a cricketer if there is at least one non-null value in that row / column.
- There were a lot of instances where we had to be careful with pronouns related to gender, and we had to incorporate appropriate conditional statements and global variables to accommodate this.
- We had to be cautious about attributes available in cricketer infobox, and convert our data accordingly to that format for infobox display.

Categories, References

- We searched through different categories that are available for cricket players in telugu wikipedia, and also took inspiration from a few existing telugu wikipedia articles on cricketers. We then combined these ideas to the data we have, and tried to display those categories which were feasible.
- The different categories which we used are listed below:
    - క్రికెట్
    - క్రీడాకారులు
    - క్రికెట్ క్రీడాకారులు
    - {{birth_year}} జననాలు
    - {{death_year}} మరణాలు (if not alive)
    - జీవిస్తున్న ప్రజలు (if alive)
    - **Male:** {{nationality}} క్రికెట్ క్రీడాకారులు, **Female:** {{nationality}} మహిళా క్రికెట్ క్రీడాకారులు
    - **Male:** {{nationality}} టెస్ట్ క్రికెట్ క్రీడాకారులు, **Female:** {{nationality}} మహిళా టెస్ట్ క్రికెట్ క్రీడాకారులు (if played in tests)
    - **Male:** {{nationality}} వన్డే క్రికెట్ క్రీడాకారులు, **Female:** {{nationality}} మహిళా వన్డే క్రికెట్ క్రీడాకారులు (if played in ODIs)
    - **Male:** {{nationality}} టీ20 క్రికెట్ క్రీడాకారులు, **Female:** {{nationality}} మహిళా టీ20 క్రికెట్ క్రీడాకారులు (if played in T20s)
    - వికెట్ కీపర్లు (if wicket-keeper)
- Cricinfo is a very reliable and official source for providing statistical details regarding cricketers (even wikipedia uses this source). Credibility was verified for awards attributes whose contents were extracted from a dataset on kaggle. Cricketer Life was used for jersey numbers extraction, whose credibility was verified. Image links were extracted from english wikipedia articles itself which guarantees credibility and avoids any copyright issues (not cited though). All the above mentioned sources were cited appropriately when information provided by them was used.

Infobox

- For ‘Cricket Players’ domain we have a default infobox template designed by Wikipedia named as “Infobox cricketer”.
- In the initial stage of the project, we had ample amount of data to be displayed in the infobox.

- We tried to design a new infobox for our domain but we faced many issues creating a new one.
- We switched back to the same infobox template of telugu wikipedia.
- We decided to drop all the attributes initially thought to be kept in infobox and included only a few attributes that existing infobox supports.

Translation/ Transliteration

Libraries

- deep translit - Used
    - It produced abundant spelling errors for a significant number of nouns, whose pronunciation slightly deviated from the original spelling.
    - It was a little slow in producing transliterated output (but still better than other tools used for translation and transliteration).
- google_trans_new - Used
    - It produced errors after processing a particular number of strings each time (roughly for 500-800 players).
    - It sometimes returned the original word in english in a few cases which had to be manually verified again.
    - It was very slow in producing translated output.
- google.transliteration - Used (invoked only in rare cases where deeptranslit produced errors)
    - It produced errors in a lot of instances where there were symbols like hyphen ('-') etc. in string to be transliterated.
    - It converted numbers to telugu while transliterating, which wasn't ideal for readability.
- translators.google - Used (invoked only in cases where google_trans_new produced errors)
    - The output produced by it was not very accurate in comparison with the output produced by google_trans_new (which was the actual google translate output).
- deep_translator - Used (invoked only in cases where both google_trans_new and translators.google produced errors)
    - The output produced by it was not very accurate in comparison with the output produced by google_trans_new (which was the actual google translate output) and translators.google.
- indictrans - Not used
    - It produced errors on installation, and had a large size for a library which assists in translation.
- indic-transliteration - Not used
    - The output produced was far less accurate (in our case) in comparison with google.transliterate and deep translit.
- Excel sheet translation with formula - Not used

- The output produced was far less accurate (in our case) in comparison with other translation libraries, (it produced erratic output for records attributes which was automated later).

Common Issues while translation, transliteration

- Few important attributes such as batting style, player role, trophy names produced erratic outputs at times on translation and transliteration → We hard-coded such cases (key-value pair of english word - telugu representation) and stored in the form of a dictionary, because there were only a few unique ones and manually storing them seemed far more effective and efficient than using any library on these fields.
- Attributes related to debut match details had strings in bizarre format which produced undesirable outputs on using libraries → These were stored as strings, which were converted to a different string representation, and then transliterated, so that the output is in desired format. This transliteration seemed to work well in most cases.
- Frequently recurring names like stat names (“Avg”, “HS”, “Wkts”, “5w”, “Span”) and format names (“T20I”, “List A”) had meaningless translations → These were hard coded into a dictionary which included their relevant translation of english-telugu as key-value pairs.
- There was a lot of overhead due to online libraries for translation and transliteration and hence it consumed a lot of time → We split it into multiple duplicates and executed simultaneously for parallelism.
- There were abbreviations in attributes like player name, relations, team names etc. which produced erratic translations when online libraries were used → These were hard coded into a dictionary which included their relevant translation of english-telugu as key-value pairs. A function was also designed to facilitate these purposes for tokens with all upper-case characters.
- Awards attribute produced bizarre outputs on translation → Transliteration was used.
- Translation and transliteration is below par for records attributes, both with libraries and with excel sheet translation → On automating and writing a script to check unique sentence structures which seemed otherwise random, they were found out to be 240 in number. Hence, these unique english sentence structures were manually translated (after splitting the task among the group), and a template string was generalized for every such record value, and every record was transformed to fit into this template).

  Template: Translated_structure + " జాబితా లో " + prefix + suffix.

  Example: If given record reads "3rd highest batting average in T20s (48.05)", it is transformed to "టి20లలో అత్యధిక బ్యాటింగ్ సగటు జాబితా లో 3 వ స్థానం (48.05)."

XML Generation

- Executing the ‘render.py’ file would generate wikitext for every cricket player whose details are present in the finalized dataset. This wikitext is dumped into a single xml file with the help of file ‘genXML.py’.
- The single XML file ‘cricket_players.xml’ contains wikitext of every cricket player whose details were collected. This xml file is imported to wikipedia to generate the articles for different cricket players.
- It is observed that there were duplicate cricket player names in the dataset which will cause a problem while generating an XML file, as they might overwrite the previous cricket player information. To resolve this issue, among all player entries with a particular full name, we consider only the player entry with most non-null values including all attributes. This seemed most optimal because most entries which have such a duplicate name, weren’t of popular cricketers.
- While creating the XML, pre-defined entitities(< > & ' ") were also taken care of by replacing them with appropriate strings(< > & ' ") respectively. These entities are replaced with the strings mentioned above in all of the Wikitext and title (cricket player’s name).

Quality Review

- We rendered different cricket player articles (about 20) and ensured that all the edge cases are handled properly. These articles corresponded to different categories of cricketers (male, female, famous, not famous (sparse data) etc.), which ensured diversity and more efficient error handling.
- We regularly had review meetings with our mentor (Sai Teja) and enriched the structure and quality of the article, by making corrections based on suggestions.
- We rendered articles pertaining to different categories (as described above) and got them reviewed.
- As suggested by language experts, we made appropriate corrections and added randomization in sentences wherever possible.
- Entire translated/transliterated dataset was sent for review to ensure that there are no mistakes in the transliterated/translated data.
- We rendered many articles and checked for uniformity (spaces between words, paragraph breaks etc.) in the template, and made necessary corrections based on observations.

Potential Future Enhancements

- Across all sections of articles of cricket players, less generic descriptions related to the skills of the player, lifestyle etc. can always be added for making the article more content-rich and readable, to avoid monotonicity.
- More content can be added to the awards section for each player, based on availability of data, as current data doesn't contain all the information related to player's awards in most cases.
- As stats of a player are ever-changing (if the player is still playing cricket), they should be updated from time-to-time to provide updated information.
- More information related to the personal life of a player can be added for individual articles, as there is not much content in that section for the majority of players, based on available data.
- Information corresponding to specific and famous leagues, (such as IPL, Big Bash League) can be added for enhancing the article. (Although availability of such data for automating is an issue).

Github Repository Structure Details

- Detailed description regarding the structure of the github repository - regarding installation, different files and their purpose etc. can be found here.