CoWrangler: Recommender System for Data-Wrangling Scripts

Bhavya Chopra, **Anna Fariha**, Sumit Gulwani, Austin Z. Henley, Daniel Perelman, Mohammad Raza, Sherry Shi, Danny Simmons, Ashish Tiwari

Microsoft

The University of Utah

Data wrangling is time consuming

emergency_call_description
REINDEER CT & DEAD END; NEW HANOVER; Station 332; 2015-12-10 @ 17:10:52
HAWS AVE; NORRISTOWN; Station STA27; 2015-12-10 @ 14:39:21
BLUEROUTE & RAMP I476 NB TO CHEMICAL RD; PLYMOUTH; ; 2015-12-10 @ 17:35:41

I want to split the values using semicolons. But how do I deal with the third row, which has fewer splits?

```
df[['address', 'township', 'station', 'timestamp']] = \
    df['desc'].str.split('; ', 3)
```

`ValueError`: Columns must be same length as key

My script worked after searching the web! Setting the expand parameter to True lets each missing split occupy a column.

```
df[['address', 'township', 'station', 'timestamp']] = \
    df['desc'].str.split('; ', 3, expand=True)
```

It took me 10 minutes to wrangle just one column. There are 8 more!

How do I know if my script is working properly? *Is my data clean now?*

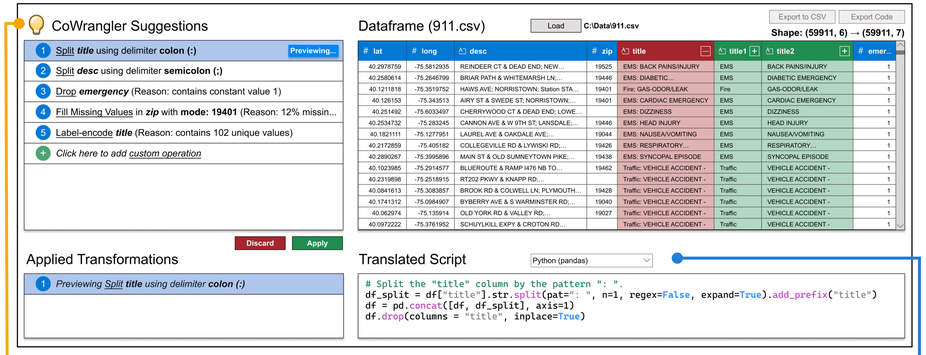
How will I navigate this API jungle? There are so many APIs and parameters!

How do I debug cryptic error messages like ValueError?

Automatic wrangling has several challenges

- How to **pick relevant** wrangling transformations from the **enormous space** of valid transformations?
- What **metric** to use to **rank** them?
- How to **translate** them to **human-readable** scripts?
- How to make the scripts **indistinguishable** from **human-authored** scripts?
- How to make sure the scripts are **performant**?

CoWrangler recommends data-wrangling scripts



Wrangling suggestions with explanations

*Select a suggestion to view **translated script** and a preview of the **transformed data***

Translated scripts for suggestions

*CoWrangler generates **performant** scripts, with **comments** and **meaningful variable names***

CoWrangler enables human-in-the-loop wrangling

Data scientists can customize and interact with CoWrangler suggestions by

- **Editing the scripts** generated by CoWrangler
- Expressing **intent by example** [1]

timestamp	month
29-07-2011 5:10:52 PM	07
17-10-2016 5:29:21 PM	10
18-02-2014 2:39:21 PM	02
04-03-2012 4:47:36 PM	03
09-12-2010 4:56:52 PM	12
12-11-2015 3:39:04 PM	11

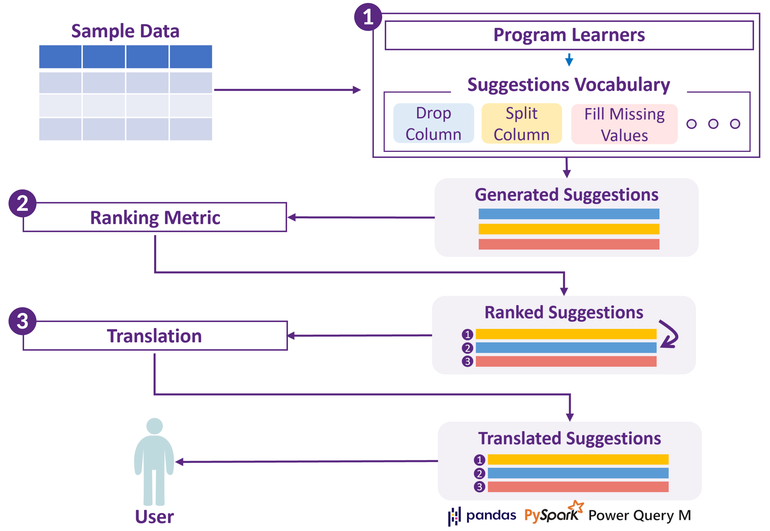
User provides an **example** to extract month

CoWrangler **extrapolates** values for all rows

```
# Derive column "month" from column: "timestamp"
from datetime import datetime

# Transform "timestamp" as per the following examples:
# 29-07-2011 5:10:52 PM -> 07
df.insert(6, "month", df.apply(lambda row : row["timestamp"].strftime("%m"), axis=1))
```

CoWrangler in a nutshell



1. **Predictive Synthesis** to learn & suggest **valid** transformations [2]
2. **Ranking** suggestions based on measurable **data quality improvements**
3. Smart **translation** to generate **performant** scripts, using vector APIs

CoWrangler achieves 53% accuracy

Benchmark: **2248 pandas operations** from **730 Kaggle notebooks**

- CoWrangler's vocabulary **supports 33%** operations
- Suggestions are **accurate in 53.4%** cases

References

[1] S Gulwani. Automating string processing in spreadsheets using input-output examples. POPL 2011
[2] M Raza and S Gulwani. Automated data extraction using predictive program synthesis. AAAI 2017