DATA 3464: Fundamentals of Data Processing

Bash and data cards

Charlotte Curtis

March 12, 2026

Topic overview

- Intro to bash and unix tools
- Data cards

Resources used:

- Data Science at the Command Line
- The Data Cards Playbook

> *Why these topics together? They both seem relevant to assignment 3*

Why bash?

- Python is great, but it's not the only tool. Command line is better to:
  - Move and rename files
  - Take a peek at the first few lines of a giant csv
  - Find and replace text in a bunch of files
  - Fetch data from the web
- Bash and unix tools are more useful than their Windows counterparts
- Git bash lets us use bash on Windows

Okay, but what is it?

- The Bourne Again SHell is a command line interface created in 1989
- A **shell** is a program to execute commands from the user
- The "Bourne shell" (`sh`) from the 1970s was standard on UNIX, bash adds to it
- Default shell for linux, (almost) macOS
- When you run these commands in Jupyter notebook, you're using bash!

```
%pip install -q some_package
```

Basic bash

```
command_name --long_flag -l -o arg1 arg2
```

- Examples of common commands:
  - `ls` to list files in a directory
  - `cd` to change directories
  - `mv` to move or rename files
  - `head` to view the first few lines of a file
  - `grep` to search for text in files
  - `sed` to find and replace text in files
  - `curl` to fetch data from the web

Variables

- Variables are defined with `=` and accessed with `$`
- File arguments are passed as `$1`, `$2`, etc. in bash scripts

```
name="Salvador"
echo "Hello, $name!"
```

- Quotes matter!
  - Double quotes `"` allow **variable expansion**
  - Single quotes `'` treat everything as literal
  - Backticks `` ` `` execute a command and insert its output

Loops

```
for file in **/*.csv; do
    echo "Processing $file"
    # Do something with $file
done
```

- `**/` means "look in all subdirectories"
- `*.csv` is a **wildcard** that matches all files ending in `.csv`
- Wildcard can also be used to match other patterns, e.g. `data_*.csv`
- You can also loop over numbers, lists of literals, etc.

Pipes

```
cat data.csv | grep -o "\d{3}[-\s]?\d{3}-\s]?\d{4}" | wc -l
```

- The pipe `|` feeds the output from one command into the next
- This example:
  - `cat` outputs the contents of `data.csv`
  - `grep` finds regex matches (for what?)
  - `wc -l` counts the number of lines that match

Redirecting input and output

- `>` sends output to a file, overwriting it
- `>>` appends output to a file
- `<` reads input from a file (less)

```
echo "Hello, world!" > greeting.txt
echo "\nNext line" >> greeting.txt
some_program < saved_inputs.txt
```

> *You might also see* `2>`*, which redirects error messages instead of output, or* `&>` *which redirects both*

Where we left off on March 12

Data cards

Enough of bash for now, you'll practice more in the lab...

- Data Cards are a structured way of documenting a dataset
- Some version of data cards are used at Kaggle, Hugging Face, and probably anywhere else you can find data

> *Example: let's fill out a data card for the OKCupid data*

Coming up next

- Basics of signal and image processing
- Labelling data for machine learning