Subsampling on large datasets

This project is a set of scripts for creating validation and training sets for the CBDA machine learning project.

The detailed ideas are in the file **instruction.md**. For the running instructions, see below.

The idea is to use machine learning to examine patient data, including patient outcomes, using machine learning algorithms, to determine those patient attributes that best determine patient outcomes, and how those attributes map to outcomes.

The intent is that once the mapping is determined, it can be used in a clinical setting to map a particular patient's attributes to the most likely outcome. In effect, to improve the diagnoses of clinical conditions.

The machine learning process is to take an actual clinical data set (typically a large one) of known patient attributes and outcomes and divide it into various subsets. The solutions we provide enable us to pick the desired columns without reading the whole datasets into memory.

Running instructions

1. First run:

   ```
   get-original-file-info.py -i test-dataset.csv -o odfi
   ```

   This produces file, odfi.pickle: Has the line count and column count of test-dataset.csv.

2. After generating two files containing the datasets for training and testing datasets, we will run(for example):

   ```
   create-sets.py -i test-dataset.csv --odfi odfi.pickle --trc 2 --vrc 4 --cc 2
   --cn 1 --oc 2 --tsc 4 --cs col.txt --ex --tr 0.8 --asc
   ```

   Required arguments:

   - **'-i', '--original-file'**: The file name of the original data set
   - **'--odfi', '--original-data-file-info'**: The file name of the Pickle file with the original data file
   - **'--trc', '--training-row-count'**: The number of rows to extract for each training set.
   - **'--vrc', '--validation-row-count'**: The number of rows to extract for each validation set.
   - **'--cc', '--column-count'**: The number of columns to extract for each validation
   - **'--cn', '--case-column'**, dest='caseColumn': The case number column ordinal
   - **'--oc', '--outcome-column'**: The outcome column ordinal
   - '--tsc', '--training-set-count': The number of training sets to create

   Optional arguments:

   - **'--mvs', '--multiple-validation-set'**: If present, just create a validation set for all training sets. Otherwise, create only a validation set for each training set *****
   - **'-s', '--starting-set-number'**: The starting set number, default to be 1

    - **'--cs', '--column-set' filename**: The file name of a file with a resticted set of column ordinals
    - **'--ex', '--external'**: Whether or not doing external subsampling, default to be false *****
    - **'--tr', '--training-percent'**: For external subsampling, setting the percentage of training samples *****
    - **'--asc', '--ascending'**: Whether or not choosing columns in ascending order in each training/validation set, default to be false *****

The ones ending with * are newly added arguments or modified arguments compared with the original version.

```
msg = 'Whether or not doing external subsampling.'
parser.add_argument('--ex', '--external', \
        dest='external', help=msg, \
        action='store_true', default=False)
```

```
msg = 'For external subsampling, setting the percentage of training samples'
parser.add_argument('--tr', '--training-percent', dest='trainingPercent', \
        help=msg, type=float, default=0.8, required=False)
```

```
msg = 'Whether or not choosing columns in ascending order in each training/validation set..'
parser.add_argument('--asc', '--ascending', \
        dest='ascendingCol', help=msg, \
        action='store_true', default=False)
```