Sample data – input.csv



Requirements:

* One python program to read the CSV file, write a new CSV file, converting each row of data to an array of double-precision numbers.
* One python program to convert the array of double-precision numbers (obtained from 1st program, above) back to each original representation (except discarded data elements, which are lost).
* Field by field conversion requirements:
  + System\_id: DISCARD
  + Address: DISCARD
  + Addr\_number: Parse directly as a DOUBLE
  + Addr\_drctn: split to two fields: YDir=North(1.0)/South(-1.0), XDir=East(1.0)/West(-1.0)
  + Addr\_street: truncate to 16 characters; convert to uppercase; then one-hot encode each letter into its own set of 26 doubles. You may read the article below about one-hot:  
    (<https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/>)
  + City: one-hot encode based on entire observed values (not character-by-character); that means the field is expanded into N double values where N is the number of possible distinct values observed in this field; it also means the City names have to be written into the field headers in order to be able to convert back from numeric later on. E.g. if there were 3 possible values X, Y, Z in the column City, they will be expanded to columns City:X, City:Y, City:Z, each one having a value of 0.0 or 1.0.
  + State: DISCARD
  + Zip: one-hot encode based on entire observed values (not character-by-character)
  + Dir: take the LAST 32 characters, convert to uppercase, one-hot encode each letter into a separate field.
  + Neighborhd: one-hot encode based on entire observed values (not character-by-character)
  + Cross\_st: one-hot first 16 characters as uppercase, like Addr\_Street
  + Complx: one-hot encode based on entire observed values (not character-by-character)
  + Market\_area: one-hot encode based on entire observed values (not character-by-character)
  + Community: one-hot encode based on entire observed values (not character-by-character)
  + Mapcol: convert the letter to its ASCII value
  + Mappage: Parse directly as a DOUBLE
  + Maprow: Parse directly as a DOUBLE