QSS20: Modern Statistical Computing

Unit 08: Fuzzy/probabilistic matching

Goal for these few sessions

- Exact matching: types of joins
  - Inner joins
  - Outer joins
  - Left joins
  - Right joins
- Basic regex for two purposes:
  1. Clean join fields for exact matching/merges
  2. Clean join fields for fuzzy/probabilistic matching/merges
- **Fuzzy/probabilistic matching and merges**

Goals for today

- Form pset 2 groups
- Recap of regular expressions
- Intro to function guide GitHub Wiki
- Plan ahead for final projects
- Fuzzy/probabilistic matching lecture (brief) and activity

Goals for today

- **Form pset 2 groups**
- Recap of regular expressions
- Intro to function guide GitHub Wiki
- Plan ahead for final projects
- Fuzzy/probabilistic matching lecture (brief) and activity

Groups for pset 2

Partner A	Partner B
Max Konzerowsky	Omario Corral-Williams
Anish Sikhinam	Justin Sapun
Emma Johnson	Rachael Williams
Daniel Xu	Filippo de Min
?	?
?	?
?	?
?	?

Goals for today

- Form pset 2 groups
- **Recap of regular expressions**
- Intro to function guide GitHub Wiki
- Plan ahead for final projects
- Fuzzy/probabilistic matching lecture (brief) and activity

Recap of regex

What do you remember?

Recap of regex

- Tips:
  - Use ‘re.sub()‘ to clean strings
  - If you just want to find all matches, use ‘re.findall()‘ (DataCamp uses this exclusively)
  - To check for ANY matches (think: boolean output), use ‘re.match‘ for a short string or ‘re.search()‘ for a long one
  - Practice with metacharacters and find a regex cheatsheet you like (here’s a good one for beginners)
- Useful commands/metacharacters:

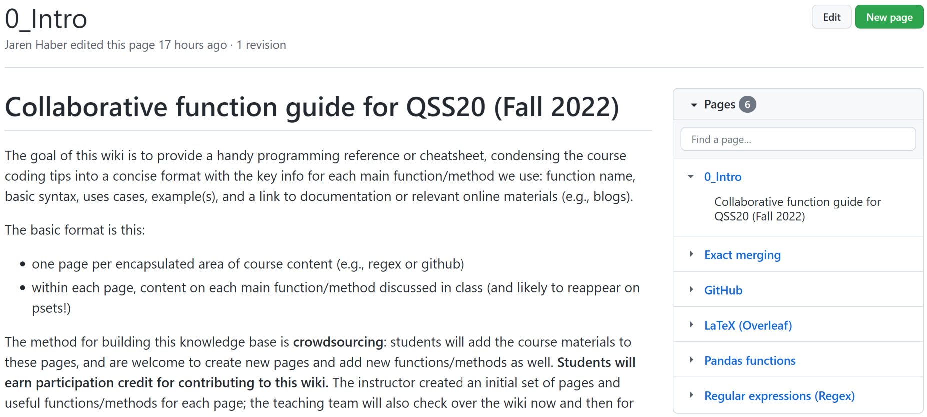
```
re.sub(pattern, replace_this, string) # for substitution
re.findall(pattern, string) # list all matches
re.search(pattern, string) # search whole string
re.match(pattern, string) # search from start
\w | \s | \d | . : chars alphanum, space, numeric, anything
? : previous char/group MAY occur
+ | * : match one or more, any number
() | (?:) : matching, non-matching group
```

Where we are

- Form pset 2 groups
- Recap of regular expressions
- **Intro to function guide GitHub Wiki**
- Plan ahead for final projects
- Fuzzy/probabilistic matching lecture (brief) and activity

Our function dictionary: Course GitHub Wiki

https://github.com/jhaber-zz/QSS20_public/wiki/0_Intro



0_Intro

Jaren Haber edited this page 17 hours ago · 1 revision

Collaborative function guide for QSS20 (Fall 2022)

The goal of this wiki is to provide a handy programming reference or cheatsheet, condensing the course coding tips into a concise format with the key info for each main function/method we use: function name, basic syntax, uses cases, example(s), and a link to documentation or relevant online materials (e.g., blogs).

The basic format is this:

- one page per encapsulated area of course content (e.g., regex or github)
- within each page, content on each main function/method discussed in class (and likely to reappear on psets!)

The method for building this knowledge base is **crowdsourcing**: students will add the course materials to these pages, and are welcome to create new pages and add new functions/methods as well. **Students will earn participation credit for contributing to this wiki.** The instructor created an initial set of pages and useful functions/methods for each page; the teaching team will also check over the wiki now and then for

Our function dictionary: Course GitHub Wiki

Activity

- Head to function Wiki and poke around.
  https://github.com/jhaber-zz/QSS20_public/wiki/
- Pick a function to document! (OK if it's not listed yet in Wiki.) Write it on the whiteboard with your name.
- Take 10 mins to begin documenting your function. (OK to borrow from slides.) If you finish, pick another one!

Where we are

- Form pset 2 groups
- Recap of regular expressions
- Intro to function guide GitHub Wiki
- **Plan ahead for final projects**
- Fuzzy/probabilistic matching lecture (brief) and activity

Get together with your final project group!

Project	**Partner A**	**Partner B**	**Partner C**	**Partner D**
SIP: SIRS	Andrew Cho	Justin Sapun	Anish Sikhinam	
SIP: Medical IDD training	Saige Gitlin	Kayla Hamann	Rachael Williams	
SIP: Medical IDD training	Emma Johnson	Omario Corral-Willians	Max Konzerowsky	
Felony sentencing	Luca D'Ambrosio	Filippo de Min	Nick Romans	Daniel Xu
Felony sentencing	Daniel Céspedes	Giulio Frey	Andy Ilie	

Next milestone for final project

Instructions for completing **final project milestone one** due Friday 10/21 by 11:59 PM:

- Copy over template to your Overleaf account (**please don't edit shared version!**)
- Fill in the memo fields in Overleaf (most of your work right now)
- Submit **one memo per group** on Canvas AND share on Overleaf with jhaber@berkeley.edu

Next milestone for final project

Activity

- Access the template and look over it with your group: https://www.overleaf.com/9461636581djcsgynkwkgk (Link is also on course website under "Final Project → Project Components".)
- Make a group working plan for how/when to get this done.

For groups doing SIP medical student training option, keep in mind these data limitations (though more data is on the way):

- Background form and pre-assessment: **53** completed
- Foundational post assessment: 13 completed
- Intermediate post assessment: 5 completed
- Advanced post assessment: 15 completed
- Community prescriber feedback: 16 of these have been received
- Individual training module evaluations: **302**

Where we are

- Form pset 2 groups
- Recap of regular expressions
- Intro to function guide GitHub Wiki
- Plan ahead for final projects
- **Fuzzy/probabilistic matching lecture (brief) and activity**

Working example: which businesses received PPP loans?

Focal dataset: sample of PPP loans for Winnetka businesses

Business name	NAICScode	City	State	Zip
CLASSIC KIDS, LLC	541921	Winnetka	IL	60093
NORTH SHORE COUNTRY DAY SCHOOL	611110	Winnetka	IL	60093

Other data:

Business name	City	State	Zip
CLASSIC KIDS	Newport Beach	CA	92660
CLASSIC KIDS UPPER WEST	Manhattan	NY	10024
CLASSIC KIDS	Winnetka	IL	60093
CLASSIC KIDS PHOTOGRAPHY	Chicago	IL	60614

What's the role of fuzzy/probabilistic matching?

- **Exact match:** would find no matches in previous example since there's no Classic Kids, LLC in the Yelp data; `pd.merge` fails us
- **Probabilistic match:**
  1. Compares a given pair of records
  2. Using 1+ fields—e.g., business name; zip code; address—what's the probability that the pair is a match?

General workflow for probabilistic matching, regardless of package

1. **Preprocess the relevant fields in the data:** none of these algorithms are magic bullets; each can have significant gains from basic string preprocessing of the relevant fields (e.g., should we remove `LLC`?; how are street addresses formulated)
2. **Decide if/what to "block" or exact match on:** when creating the candidate pairs, what's a *must have* field where if they don't match exactly, you rule out as a candidate pair?
   - **How do you decide this:** fields that are more reliably formatted (e.g., two-digit state)
   - **Main advantages:** potentially reduces false positives; reduces runtime/computational load
3. **If blocking, creating candidate pairs based on blocking variables:** if we blocked on state, for instance, this would leave the two IL businesses as candidate pairs for our focal business
4. **Decide on what fields to match "fuzzily":** these are things like name, address, etc. that might have typos/different spellings. The two components are:
   - How to define similarity: string distance functions
   - What threshold counts as similar enough
5. **Within candidate pairs, look at those fuzzy fields**
6. **Aggregate across fields to decide on "likely match" or "likely not"**

Specific workflow depends on (1) manual versus (2) package

1. In activity code, we'll (1) first do things manually and then (2) use a package
2. Packages in Python:
   - `recordlinkage`: focus of example code
   - **Others:** `fuzzy-matcher`; `sklearn` if we have a small set of "true matches" and want to build a model that predicts matches
3. Packages in R: `fast-link`; `RecordLinkage`

Guide to data and notebooks

Datasets:

- `sd_forfuzzy.csv`: sample of businesses from San Diego tax certificate data used in exact merging activity
- `pploans_forfuzzy.csv`: sample of businesses receiving federal PPP loans

Activities

- **As a class:** https://github.com/jhaber-zz/QSS20_public/blob/main/activities/solutions/06_merging_fuzzy_codeexample.ipynb
- **Then in small groups:** https://github.com/jhaber-zz/QSS20_public/blob/main/activities/06_merging_fuzzy_activity_blank.ipynb