**Real Time Location System - Predicting Location with K- Nearest Neighbor**

Bin Yu, Nuoya Rezsonya, Yejur SIngh Kunwar, Iram Bakhtiar

MSDS 7333

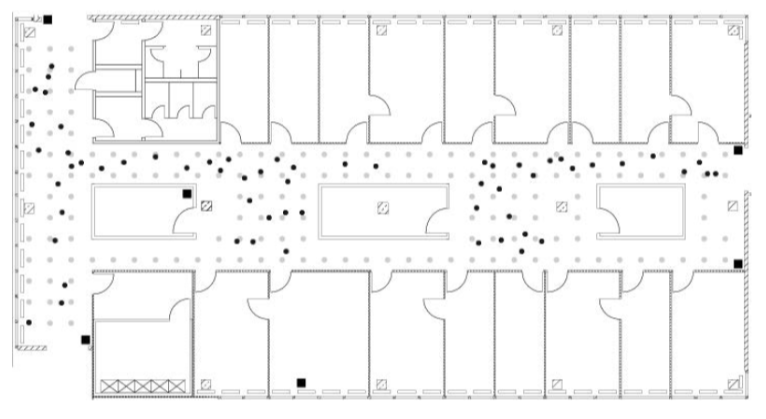
**Introduction**

Real-time locating systems(RTLS) are utilized to identify and locate objects or people in real time, usually indoor. In most RTLS, a wireless tag will be attached to the desired objects so that fixed reference points could receive the signals from the tags to determine the locations. In the real world with smart homes and buildings, this technology has been used in assembly lines to locate the automobiles, hospitals to locate medical equipment and warehouses to locate the products etc.

In this case study, RTLS uses access points to measure the strength of signals from a handheld device and then predict the handheld device's location using K-NN algorithm. The way RTLS works could be broken down into three steps. The first step is to acquire a reference set of data before implementing this application. This data contains the measurements of the signal strength between a hand-held device and fixed access points. It is also referred to as offline data in the study and is considered training data. The second step involves using the training data to establish a model for the location of the device as a function of the strength of signals between the device and each access points. The third step is using the model to predict the location of a tag when the location is unknown.

In the case study from Nolan and Lang [1], the offline data has been provided which has 6 access points (shown as black squares in figure 1) within a hallway on a specific floor at University of Mannheim and 166 known points/locations with 1 meter from each other as tags. The grey dots denote the offline data or training data, and the black dots denote the online measurements recorded at randomly selected points. See Figure 1: The Floor Plan.

**Figure 1: Floor Plan of the Test Environment**



This offline data has been used to predict the locations of a device whose location is unknown. Predictions have been tested on the testing data which is the "online" data. In "online" data, 60 locations which are not included in the previous 166 points and orientation have been chosen randomly. A total of 110 signals have been measured from the devices to each access point. "Online" data is used to determine the performance of the RTLS model.

In our case study, we will conduct a more thorough data analysis into these two MAC addresses (their selection will be detailed in ensuing sections) including determining locations by using data corresponding to both MAC addresses. Then answer the following questions:

* Which of these two MAC addresses should be used and which should not be used for RTLS?
* Which MAC address yields the best prediction of location?
* Does using data for both MAC addresses simultaneously yield more, or less, accurate prediction of location? (Note: this portion is derived from Exercise Q.9 in Nolan and Lang.)
* While k-nearest neighbors has proven to be a good approach to determine location, alternate approaches have been proposed. One simple alternative approach is to use weights on the received signal strength, where the weight is inversely proportional to the “distance” from the test observation. This allows for the “nearest” points to have a greater contribution to the k-nearest neighbor location calculation than the points that are “further” away.
* Implement this alternative prediction method.
* For what range of values of weights are you able to obtain better prediction values than for the unweighted k-nearest neighbor approach? calcError() can be used to compare this approach to the simple average.

**Methodology**

The methodology followed to conduct the analysis included data exploration, data preparation (required enormous amount of data cleansing before the data could be used), data visualization and lastly an improved K-NN model(this can be changed) to more accurately predict the location of a hand held device that the location is unknown.

The code used in the case study is the modified versions of the example code provided from Chapter 1 of Data Science in R: A Case Studies Approach to Computational Reasoning and Problem Solving by Deborah Nolan and Duncan Temple Lang. The source code can be found at http://rdatasciencecases.org/code.html and the dataset is at <http://rdatasciencecases.org/Data.html>.

**DATA EXPLORATION**

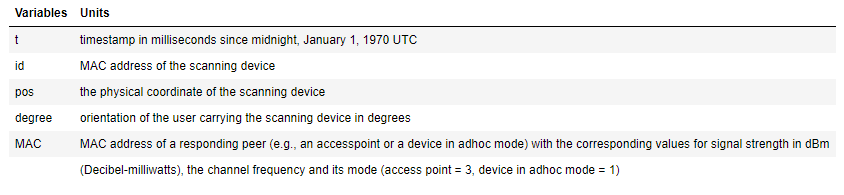
For our case study analysis, we will be using the offline data provided in the book, Data Science in R: A Case Studies Approach to Computational Reasoning and Problem Solving by Deborah Nolan and Duncan Temple Lang.

As mentioned earlier, we will use this offline data as the training data to build the prediction model. The training data set has a total of 151,392 rows of which 5,312 rows are comments. We are not concerned with the comments, and as a result a total 146,080 data rows will be used in our data analysis. This is matching with our expectation of the number of data records in the file (166 locations x 8 angles x 110 recordings).

**Data Import**

The data are not provided in a tabular data format and the function read.table() cannot be utilized to directly import it into the data frame. It requires significant processing before the data becomes ready to be used for our analysis. After data is imported, it is observed that the main data elements are separated by semicolons. Also each field has the label and "=" followed by a value. It can be seen that the last part of the record is really a matrix containing 4 columns: MAC, orientation, channel frequency, and mode. In the first record, there are 8 MACs has mode 3(access point) and 2 MAcs with mode 1(Adhoc mode).

All the records/data points in the data set contain the following Units of Measurement fields [1](from Nolan and Lang Book,page 7 Table 1.1: Units of Measurement)



**Preparing Data For Analysis**

The data used for this analysis was not structured and could not be imported using readlines() into a dataframe, it required significant processing for it to get ready to be used. Each line was parsed using strsplit function and ;=, was used as token to parse each line. We create a function that will define the logic for importing data and for creating matrices for each line of data in the “offline” dataset and merge into a data frame that will be used for the final analysis.

We are not concerned with the comments, thus all comments are removed and the dataset is saved into offline data frame. The data frame is comprised of 1,181,628 records and 10 columns.

To further prepare data for analysis, we also provide headers to the variables and convert the variables to the correct format for the analysis. We now have 10 columns in our data frame and the variables for time, position X, position Y, signal, and orientation are all converted to numeric variables.

Since we are only use the signal strengths measurement in our analysis, we don't need adhoc (type =1) data. all rows for ad-hoc mode are removed which make up around 203,85 records from the data frame. We remove the field type as well since all the types are access points only. After this cleanup process, we are left with 978,443 rows and 9 columns in the offline data frame.

Skipped to last sections…had to tweak the above…my OCD

**Find the Nearest Neighbors**

**k-Nearest Neighbor**

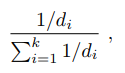
In the case study, the simple and intuitive non parametric machine learning technique k-Nearest Neighbor (or kNN) algorithm was used to predict the new locations. kNN is based on the premise that new observations will be compared to known locations of known data points/observations. To explain further where k=1, the location of the new observations (signal strengths by a new device in an unknown location) is assigned or estimated by finding the observation/location in our training dataset with the closest signal strength. The position of the new device is predicted by aggregating the locations of the k-points with the closest signal strength in our training data set.

We need to reiterate that Clustering is quite subjective, and the number of neighbors being assigned will directly affect the performance of the algorithm. That is the reason why the numbers of neighbors in a kNN model is so important and should be adjusted to either improve underfitting or reduce overfitting.

So far we have determined that using the k-NN algorithm with the mean squared error as the performance evaluation metric, removing 00:0f:a3:39:e1:c0 will yield the best accurate prediction of location since this method grants the lowest mean square error.

So, while k-nearest neighbors has proven to be a good approach to determining location of new devices, alternate approaches have been proposed. One simple alternative approach is to use weights weighted k-NN algorithm. In the weighted k-NN algorithm all the k-nearest neighbors which are particularly close to the new observation (y, x), should get a

higher weight in the decision than such neighbors that are far away from (y, x)[2] (https://epub.ub.uni-muenchen.de/1769/1/paper\_399.pdf). For our case study this means that we need to use weights on the received signal strength, where the weight is inversely proportional to the “distance” from the test observation. This allows for the “nearest” points to have a greater contribution to the k-nearest neighbor location calculation than the points that are “further” away. According to the authors Nolan and Lang, the following computation maybe used for assigning weights to each k nearest neighbor:



In the above formula “di” is the distance from the new point to the nearest k-neighbor in terms of signal strength.

We implement this alternative prediction method by modifying the findNN function to calculate the distance from the test data to all of the training data and get the top 3 nearest k-neighbors. Further, the predXY function is modified as well to calculate the wights for te k-nearest neighbors. The weights array are multiplied by matrix of x,y to get the weighted values. Running the weighted k-NN it is observed that K Min = 6 produces the optimal results with a MSE of 1015.18

|  |  |  |
| --- | --- | --- |
| Regular KNN without 00:0f:a3:39:dd:cd | Error: 1038.5 | K Min = 6 |
| Regular KNN without 00:0f:a3:39:e1:c0 | Error: 935.61 | K Min = 6 |
| Regular KNN when keep both | Error: 1100.0 | K Min = 4 |
| Regular KNN using Manhattan Distance | Error: 51134.68 | K Min = 20 |
| Weighted KNN | Error: 1015.18 | K Min = 6 |

Looking at the table above, when we compare the MSE values, weighted KNN is very minimally better at predicting with an MSE of 1015.18 compared to the Regular KNN without 00:0f:a3:39:dd:cd with an MSE of 1038.5

The Manhattan distance is based on absolute value distance, as opposed to mean squared error. In other words the distance between two points is the sum of the (absolute) differences of their coordinates [2] (add to refs <http://www.ieee.ma/uaesb/pdf/distances-in-classification.pdf>). The below is the calculation for the Manhattan distance:

