Unit 6: Case Study on Using KNN with Wireless Networks for Location Detection

Damon Resnick
Collaborators: Rajni Goyal and Jim Hosker

February 22, 2018

Abstract

K nearest neighbors is used to estimate the locations of a wireless device using the strength of the signal it measures from multiple wireless access points and wireless devices around it. A weighted average of the k nearest neighbors is used to slightly improve the estimated locations. Signal analysis is also performed to explore the signal strengths of the wireless access points to determine which access points are better at predicting the device location.

1 Introduction

This case study focuses on the use of k nearest neighbors to determine the location of a wireless device based on the strength and positions of the signals of the wireless devices around it. This work mirrors the work done in chapter one of "The R Series" text book "Data Science in R A Case Studies Approach to Computational Reasoning and Problem Solving" by Deborah Nolan and Duncan Temple Lang. [1]

1.1 Training Data

The data in this case study is data from a handheld device of roughly 7 access points and many multiple other wireless devices scattered throughout a typical modern office floor. From the information and question page of the case study a good description of the data is provided. Here is a summary of what was provided in that description:

This is the first line of the raw data from the training set and details the first observations from position (0,0) at an angle of 0°.

```
t=1139643118358;id=00:02:2D:21:0F:33;pos=0.0,0.0,0.0;degree=0.0;
00:14:bf:b1:97:8a=-38,2437000000,3;
00:14:bf:b1:97:90=-56,2427000000,3;
00:0f:a3:39:e1:c0=-53,2462000000,3;
00:14:bf:b1:97:8d=-65,2442000000,3;
00:14:bf:b1:97:81=-65,2422000000,3;
00:14:bf:3b:c7:c6=-66,2432000000,3;
00:0f:a3:39:dd:cd=-75,2412000000,3;
00:0f:a3:39:e0:4b=-78,2462000000,3;
00:0f:a3:39:e2:10=-87,2437000000,3;
02:64:fb:68:52:e6=-88,2447000000,1;
02:00:42:55:31:00=-84,2457000000,1
```

Taken from Table 1, page 7, of the Nolan and Lang text:

t in units of milliseconds and is the number of milliseconds since midnight, 1/1/1970 UTC.
id is MAC address of the scanning device.
pos represent the x, y, z coordinates of the scanning device.
degree represents the orientation angle of the scanning device in degrees.

The subsequent data are the MAC addresses, signal strength in dBm, channel frequency, and the devices mode of operation (either 3 for access point or 1 for device in adhoc mode) for the observations taken at that time, position, and orientation by the scanning device.

Data exploration is detailed in the first few sections of the text. The code used is straightforward. Brief comments on different parts of the code are provide within the code attachment provided.

The training data includes roughly 150,000 measurements taken with 5-13 distinct observations per measurement depending on the location and orientation of the scanning device. All measurements were taken by the same scanning device. The device took measurements at 166 different locations in a large office floor roughly 35×18 m^2, with eight different orientations in increments of 45 degrees: 0, 45, 90, 135,180, 225, 270, and 315. The size and shape of the room is shown in Figure 1 as well as the approximate locations of 6 of the access points.

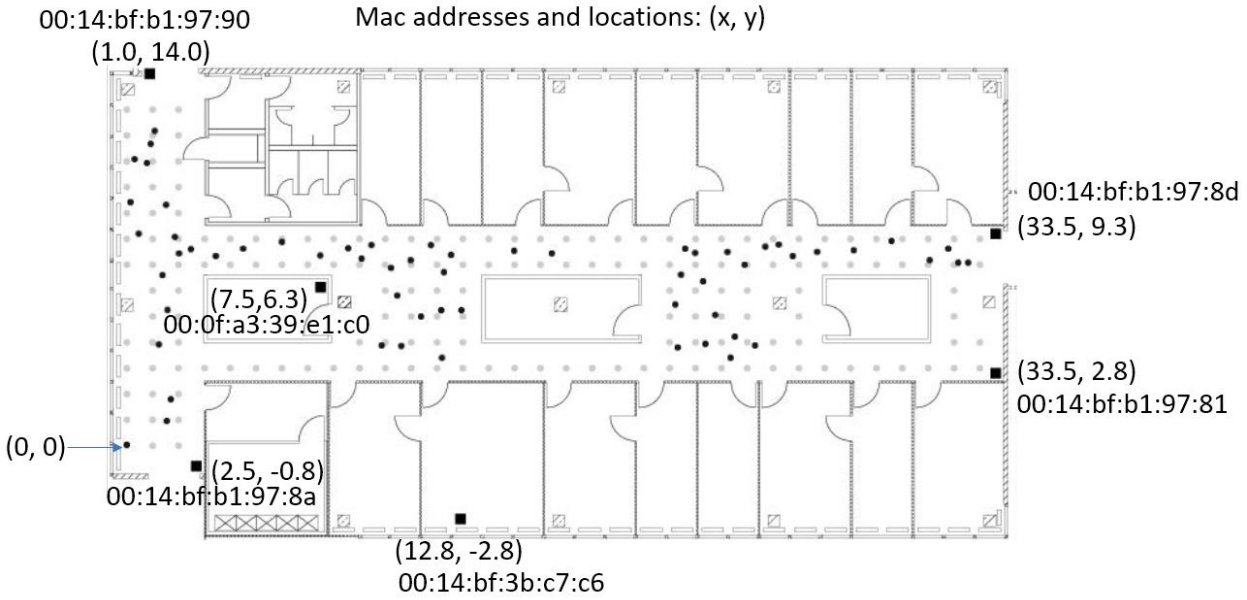

Figure 1: The size of the office floor is roughly 35×18 m^2. The black solid squares represent the locations of the 6 major access points while the black dots represent the possible locations of the scanning device. The specific x and y coordinates and mac addresses for the 6 access points are also shown as well as the origin. [1]

The exploratory data analysis was done by following the code in the text. The result of the exploration was the creation of a function that makes all the data cleanup in one function called simply readData(). The readData() function also uses the processLine() and roundOrientation() functions to load, clean, and rearrange the data in a way that makes it easy to perform the k-nearest

neighbors technique. Explanations and comments on the code can be found in the appendix and attached documents.

Roughly 5500 observations were recorded at each location. Figure 2 shows the number of observations at each location on the floor. You can see that most locations have a similar number of observations but there are three gaps in the middle of the room as well as major gaps along the top and bottom parts of the floor. This is presumably due to not being able to take measurements in private offices and does not represent a lack of signal in those reasons.

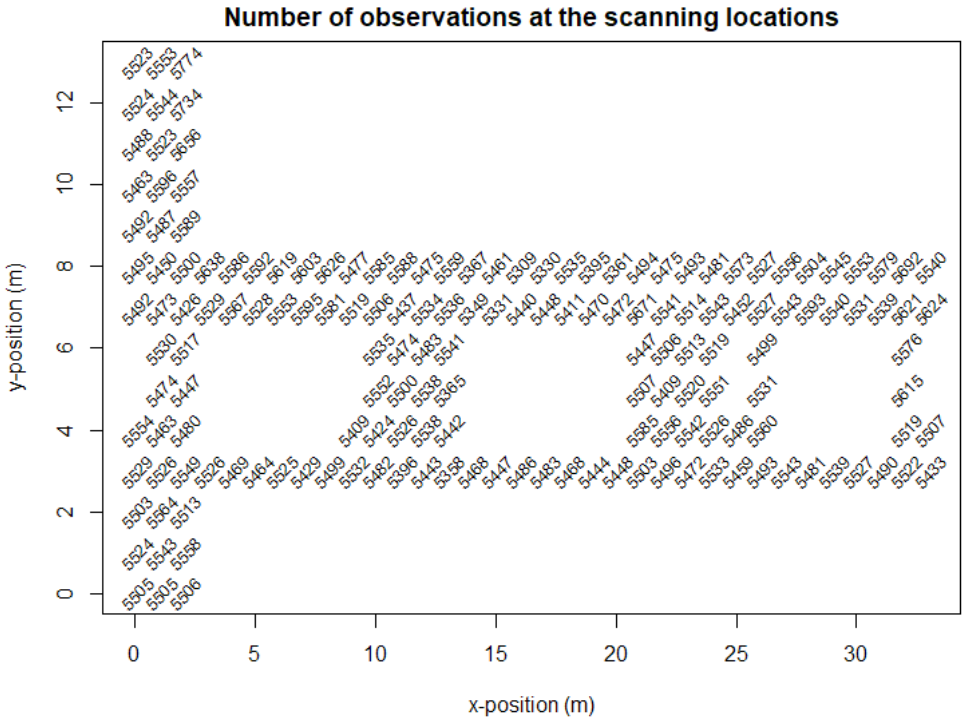

Figure 2: The number of observations for every location are shown.

Another way to visualize the strength of the signal from different access points in different locations is with heatmaps. Figure 3 shows heatmaps for the 7 access points that have the most hits. We know that at least six of these are in the room at the locations shown in Figure 1. Looking closely at the heatmaps we see that the seventh also appears to be located at the same location as the first. These are maps of the signal strength in a plane, so this could mean those two access points are at the same location or that one of them is on the ceiling or in another room right above or below the other.

Another thing to note is how the signal strength falls off with distance. Figure 4 shows the signal strength vs distance for the 6 primary MAC addresses. As you can see there is a tiny bit of curve to the relationship, but it looks very linear. This implies that there is a direct correlation between the signal strength and the distance from the access points. This will come in handy when applying the k-nearest neighbors technique to estimating the location of the scanning device by the strength of the signals around it. This relationship will also further allow us to weight the nearest neighbors by this distance in order to better estimate the location.

Up to this point, the code to generate all these figures but Figure 3 was taken directly from the text. [1] The only changes that were made to the code for Figure 3 involved simply changing the order of access points and the removal of the orientation portion of the heatmaps to show only 1 orientation. Please see the attached code for the details. This was actually a very useful bit of code to run because it leaves no doubt as the location, in the x-y plane, of the seventh MAC address. This proved important when using that access point to help improve the accuracy of the location estimation when k-nearest neighbors is applied!

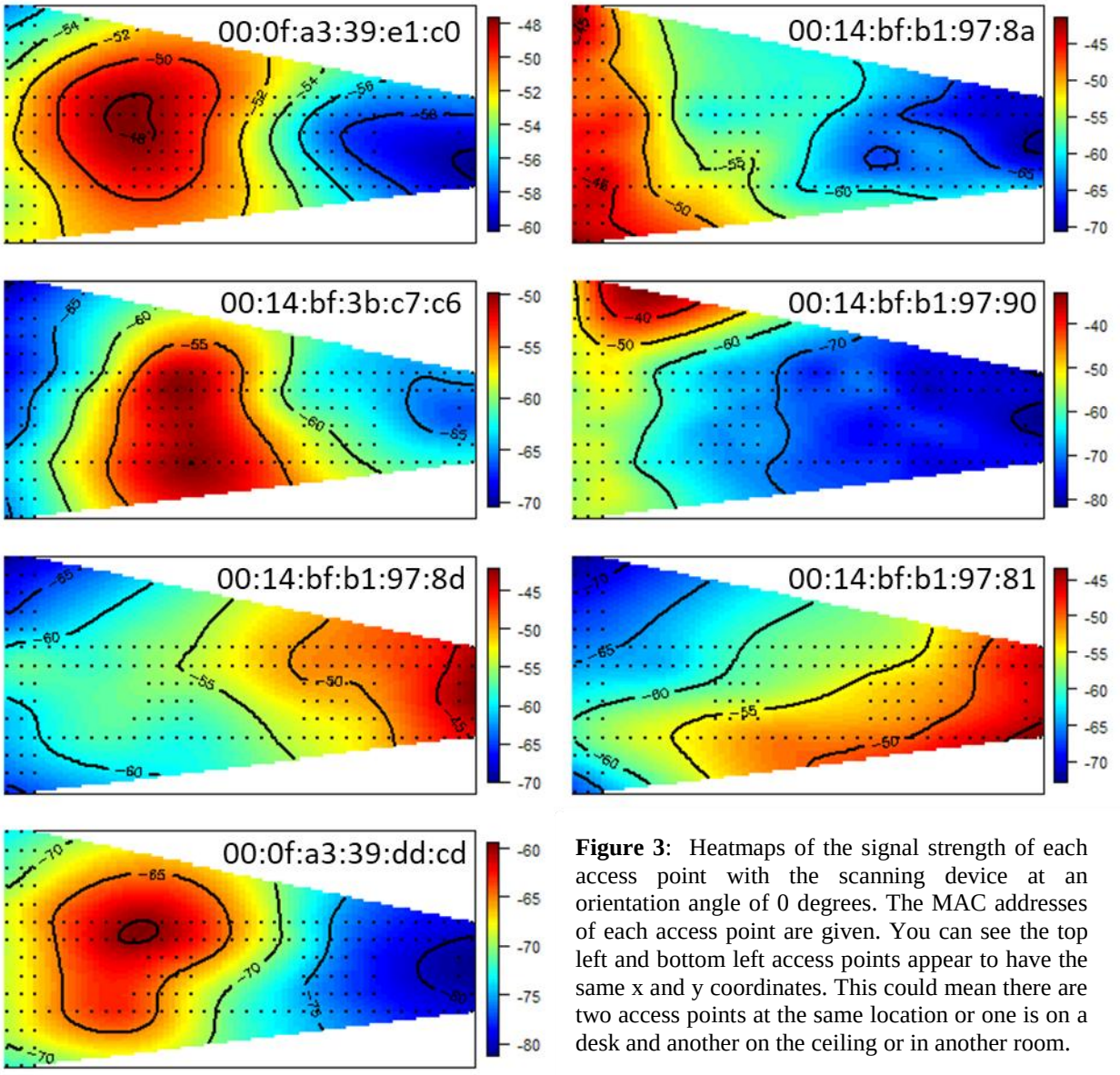

Figure 3: Heatmaps of the signal strength of each access point with the scanning device at an orientation angle of 0 degrees. The MAC addresses of each access point are given. You can see the top left and bottom left access points appear to have the same x and y coordinates. This could mean there are two access points at the same location or one is on a desk and another on the ceiling or in another room.

1.2 Test Data

As we have stated the object of this study is to take data from a scanning device and estimate or predict the location of that device based on the signals it received from the access points around it. In this case we use a k-nearest neighbors, or k-NN, technique. K-NN is a non-parametric

algorithm primarily used for classification but can also be used for regression. The basic idea is that the objects closest to another object are more likely to belong to the same class as the object in question. For this study, we cheat in a way by already knowing the nearest neighbors in a way and averaging their location to estimate the location of the scanning device.

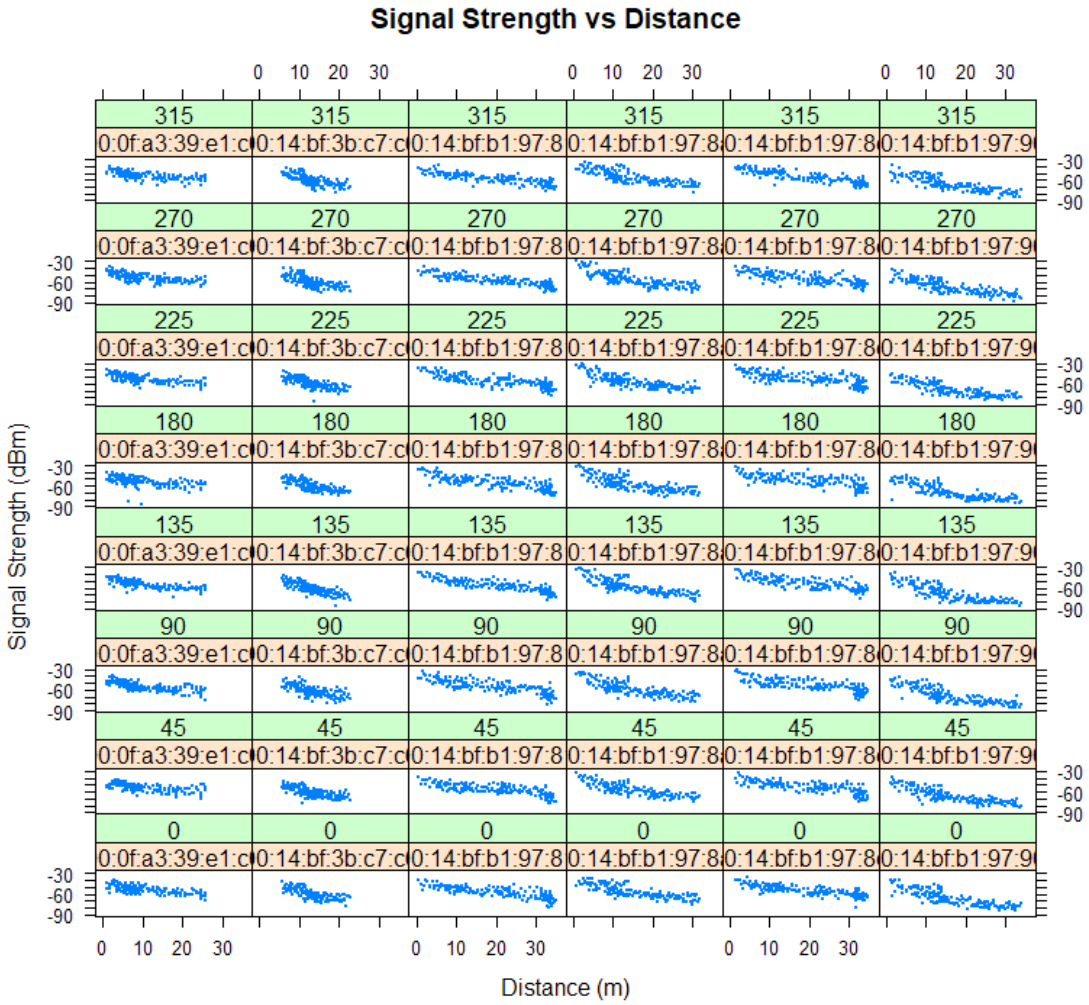

Figure 4: Signal strength vs distance is shown. The basic relationship appears to be roughly linear allowing the use of signal strength as a way of estimating distance.

For this approach the test data, or data used for the estimation, must be prepared in a way to maximize the effects of this technique. This is again outlined in the text with the code attached with extra comments to explain different parts. The test data is very similar to the training data with the major difference being that it has 6800 rows instead of 150,000. The results are determined with the test data while we determine the best k value to use with the training set.

2 K-nearest neighbors

K-nearest neighbors, or k-NN, is a non-parametric algorithm primarily used for classification but can also be used for regression. As stated previously, in this case study k-NN is used by simply

determining the nearest neighbors, by distance from the device. This distance is obtained by utilizing the approximate linear relationship between the signal strength and distance. So that the difference in the signal strength, from the same access point, between two locations is taken to be directly proportional to the distance between the two points. We will see later that this is just an approximation and breaks down in some cases and can negatively impact the estimated locations in certain regions of the floor near certain access points.

3 Results

This case study presented two basic problems to solve with a few twists. The first question is about the mysterious seventh access point. In the text, the primary six access points in Figure 1 are used to estimate the locations of the scanning device. However, there is another access point that appears to be located at the same location as the access point located at (7.5, 6.3) on the map. This was noted in Figure 3. If you look at the heatmaps you can see that the access point e1:c0 appears to be located at the same location as the dd:cd access point. The question is which of these two access points is better to use for predicting the locations of the scanning device? And is it better to use one of those access points or both simultaneously with all the other access points?

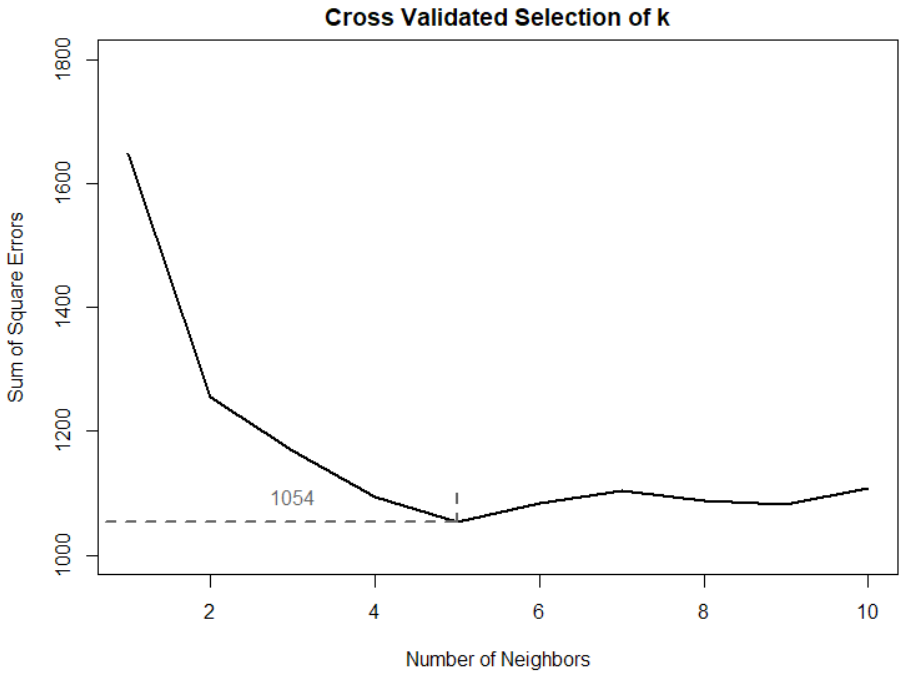

Figure 5: Sum of square errors versus the number of nearest neighbors chosen. This was done on the training set and shows that a value of k = 5 is an optimal value for the number of nearest neighbors to use in estimating the location of the scanning device.

3.1 Which access point is better to use?

To answer the first part of the question we use the k-NN technique. We first use the training set to find the best value of k to use when we estimate the locations of the scanning device in the test data. This was done using the code provided in the text and it was determined that the optimal

value of k fluctuated between 3 and 6. We settled on a k = 5 value as it seemed to more optimally give the better prediction for the case of the six primary access points. Cross validation was used to make this determination. As you can see in Figure 5. A value of k = 5 does a good job.

The metric used to determine the accuracy of these predictions is a simple sum of square errors calculation. Basically, the location of the scanning device is predicted based on the average positions of the k-nearest neighbors. Once this predicted location is calculated it is compared to the actual location of the scanning device and the distance between those two points is calculated. This difference is squared and added to the difference squared of all other measurements to determine a sum of all the square errors. This is a simple metric to use for a location estimation that does not inherently favor any location over any other. However, this metric is only used to show that values match up with those in the text. The sum square error metric does not normalize for the number of locations tested so it should not be used to compare with other data sets. If this study was to be extended to other data sets a version of the sum square errors normalized to the number of locations would be a more appropriate metric.

After the training set was used to determine the value of k the next step was to apply the technique to the test set. Using a value of k = 5 we found that the sum of square errors for the test set using the primary six access points was 275.51. Repeating that calculation while replacing the e1:c0 access point with the dd:cd access point, we obtained a sum of square errors of 249.92. And with all seven access points included the sum of square errors was 209.61.

This was an interesting result. The fact that using all seven access points decreased the error was expected but the fact that the e1:c0 provided a larger error than the e1:c0 case was at first extremely puzzling. If you look closely at Figure 3 you will note that the signal strength of the e1:c0 access point is 10-15 dBm higher on average in most locations. One might think this should provide a better prediction of location, but it does not. Shouldn't a stronger signal correspond with a better estimation? Well in general out in an open field this would probably be the case. But this is an enclosed space, with obstructions on the floor, such as office walls and furniture that can have an effect on the signal. For these reasons the *variation* in the signal strength of e1:c0 is not as uniform and in fact the signal strength itself is uniform in large regions. This is shown best in Figure 6.

If you look carefully at Figure 6, which is a heatmap of the signal strength of e1:c0 and dd:cd at an orientation angle of 45 degrees, you can see that there is a larger region in which the signal strength is very uniform for e1:c0. Since our technique relies on the difference in signal strength to estimate the location of the scanning device, e1:c0 would not give very good estimates in this region because the difference in signal over that region is so small. A distance of several meters may have the same difference in signal strength as the distance of one meter.

Another way to justify the argument that an access point that generates large regions of uniformity reduces the accuracy of the prediction is to look at the heatmaps for other access points and see if removing those points in favor of the others increases accuracy. It was found that removing access point 97:8a produces a sum square errors of 243.57 and removing access point 97:90 produces a sum square error of 228.34. If you look carefully at Figure 3 you can see that both of those access points generate large regions of uniformity. In fact, access point 97:90 generates the largest regions of uniformity of all the access points in regions that have a lot of measurement points.

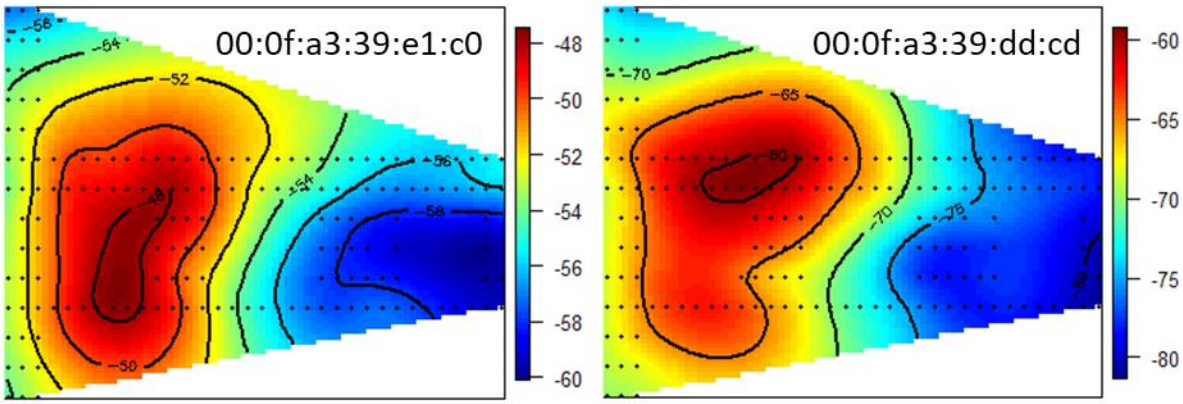

Figure 6: These two heatmaps show the relative signal strength of two different access points at an orientation angle of 45 degrees. There is clearly a larger region, in access point e1:c0, of a very uniform signal. Since the technique used here to estimate the location of the scanning device depends on the difference in signal strength, the e1:c0 access point will provide more uncertainty in the estimated location than the dd:cd access. This effect is present for both access points, but since the region of uniformity is larger for the e1:c0 access point we expect this results in a larger error in predicting the location of the scanning device. Basically, the scanning device has a harder time figuring out where it is because one point looks just like many other points.

3.2 Applying weights to the nearest neighbors

The second broad question to answer is whether adding weights that depend on the inverse of the distance will improve the accuracy of the predicted locations. In theory this makes sense because this would tend to weight the closer nearest neighbors higher than the neighbors farther away. Weight the closer nearest neighbors more heavily should improve the prediction.

In order to change the code provided to apply the weights, changes were made to the findNN() function as well as the predXY() function. In the findNN() function the weights were created by using sort(1/dists). The weighted_closests were created by using trainSubset[closest, 2:3]*weights[1:k]/sum(weights[1:k]). These were then combined: out = cbind(trainSubset[closest,1] ,weighted_closests) as the output of the findNN function. In the predXY() function the only change that was made was to change the mean(x[1:k]) to sum(x[1:k]). This was done as it was mathematically equivalent to a weighted average. These changes are noted and commented on in the code attached.

It was found that weighting the nearest neighbors in this way decreased the sum square error from 275.51 to 270.55 for the first six access points; a reduction of less than 2%. Compared to using the dd:cd access point or all the access points simultaneously it turns out that while using a weighted neighbors approach does decrease the error it decreased the error the least. However, this method can be combined with all the other options and when the weighted method is combined with using all 7 access points the sum square errors did not change measurably at all. It went from 209.61 to 209.62. These two values are practically identical. Therefore weighting the nearest neighbors in this way for k = 5, at least, does not really help all that much.

4 Conclusion

K-NN was used to estimate the locations of a wireless device using the strength of the signal it measures from multiple wireless access points around it. It was determined that MAC dd:cd did a better job of predicting location than MAC e1:c0, and that including all MAC addresses improved the prediction the most. It was also found that a weighted average of the k nearest neighbors was used to slightly improve the estimated locations in the case of using just the first six access points, otherwise it did not noticeably improve the accuracy of the predictions for K = 5 when all access points were used. Signal analysis was also performed to explore the signal strengths of the wireless access points and make some interesting determinations.

References

[1] "The R Series" text book "Data Science in R A Case Studies Approach to Computational Reasoning and Problem Solving" by Deborah Nolan and Duncan Temple Lang, CRC Press, 2015.

Appendix A: Code

Listed here is the code that was changed in order to answer the questions. The rest of the code is in an attached .txt file.

To answer the first question about which access point is better only a few places in the code from the text needed to be changed:

```
offlineSummary = subset(offlineSummary, mac != subMacs[1]) # Changed [2] to [1] to delete the other mac address

# Matrix of positions for access points sans [1]
AP = matrix( c( 7.5, 6.3, 2.5, -.8, 12.8, -2.8,
                1, 14, 33.5, 9.3,  33.5, 2.8),
             ncol = 2, byrow = TRUE,
             dimnames = list(subMacs[ -1 ], c("x", "y") )) ]  # Changed [-2] to [-1] to delete the other mac address

offline = offline[ offline$mac != "00:0f:a3:39:e1:c0", ] # Changed != "00:0f:a3:39:dd:cd" to != "00:0f:a3:39:e1:c0"
```

To answer the first question about whether all access points work better only a few places in the code from the text needed to be changed:

```
# offlineSummary = subset(offlineSummary, mac != subMacs[2]) # This was commented out

# Matrix of positions for access points sans [1]
AP = matrix( c( 7.5, 6.3, 7.5, 6.3, 2.5, -.8, 12.8, -2.8,
                1, 14, 33.5, 9.3,  33.5, 2.8),
             ncol = 2, byrow = TRUE,
             dimnames = list(subMacs[  ], c("x", "y") )) ]  # Changed [-1] to [ ] to delete no mac address

#offline = offline[ offline$mac != "00:0f:a3:39:dd:cd", ] # Commented out
```

And every 4:9 changed to 4:10 in findNN() as well as every 6:11 changed to 6:12 in the estXY

To answer the last question involving the weights the findNN() function was adjusted to include the weights and output weighted locations, and the predXY() function was changed to accodate this change by using the sum function instead of the mean function as part of the mean was already done in the findNN() function:

```
findNN = function(newSignal, trainSubset, k = 3) {
   diffs = apply(trainSubset[ , 4:9], 1,
                 function(x) x - newSignal)
   dists = apply(diffs, 2, function(x) sqrt(sum(x^2)) )
```

```
   closest = order(dists)
   weights = sort(1/dists, decreasing = TRUE)
   # Weighting each location
   weighted_closests = trainSubset[closest, 2:3 ]*weights[1:k]/sum(weights[1:k])
   out = cbind(trainSubset[closest,1],weighted_closests)
   return(out)
}


predXY = function(newSignals, newAngles, trainData,
                  numAngles = 1, k = 5){

   closeXY = list(length = nrow(newSignals))

   for (i in 1:nrow(newSignals)) {
      trainSS = selectTrain(newAngles[i], trainData, m = numAngles)
      closeXY[[i]] =
         findNN(newSignal = as.numeric(newSignals[i, ]), trainSS, k = k)  # using findNN
   }

   estXY = lapply(closeXY,
                  function(x) sapply(x[ , 2:3],
                                     function(x) sum(x[1:k])))      # changed from mean(x[1:k])
   estXY = do.call("rbind", estXY)
   return(estXY)
}
```

Beyond these changes only minor changes were made to the code in the text to provide the outputs displayed in this report and to help explore the data more fully.

Please see attachment for entirety of code.