QSS20: Modern Statistical Computing

Session 07: Fuzzy/probabilistic matching

Goal for next 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**

Class today

- **Housekeeping:** many office hours slots open; please stop by if still confused about writing/debugging user-defined functions or any other pset 1 and 2 material! (optional DataCamp `Writing your own functions` in your assignments tab)
- Final project overview and answering questions
- Fuzzy/probabilistic matching: mainly activity rather than slide-based

Class today

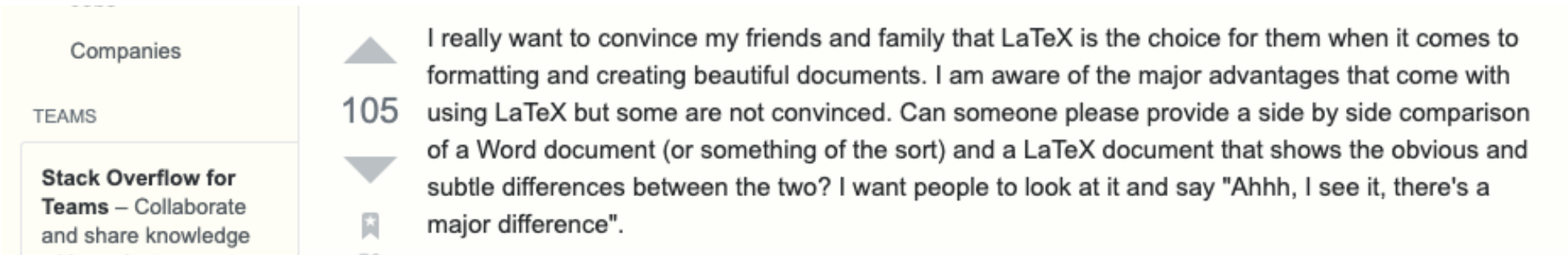
- Housekeeping: many office hours slots open; please stop by if still confused about writing/debugging user-defined functions! (optional DataCamp module in your assignments tab)
- **Final project overview and answering questions**
- Fuzzy/probabilistic matching: mainly activity rather than slide-based

Relevant documents

1. **Guide to final project:** https://docs.google.com/document/d/1zi8cHCmP_5vO6PbmPhKa3vvBJPJe0-fS3eZQgaMqhiU/edit
2. **Form to fill out by Wednesday 02.02 at 11:59 PM EST:** https://forms.gle/6idtfUzb5e81J2oV6
3. **Memo one to submit by Wednesday 02.09 at 11:59 PM EST (one per group):**
   - Copy over template to your Overleaf account; will walk through
   - Edit in there
   - Submit one per group (Canvas for PDF and share editable doc with bjohnson88@gmail.com)

Overview of LaTeX before memo

- LaTeX: typesetting language
- As discussed in sofware setup, can work with locally using things like TexMaker, etc.
- Here, we'll be interacting with it via Overleaf, which is similar to Google docs but for LaTeX and facilitates collaboration/easy(or easier...) troubleshooting of compile errors



Non-exhaustive list of things that can cause compilation errors

- Underscores or certain special characteristics without an "escape" before them– eg:

```
## causes error due to underscore without escape
The file is called: file_here.R
## works
The file is called: file\_here.R
## comments out rest of code after percent symbol
This increased by 5%
## works
This increased by 5\%
```

- Start entering math mode but fail to exit it, e.g.

```
## causes errors
We calculate fraction as $\dfrac{5}{10} and then do...
## works
We calculate fraction as $\dfrac{5}{10}$ and then do
```

“Environments”, or ways to go beyond standard text

- Itemized list

```
\begin{itemize}
\item First item...
\item
\end{itemize}
```

- Numbered list

```
\begin{enumerate}
\item First item...
\item
\end{enumerate}
```

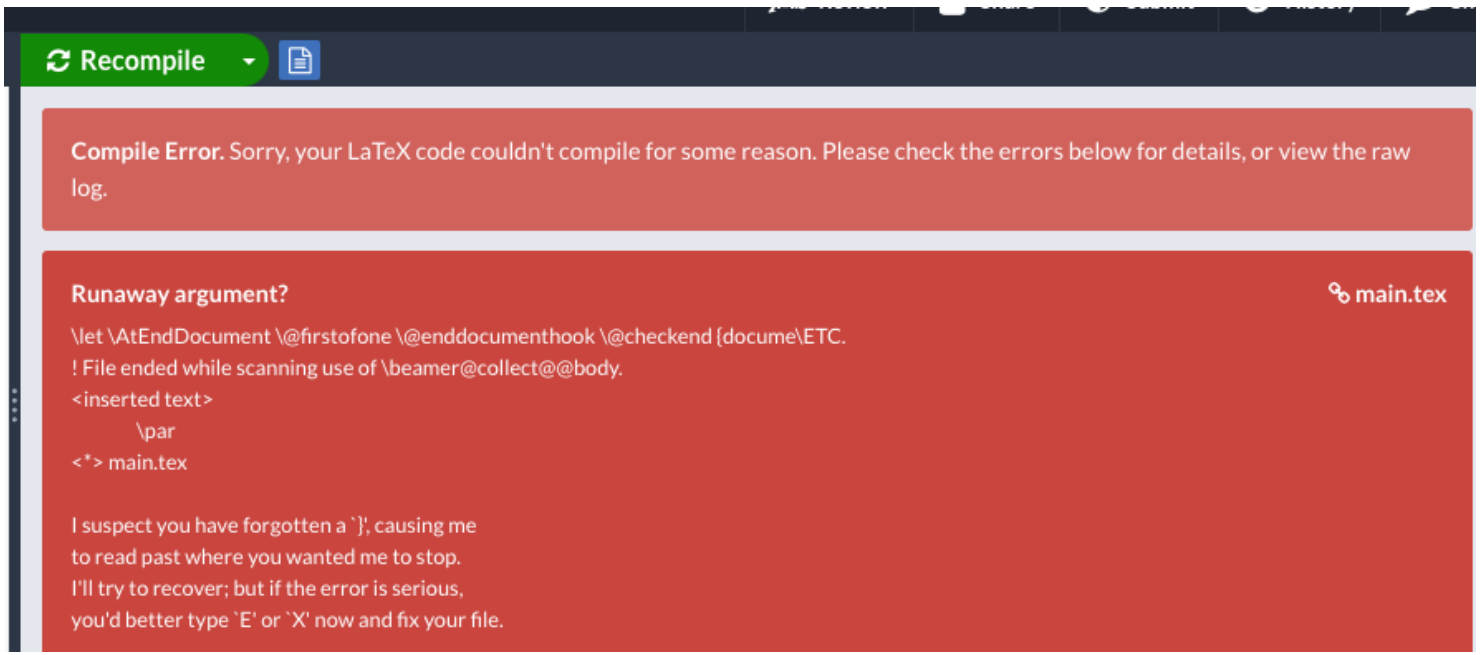
- Figure

```
\begin{figure}
\caption{my caption}
\label{fig:myfig}
\includegraphics[scale = 0.5]{example_graphic.png}
\end{figure}
```

Leads to another set of compilation errors

- Runaway argument or forgotten end group
- Usually means you began an environment but forgot to end it; can happen with long tables, deeply nested lists, etc. where easy to lose track

Example:



Compilation errors

- Common w/ complicated docs
- Ways to address beyond googling: try to recompile relatively frequently since especially on Overleaf, error messages are not always the most informative w.r.t. line numbers

Other useful commands

```
## create a numbered section and give it a label to cross-ref
\section{This is my section outlining disparities}
\label{sec:disparities}

## reference a section in text
In Section \ref{sec:disparities} I discuss...

## reference a table or fig in text
In Table \ref{tab:tabname}, I show why Figure \ref{fig:myfig} shows

## stop a table or figure from going into the next section
## (in addition to stuff at the start of the \begin{table} command
\FloatBarrier
```

Overview of final project memo one (02.09)

https://www.overleaf.com/6332281418zjztwtkkvqjt

Class today

- Housekeeping: many office hours slots open; please stop by if still confused about writing/debugging user-defined functions! (optional DataCamp module in your assignments tab)
- Final project overview and answering questions
- **Fuzzy/probabilistic matching: mainly activity rather than slide-based**

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

1. **As a class:** https://github.com/rebeccajohnson88/qss20_slides_activities/blob/main/activities/w22_activities/solutions/05_merging_fuzzy_codeexample.ipynb
2. **In small groups after:** https://github.com/rebeccajohnson88/qss20_slides_activities/blob/main/activities/w22_activities/05_merging_fuzzy_activity_blank.ipynb
3. **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