Unit 2 Case Study – Real-Time Location Systems
Daniel Davieau, Lei Jiang, Kristen Rollins
21 January 2020

I. Introduction

Due to the surge in growth of wireless networking, being able to dependably track people and things has become a subject of high interest. Such tracking has abundant uses, such as tracking merchandise in stores to prevent theft, recording the location of a medical patient who may have a flight risk, or logging item locations in a warehouse to efficiently ship products. Not only should the movements be recorded, but locating these objects in real-time is often of utmost importance. Therefore, real-time location systems (RTLS) have understandably become a substantial topic of research. Specifically, indoor positioning systems (IPS) are the subject of this investigation, as they improve upon the shortcomings of GPS systems and are made possible via now-ubiquitous WiFi signals.

A statistical IPS system was developed for research in a building at the University of Mannheim, and the experiment has been described and analyzed in detail in the Nolan and Lang textbook, *Data Science in R*. We were tasked with expanding the analysis found there to explore possible improvements to the RTLS system. Specifically, we examined their decision to remove a redundant router from the training data, tested the system without removing any access points, and implemented a weighted *k*-Nearest Neighbors (*k*-NN) approach to supplement their conventional *k*-NN method.

Our analysis found that excluding the access point with MAC address ending in c0 yielded better results than excluding the point ending with cd. Using both access points in conjunction did not yield better results. We also found that a weighted *k*-Nearest Neighbors approach did yield increased performance as opposed to the ordinary *k*-Nearest Neighbors methods.

II. Background

It is important to understand the setup of the RTLS system that this analysis is built upon, in order to grasp the context of the predictions we will be making. First of all, the system was built on the first floor of a university building; the building has many internal rooms and walls, which may introduce error but also opportunity for experimentation and improvement. The 15x36 meter floor plan can be seen in Figure 1.

In the design, there are 6 routers, or access points, scattered around the floor denoted by black squares. The grey circles represent “offline” data, i.e. measurements taken with hand-held devices at fixed distances of one meter apart. The offline data coming from these locations provide training data which can be used to predict the location of new devices on this floor. The black circles denote “online” data, a set of randomly positioned devices that were collected to test the RTLS system.

The offline and online data provided with this system required rather extensive cleaning to get it into a usable format. This process is documented in the Nolan and Lang text, chapter 1, which we chose to follow. Additionally, extensive exploration of the data was performed in the

text, which we followed closely as well in order to understand the structure and characteristics of the data. For brevity we do not include all this work here, but these supplemental visuals can be explored in our code base (referenced in Appendix).

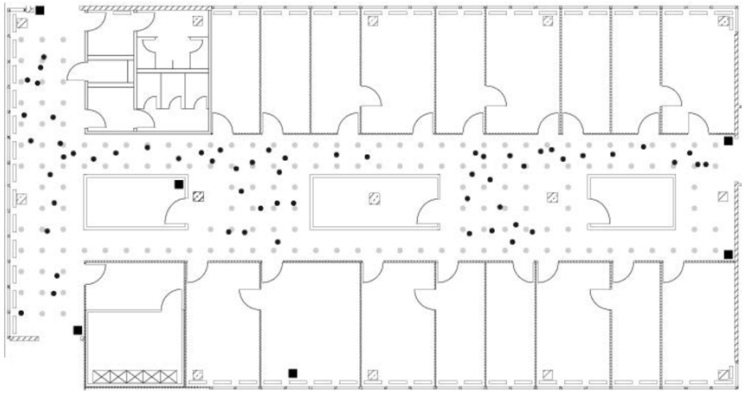

Figure 1. Floor Plan of the Test Environment

The primary aspect of the data exploration to note is that while 6 access point are depicted in the RTLS design, there were actually 7 distinct MAC addresses of routers that recorded signals. It was found that two of these access points actually had the same x and y coordinates within the building, thus they were expected to be redundant. The Nolan and Lang text chose to remove the router address ending with "cd" rather than the address ending in "c0", presumably because there were slightly more signals recorded for the c0 access point. As seen in Figure 2, the behavior of these two routers was not equivalent with respect to orientation of the offline devices, suggesting that the choice of which router to preserve may result in differences in predictions. Our objective was to determine whether the correct decision was made in the textbook analysis.

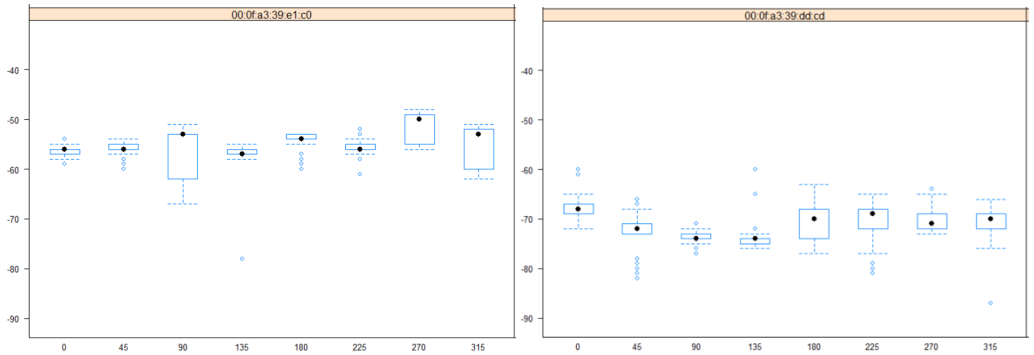

Figure 2. Signal Strength vs. Orientation for c0 (left) and cd (right) MAC addresses

III. Methodology

The primary method we used to determine which access point(s) to include was the *k*-Nearest Neighbors (*k*-NN) algorithm. In this method, the *k* nearest training points are found for each point in the test set, whose values are averaged to predict the value of the test data point.

In our context, we calculated the similarity between the 6 (or 7) measured signal strengths of an online point and the 6 (or 7) signal strengths recorded for the offline points. Then the known (x, y) locations of those closest *k* neighbors were averaged to predict the location of the online device. Note that we used Euclidean distance (the typical straight-line distance) for this similarity calculation. Figure 3 indicates that the signal strength is negatively correlated with the distance. Therefore, the further the access point is, the weaker the signal is.

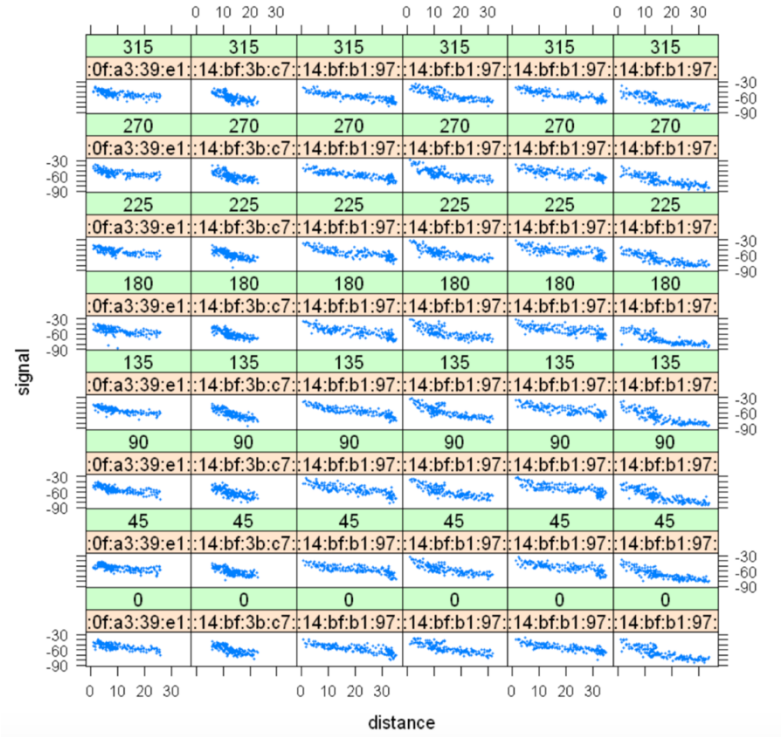

Figure 3. Signal Strength vs. Distance for 6 Access Points

The value *k* can be tuned to reduce overfitting or underfitting concerns. To determine the optimal value of *k* for our algorithms, we used the mean squared error (MSE) metric to measure the error between the actual and predicted locations. Additionally, we used 11-fold cross validation in conjunction with the MSE to determine the optimal *k* while excluding the c0 address, excluding the cd address, and preserving both. After comparing the performance of these models, we implemented a weighted *k*-Nearest Neighbors algorithm to apply to the best combination of access points. The weighted *k*-NN uses the same concept as ordinary *k*-NN, but instead of each neighbor having equal "voting" power, the neighbors have weights inversely proportional to their distance (in signal strength) from the point being predicted. In this way, we still incorporate information from all of the *k* closest points, but the closer points will have more influence. For this method we used the formula

$$\frac{1/d_i}{\sum_{i=1}^{k} 1/d_i}$$

for the weights, as suggested in the Nolan and Lang text.

IV. Results

Figure 4 illustrates the results from running all of our *k*-NN models, both ordinary and weighted. The graph depicts the cross-validation error for the online data for different values of the parameter *k*, which we allowed to range from 1 to 20. For all the models, we see a very high error rate when *k* is 1, then a steep drop as *k* increases, then gradually increasing errors as *k* becomes too large. Since we want to minimize the error, for each of these lines we chose the *k* that resulted in the lowest MSE.

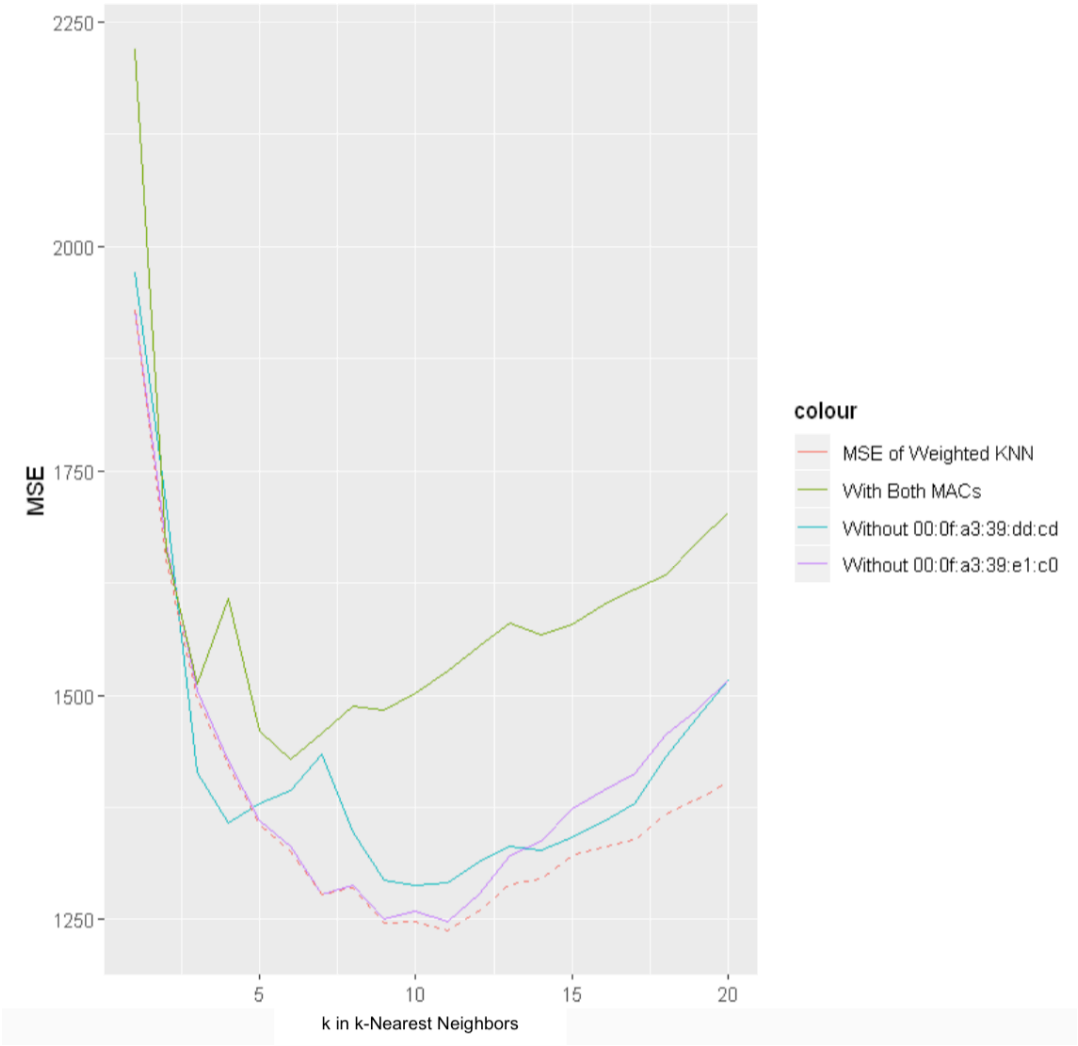

Figure 4. Comparison of Learning Curves across Models

We were first interested in comparing the three regular *k*-NN models; one where we excluded the cd MAC address (as the textbook did), one where we excluded the c0 address, and another where we used both access points to make predictions. We found that excluding the c0 address yielded better results than removing the address ending in cd. The difference in MSE was not excessive, but substantial enough to conclude that the Nolan and Lang text excluded

the wrong access point. They were correct to exclude one of the access points at least, since keeping both routers resulted in considerably worse predictions.

After making this first discovery, we then applied our weighted *k*-NN model to the data while excluding the c0 access point, since this yielded the best predictions before. We observed from Figure 4 that the weighted model closely follows the errors of the regular model with the corresponding access points, until around *k*=9 where they begin to diverge. This was expected behavior because as *k* increases and more neighboring points are included, the weights of those more distant points will differ more between the models (i.e. a weight of 1, versus 1/d). Here we found that the divergence was in favor of the weighted model, as it achieved the lowest error rate overall.

Table 1 summarizes our findings. As seen, out of the three ordinary *k*-Nearest Neighbor models, the model that excluded the c0 MAC address yielded the best results (lowest MSE). Using a weighted *k*-Nearest Neighbors model further improved the performance of the RTLS system, as it achieved the lowest error out of all the models.

Table 1. Model Comparison

Model	**MSE**	**Optimal *k***
k-NN, excluding cd	1288.55	10
k-NN, excluding c0	1247.37	11
k-NN, keeping both MAC addresses	1428.89	6
Weighted *k*-NN, excluding c0	1237.72	11

V. Conclusion

In summary, our results indicated that for this particular RTLS system, excluding the MAC address ending with c0 produced better results than removing the cd address, in contrast with the decision in the Nolan and Lang text. We also evaluated the system while preserving all 7 MAC addresses, but this yielded poorer results. Furthermore, for this system we would recommend utilizing the weighted *k*-NN algorithm that we created since it yielded the best results overall. However, if extending these results to new RTLS systems, we would recommend continuing with a regular *k*-NN model to predict the location of new devices. We hold this opinion because developing the weighted *k*-NN took significant developer-time, and we do not believe the increase in performance to be substantial enough to justify this added effort. Nevertheless, we hope this investigation has enlightened and improved the use of RTLS systems and their abundant applications.

VI. References

- Nolan, D., and Temple Lang, D. (2015), *Data Science in R: A Case Studies Approach to Computational Reasoning and Problem Solving*. Boca Raton, FL: CRC Press (NTL)
- R code provided by Dr. Robert Slater, adapted from http://rdatasciencecases.org/

VII. Appendix

Our full R code base, including supplemental visuals, can be found in a separately submitted Jupyter notebook file, called lJiang_kRollins_dDavieauCaseStudy2_Code.ipynb.