STAT 133, SPR 2015
Project 2
Part 1: Due Monday May 4
Whole Project Due Friday May 8

Predicting Location via a Statistical Indoor Positioning System

Introduction

The growth of wireless networking has generated commercial and research interests in statistical methods to reliably track people and things inside stores, hospitals, warehouses, and factories. With the proliferation of wireless local area networks (LANs), indoor positioning systems (IPS) can utilize WiFi signals detected from network access points to answer questions such as: where is a piece of equipment in a hospital? where am I? and who are my neighbors?

To build an indoor positioning system requires a reference set of data where the signal strength between a hand-held device such as a cellular phone or laptop and fixed access points (routers) are measured at known locations throughout the building. With these training data, we can build a model for the location of a device as a function of the strength of the signals between the device and each access point. Then we use this model to predict the location of a new unknown device based on the detected signals for the device. In this project, you will examine nearly one million measurements of signal strength recorded at 6 stationary WiFi access points (routers) within a building at the University of Mannheim and develop a statistical IPS.

The Data

Two data sets that we will use are available on the CRAWDAD site (A Community Resource for Archiving Wireless Data At Dartmouth). These are also available in the class bcourses site at Files/Data/offline.tar.gz and online.tar.gz. You can download them to your computer and uncompress them to files called offline.final.trace.txt and online.final.trace.txt, respectively. The offline data contains signal strengths measured using a hand-held device on a grid of 166 points spaced 1 meter apart in the hallways of one floor of a building at the University of Mannheim. The floor plan, which measures about 15 meters by 36 meters, is displayed in Figure 1. The grey circles on the plan mark the locations where the offline measurements were taken and the black squares mark 6 access points. These reference locations give us a calibration set of signal strengths for the building, and we use them to build our model to predict the locations of the hand-held device when its position is unknown.

In addition to the (x,y) coordinates of the hand-held device, the orientation of the device was also provided. Signal strengths were recorded at 8 orientations in 45 degree increments (i.e., 0, 45, 90, and so on). Further, 110 signal strength measurements were recorded to each of the 6 access points for every location-orientation combination.

In addition to the offline data, a second set of recordings, called the online data, is available for testing models for predicting location. In these data, 60 location and orientation pairs were chosen at random and 110 signals measured from them to each access point. The test locations are marked by black dots in Figure 1. In both the offline and online data some of these 110 signal strength values were not recorded. Additionally, measurements from other hand-held devices, e.g., phone or laptop, in the vicinity of the experimental unit appear in some offline records.

When we examine the files ourselves with a plain text editor, we find that each of the two files (offline and online) have the same basic format and start with something similar to

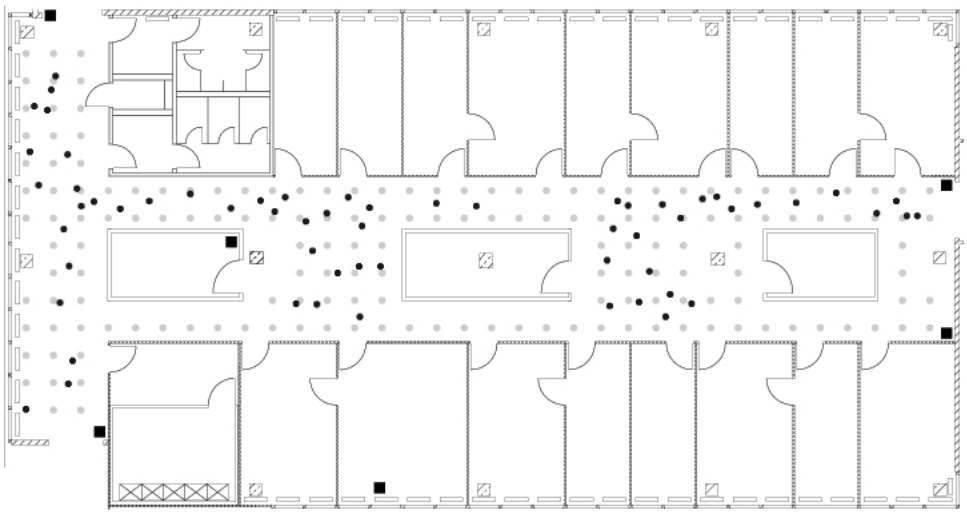

Figure 1: Floor Plan of the Test Environment

```
# timestamp=2006-02-11 08:31:58
# usec=250
# minReadings=110
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
```

Note that the fourth and subsequent lines displayed here are actually just one line in the text file,' but this one line has been formatted here on multiple lines for readability. We have added \ to indicate a continuation of the line.

The available documentation indicates that the format of the data is:

```
t="Timestamp";
id="MACofScanDevice";
pos="RealPosition";
degree="orientation";
MACofResponse1="SignalStrengthValue,Frequency,Mode"; ...
MACofResponseN="SignalStrengthValue,Frequency,Mode"
```

The units of the measurements are described below:

t	timestamp in milliseconds since midnight, January 1, 1970 UTC
id	MAC address of the scanning device
pos	the physical coordinate of the scanning device
degree	orientation of the user carrying the scanning device in degrees MAC
MAC	address of a responding peer (e.g., an access point or a device in adhoc mode) with the corresponding values for signal strength in dBm (Decibel-milliwatts), the channel frequency and its mode (access point = 3, device in adhoc mode = 1)

The MAC (media access control) variable refers to the MAC address of a hardware device, which is a unique identifier that allows a network card for a computer, access point, or other piece of equipment to be identified on a network. By convention, this identifier is written in the form mm:mm:mm:ss:ss:ss where mm and ss are 2 hexadecimal digits (0, 1, ..., 9, a, b, c, d, e, f). The first of these 3 sets of pairs of digits, i.e., mm:mm:mm, identifies the manufacturer of the equipment. The second set of 3 pairs (the ss) identifies the particular piece of equipment, both the model and the unique device.

The MACofResponse1 ... MACofResponseN in these data indicate that one line consists of a variable number of MAC address measurements. That is, these records are not of equal length, but form ragged arrays that depend on the number of signals detected. Additionally, there are more than 6 MAC addresses for some of these records. The extras come from other floors of the building or other devices.

Part I: Data Processing

Now that the data have been described. Your team's task is to process the data into a form that can be analyzed in R. There are two basic tasks in this work. The first is to get the data into R as a data frame of character variables. The second task is to clean and convert these variables and eliminate unwanted rows to create a data frame more suitable for statistical analysis.

Task 1 Your first task is to read the data into R and save it as a data frame, where one row/record in the input will populate n rows in the data frame. Here n is the number of MAC responses in that record.

Use readLines() to read each line of the offline data into R as a character string. Then, use regular expressions to split each lie up into fields, and create a set of 10 variables: `time`, `scanMac`, `posX`, `posY`, `posZ`, `orientation`, `mac`, `signal`, `channel`, `type`, where one record in the input file results in multiple records in the data frame, one for each MAC, with the information such as time, scanMAC, posX, etc. repeated for each of these records. Use the structure in the way that the data are recorded to create these variables. Wrap the code into a function that processes one record from the input file. Call this function processLine():

```
processLine = function(x)
{
  # x is a string  that corresponds to one line from the data file
  # returns  a character matrix with 10 columns
}
```

This data will be processed as follows. First the file will be read in.

```
txt = readLines("offline.final.trace.txt")
```

Then drop any lines that do not have information in them (e.g., the first three lines), and process the remaining lines with:

```
tmp = lapply(txt, processLine)
offline = as.data.frame(do.call("rbind", tmp))
names(offline) = c("time", "scanMac", "posX", "posY", "posZ",
                   "orientation", "mac", "signal", "channel", "type")
```

The return value `tmp` is a list of character matrices, one for each line in `txt`. The call to rbind() binds them all together into one large matrix, which we convert to a data frame and provide variable names. Notice that you are to order the columns in `tmp` in the same order as the names given for `offline`.

Task 2 In the next step of data preparation, you are to clean and process these character variables into formats that can be more easily analyzed. To do this:

- Convert data that should be numeric.
- Explore the data. Verify that the values look reasonable.
- Drop any irrelevant variables, i.e., variables that have the same values for all records or where the same information is captured in another variable.
- Round the values for orientation to the nearest 45 degrees, but keep the original values too.
- Drop all records that correspond to adhoc devices, and not the access points. There will still be about a dozen MAC addresses in the data. Use exploratory data analysis to figure out which are the 6 MAC addresses on the floor. According to the data documentation, these 6 include 5 Linksys/Cisco and one Lancom L-54g routers. You can look up the MAC addresses at `http://coffer.com/mac_find/` to find the vendors. This may prove helpful in narrowing down the MAC addresses to keep.

After you have processed your data, wrap this code up into a function called cleanData(). This function's signature is as follows:

```
cleanData = function(data, keepMacs = c("mm:mm:mm:ss:ss:ss", etc)) {
  # data is the output from the above processing, e.g., offline
  # keepMacs is a character vector of the 6 MAC addresses

  return(dataframe)
}
```

And we call the function as follows `offline2 =cleanData(offline)`. Confirm that this data frame matches the one that you produced incrementally.

Part II: Visualization

Although visualization is listed as part II of the analysis, it should be integrated throughout your data preparation and analysis. Use visualization to learn more about signal strength. For example, the signal strength has been measured multiple times at each access point for each location

and orientation. How do these measurements of signal strength behave? That is, what is the distribution of the repeated measurements at each location and orientation? Does signal strength behave similarly at all locations? Or does, the location, orientation, and access point affect this distribution?

Additionally, in a laboratory setting, signal strength decays linearly with log distance. In practice, physical characteristics of a building and human activity can add significant noise to signal strength measurements. How can we characterize the relationship between the signal strength and the distance from the device to the access point? How does the orientation affect this relationship? Is this relationship the same for all access points?

In part III, you will predict location based on signal strengths. Explore the effectiveness of this method through visualization.

In general, be creative in your plot making.

Part III: Nearest Neighbors, Cross-validation, and assessment

We will use nearest neighbors to predict location (x, y) based on signal strength. We naturally think of measuring the distance between two sets of signal strengths with Euclidean distance, i.e.,

$$\sqrt{(S_1^* - S_1)^2 + \cdots + (S_6^* - S_6)^2}$$

where S_i is the signal strength measured between the hand-held device and the i^{th} access point for a training observation taken at some specified location, and S_i^* is the signal measured between the same access point and our new point whose (x, y) values we are trying to predict. With k-nearest neighbors, we find the k closest training points (in the signal strength domain) and estimate the new observation's position by an aggregate of the (x, y) positions of the k training points.

One complication that you will need to address is that you have about 110 measurements to each access point. This will be the case when your system goes live as well.

Given that we are computing distances between vectors of 6 signal strengths, it may be helpful to organize the data in a different structure. Specifically, rather than a data frame with one column of signal strengths from all access points, let's organize the data so that we have 6 columns of signal strengths, i.e., one for each of the access points. The new data frame will not need to keep MAC address for the access points. It will have variables S1, S2, S3, S4, S5, S6, which correspond to the signal strengths for each of the 6 access points. Consequently, the number of rows in this data frame will be reduced by a factor of 6.

Use cross-validation to determine the best k to use. Exercise caution when you cross-validate because of the duplication of measurements at each location. You do not want to split the 110 measurements taken at a location across folds. Instead, you will want to keep them together, or you will want to summarize them into one record. You may find it useful to create a variable with unique labels for each location-orientation combination to help work with all the "duplicate" records.

To assess the predictive capability of your method for a particular value of k, we cannot use the notion of Type I and II errors because we are predicting a numeric pair (x, y). Instead, you will want to use a loss function of some kind that measures how far from the truth your prediction is. You might want to simply use Euclidean distance again, e.g.,

$$\sqrt{(x_{truth} - x_{pred})^2 + (y_{truth} - y_{pred})^2}$$

Or, you might want to dream up your own loss function that you think is more appropriate.

Finally, after you choose k, you are ready to apply your method to a true test set of data and assess how well it does. Online data are available in `online.final.trace.txt`. These observations form your test data. The test data represent 60 unique locations in the building. Measurements are taken at each of the 60 locations for only one angle, which for prediction purposes, you can assume is known. Roughly 110 signal strength are measured for each access point. You will need to read in this data and process it using the functions that you wrote in Tasks1 and 2. You will also want to then format the data as you did for the nearest neighbor method, i.e., create a data frame with six columns for the signal strengths to the six access points.

BY MAY 4: Your team is to turn in a .R file with two functions processLine() and cleanData() (see below). These functions must be well documented so that every decision that you made about transforming variables, dropping records, etc. is justified. Do not include any additional R code.

BY MAY 8: Your team is to turn in the final project. This project should include a file of code that performs the above tasks, and an .Rmd file (raw and knitted) that holds a report with graphs explaining your analysis.