Handy Command-line One-liners for Starting Data Scientists

(from https://medium.com/@rama100/handy-command-line-one-liners-for-starting-data-scientists-81f933241128)

Category	Task	One-Liner	Comments
File exploration	Count the number of lines in a file	*wc -l filename*	
	Show the column names, one in each line, preceded by line numbers (i.e., grab the header row, transpose it and prefix line numbers)*	*head -1 filename | tr '\t' '\n' | nl* *head -1 filename | tr ',' '\n' | nl* *head -1 filename | tr ' ' '\n' | nl*	Helpful when you have numerous columns in a new file and want to get the lay of the land e.g., knowing that "average_selling_price" is column # 39 is useful for some of the examples below.
	Page through the file, showing line numbers	*less filename | nl*	
Row-oriented operations	Show line #4212	*sed '4212q;d' filename*	Very useful when you are trying to load the file into a database and the load fails at line #4212, for instance. Also, this command will conveniently quit after printing the 4212nd line; very considerate if your file has a million lines!
	Show lines with "foo" in any field show lines with "foo" in any field, ignoring case	*grep 'foo' filename* *grep -i 'foo' filename*	
	Show lines with 'foo' in field #18*	*awk -F\t '$18 == "foo" ' filename* *awk -F, '$18 == "foo" ' filename* *awk '$18 == "foo" ' filename*	
	Show rows that have fewer fields than the header row	*awk 'NR==1 {x=NF}; NF < x' filename*	Quick check to see if any rows are incomplete
	Remove lines with "foo" in any field and save the rest into a new file	*sed '/foo/d' filename > newfile*	
	Remove lines with 'foo' in field #18 and save the rest into a new file*	*awk -F\t '$18 != "foo" ' filename > newfile* *awk -F, '$18 != "foo" ' filename > newfile* *awk '$18 != "foo" ' filename > newfile*	
	Remove the first line and save the rest into a new file	*sed '1d' filename > newfile*	Great for stripping a header row before further processing
	Remove the first 8 lines and save the rest into a new file	*sed '1,8d' filename > newfile*	
	Remove line #42 and save the rest into a new file	*sed '42d' filename > newfile*	
	Remove lines 233 to 718 and save the rest into a new file	*sed '233,718d' filename > newfile*	

* The first one-liner is for tab-delimited files, the second for comma-delimited files, and the third for space-delimited files.

	Remove the last line and save the rest into a new file	*sed '$d' filename > newfile*	
	Remove the last 8 lines and save the rest into a new file	*sed -e :a -e '$d;N;2,8ba' -e 'P;D' filename > newfile*	Ugh! Let me know if you know of a better way
	Remove blank lines from the file and save the rest into a new file	*sed '/^$/d' filename > newfile*	
	Remove duplicate lines and save the rest into a new file, preserving the original order	*awk '!seen[$0]++' filename > newfile*	
	Remove duplicate lines and save the rest into a new file, original order may not be preserved	*sort -u filename > newfile*	
	Remove lines with a missing value in field #18 and save the rest into a new file*	*awk -F\t '!$18' filename > newfile* *awk -F, '!$18' filename > newfile* *awk '!$18' filename > newfile*	
Column-oriented operations	Show just col #42*	*cut -f42 filename* *cut -d, -f42 filename* *cut -d' ' -f42 filename*	
	Show the unique values in column #42 with counts*	*cut -f42 filename | sort | uniq -c* *cut -d, -f42 filename | sort | uniq -c* *cut -d' ' -f42 filename | sort | uniq -c*	Useful for understanding a categorical field. A histogram, essentially.
	Remove the 1st field and save the rest into a new file*	*cut -f2- filename > newfile* *cut –d, -f1–18,43- filename > newfile* *cut –d' ' -f1–18,43- filename > newfile*	
	Remove field #42 and save the rest into a new file*	*cut -f1–41,43- filename > newfile* *cut –d, -f1–41,43- filename > newfile* *cut –d' ' -f1–41,43- filename > newfile*	
	Remove fields #19–42 and save the rest into a new file*	*cut -f1–18,43- filename > newfile* *cut –d, -f1–18,43- filename > newfile* *cut –d' ' -f1–18,43- filename > newfile*	
Combining/ splitting files	Stack files column-wise	*paste file1 file2 > newfile*	Useful if you have two or more files with the same rows but different sets of columns and you need to combine them side-by-side
	Stack files row-wise	*cat file1 file2 > newfile*	Useful if you have two or more files with the same columns and you need to 'pancake' stack them. Assumes file2 doesn't have a header row. If it does, first remove it using a one-liner :-)
	Split a file into two files with 3000 rows in the first file and the rest in the second	*csplit -sf prefix filename 3001*	Useful for train/test splitting (the resulting files will be prefixed with whatever you specify as 'prefix')

* The first one-liner is for tab-delimited files, the second for comma-delimited files, and the third for space-delimited files.

Random selection	Randomly shuffle the rows and save to a new file	*awk 'BEGIN{srand();}{print rand()"\t"$0}' filename | sort -k1 -n | cut -f2- > newfile*	
	Randomly choose X% of rows and save to a new file (X= 10% in the code snippet)	*awk -v X=10 'BEGIN {srand()} rand() <= 0.01*X' filename > newfile*	
	Select every 10th row and save to a new file	*awk 'NR%10' filename > newfile*	

* The first one-liner is for tab-delimited files, the second for comma-delimited files, and the third for space-delimited files.