# Geocoding With Machine Learning: Requirements Document

Copyright© 2018 freelancer.com user afterhourstech

Contents

[Geocoding With Machine Learning: Requirements Document 1](#_Toc515272299)

[Overview 1](#_Toc515272300)

[Input Format 1](#_Toc515272301)

[Output Format 1](#_Toc515272302)

[Algorithm Specifications 1](#_Toc515272303)

[UI/UX Specifications 1](#_Toc515272304)

[Technical Requirements 2](#_Toc515272305)

[Programming Language 2](#_Toc515272306)

[Libraries 2](#_Toc515272307)

[Runtime Environment 2](#_Toc515272308)

## Overview

I need someone to develop for me a very simple, production-ready machine-learning (ML) application. All the details are specified in this requirements document and no “invention” is required.

## Input Format

The input file (input.csv) contains the fields: [address addr\_number addr\_drctn addr\_street city state zip dir neighborhd cross\_st cmplx market\_area community mapcol mappage maprow]. The [address] field is redundant and should be ignored as an ML input, but is required for referencing and joining rows.

## Output Format

The reference file (reference.csv) contains the fields [latitude, longitude]. The two files are related to the input file by the [address, city, state, zip] values.

## Algorithm Specifications

The algorithm chosen for this project is a Deep Neural Network with an Auto-Encoding Layer.

* I need the ML algorithm to be trained on the reference latitude, longitude values using the inputs from the input file, but this will not be done directly in one step. There is a separate auto-encoding step (Neural Network #1) before training Neural Network #2.
* Remember do not use the [address] field in the algorithm input, because it is redundant data.
* The business goal: Given any out-of-sample input (i.e. a future input or an input left out from the training to simulate “future input”), the trained algorithm must predict the latitude/longitude values.
* Null or missing data must be handled by ignoring null or missing data in the error evaluation function. Other means of handling null or missing data are not acceptable.
* The accuracy of prediction must be better than the average distance to Zip Code centroid, since this is an obvious performance level that can be accomplished without any Machine Learning by simply calculating the centroid of all locations at each Zip Code and then assigning all predictions to the same value. Failure to exceed this benchmark will indicate an incorrect implementation and will not be accepted.

## UI/UX Specifications

The solution must have a CLI (Command Line Interface). A GUI (Graphical User Interface) is not required.

The following commands must be supported:

* autoencoder\_train -input <input.csv> -neurons <N> -o <autoencoder.dat>
  + reads <input.csv>
  + <N> is the number of neurons for a single hidden layer
  + writes a few file: <autoencoder.dat>
  + output: the final error rate in json format: { error\_rate: <E> }
* autoencoder\_run –input <input.csv> -encoder <autoencoder.dat> -o <encoded.csv>
  + reads <input.csv> (could be same file as above or a different file in same format)
  + reads <autoencoder.dat> which was output from previous command
  + output: <encoded.csv> containing auto-encoded data. Each row is the autoencoder output plus the referential meta-data [address, city, state, zip]
* network\_train -input <encoded.csv> -trainset <reference.csv> -neurons <N> -o <nn.dat>
  + reads <encoded.csv> which is output from previous command, contains encoded input
  + reads <reference.csv> which is given file
  + uses the referential meta-data [address, city, state, zip] to relate each row in the encoded input file <encoded.csv> to the corresponding row in <reference.csv>.
  + trains a neural network using a single hidden layer of <N> neurons
  + writes the trained neural network to a file <nn.dat>
* network\_run –network <nn.dat> -input <encoded.csv> -o <predictions.csv>
  + reads <nn.dat> the serialized network from previous command
  + reads <encoded.csv> : the encoded input from a previous command
  + writes <predictions.csv> which is the nn output, in the same format as reference.csv
* evaluate -input <predictions.csv> -reference <reference.csv>
  + Reads both files and prints the prediction error for each row matched using the referential meta-data [address, city, state, zip].

## Technical Requirements

### Programming Language

You may choose the programming language from one of the following: Java, Python, R, C, C++ or Scala.

### Libraries

Existing open-source Machine Learning algorithm implementations must be used. Custom-coded algorithms or proprietary libraries will not be accepted.

### Runtime Environment

The solution must be able to run natively without modification on Windows 10 (April 2018 Edition) and Linux 16.04 (Ubuntu). The use of Virtual Machine or Containers in lieu of portability is not acceptable.