Hand Gesture Recognition Via Classification

Yuyang He 116010068 Xinhai Hou 116010073 Ran Hu 116010078

12, Dec, 2017

1 Introduction

We have entered the era of artificial intelligence and big data. Computer programming can help us solve extremely complex problems and calculations in a few seconds. Our task is to find an optimal way of training the computer to recognize hand gestures from several points in a three-dimensional coordinate. There is a wide range of applications of hand gesture recognition, including virtual reality and games, robot control, and interactive sign language etc. By evaluating and comparing several methods for the classification and recognition of hand gestures, we finally find some preferable ways of dealing with unordered, extremely large data sets with a great number of missing values.

2 Data Description

The data is collected from 12 users performing 5 hand postures. There are 11 markers attached to a left-handed glove. But actually, each hand posture has different number of recorded markers. For example, the first posture has 7 markers in record, and the fifth posture has all the 11 markers in record. Due to the resolution of the capture volume and the orientation and configuration of the hand and fingers, many records have missing markers. Additionally, extraneous markers are also possible because of some external factors.

In the training and testing data sets, 5 classes specify the actual hand gesture of a certain record. 14 users are used to evaluate classification algorithms on a leave-one-user-out basis wherein each user is iteratively left out from training and used as a test set, which can be used to get the result of the training error. There are basically 36 predicters, which are 12 (Xi, Yi, Zi), $i \in 0,1,\cdots,11,12$ points in a three-dimensional coordinate after certain transformation from the original position. One row of the record has at least 3 markers and at most 12 markers. The markers are unlabeled and unordered.

By observing the data, we found that there is no correlation between different predictors. Besides, standardizing the data is meaningless. However, there are visible features of three coordinates: the z coordinate varies a little while the y coordinate of each posture has distinct features. Additionally, since we do not know why and how the markers are unordered, we can consider them as completely random data, or about the same as ordered data.

In conclusion, it is a large, unordered point sets with various sizes. Some latent features of this data

set are still unknown to us.

3 Ideas

3.1 Raw Data

The sequence between x1y1z1 and x2y2z2 is disordered. However, the instruction said that the order among the xyz is fixed. Besides, the coordinate of x, y and z will not miss separately. That means we can treat every observation as a group of points in the 3D space. That naturally arouse the thought that the point sets with same class may have special patterns and they may gather at some area. If it does, then given a new point set we can scale the similarity between the most like point of set and predict the class as this class. Actually, this process is the core idea of KNN. In KNN we describe the difference by Euclidean distance between two point in three dimension. How to describe the distance from one point set to another?

When talking about the similarity between one cluster to another cluster, we use linkage. This approach doesn' t care about the amount or order of every point in one cluster, which satisfy the condition in this problem. Besides the linkage, when talking about the similarity between one cluster to another, we also consider the type of distance in the example. By refering to resource of the type of distance, we discover several types like Euclidean, Manhattan, Chebyshev etc.

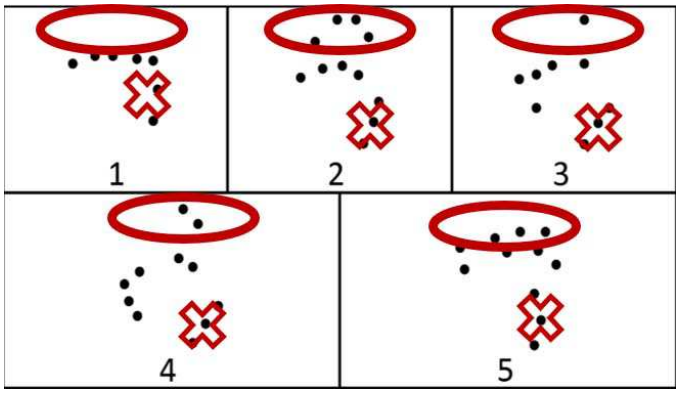
After defining the similarity between the points set, we can easily use the idea of KNN to predict the class of each observation. We first scale the similarity between each points set without class in the test data set and every known points set with class in training data set. Then choose the most similar K neighbors. And predict the class as the one with most frequences in the K neighbors.

However, the similarity we defined is quite different from the ordinary KNN, and knn function in R can not be applied for this approach. So we have to write by ourselves. But the time complexity is too high. If we use 100% training data and test data, we will spend a month on the Chengdao computer. So here we have to choose a compromised method. We randomly choose the 800 (1.3%) observation as training data and choose 10 observation as the test data 10 times to train the model in order to get the optimal linkage and K and distance type.

3.2 Characteristic Data

When we got the data, we tried to find out some features to do the classification. According to observation of the gestures, differences only come from the red circle parts, and the red cross parts are almost the same. Thus, we only consider points eliminating the lowest three Ys, which are likely to be the red

cross parts. Following is the relative pattern.



Gestures

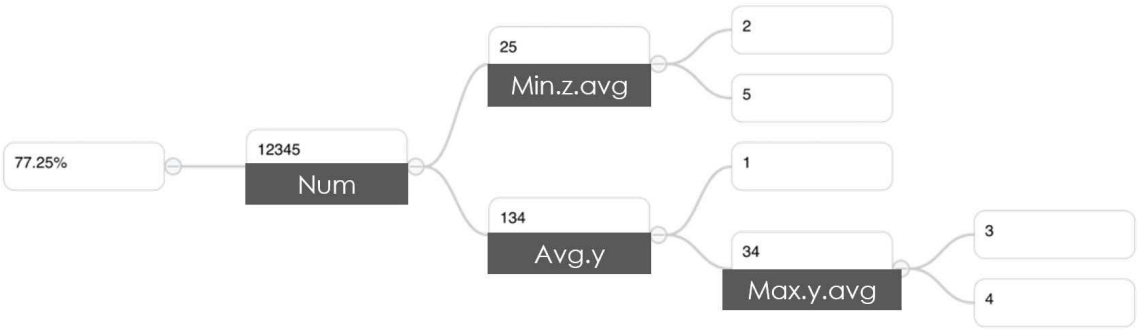
After elimination, we try to calculate some features as follows.

class	num	avg_x	avg_y	avg_z	max.y.avg	min.z.avg
1	2.607084	27.68335	81.80801	-44.7143	84.47017	-52.2351
2	7.532449	41.77016	115.2038	9.362989	147.2081	-13.9774
3	3.243764	42.50733	105.6333	-17.6998	133.5251	-29.0337
4	4.36556	43.54641	115.0988	-18.732	147.267	-30.1922
5	6.681618	30.80193	112.691	-16.985	136.3685	-50.9588

Features

We find out that the differences among those 5 classes in one feature are not so large, but different classes seem to perform differently in different features. Thus, we are inspired by the idea of tree method. After careful analysis of the gestures and positions of marks, we decide to use step tree classification method, depending on the features.

The followings is our mind mapping.



Mind Mapping

3.3 Pruned Data

Because we cannot perform normal KNN, support vector machine and other classification methods on the raw data set, we need to manipulate the data to be a new one with equal sizes and no missing values.

Our first idea is not pruning the data, but filling missing values. There are many different ways of padding the data through some fancy packages. However, considering that there is little relationship between the missing values and their neighbors, we thought that it may be hard for us to find an optimal way of filling the data. Therefore, we come up with the idea of pruning the data. Here, we provided two kinds of pruned data.

The first one is: making the training data to be a 15,000*16 matrix and the test data to be a 3,000*16 matrix. We take out the "User" for testing convenience. Later, we can include "User" to get the training error rate. The first row of the new data is "Class", and the remained 15 columns are the (X, Y, Z) coordinates of 5 markers. Why do we choose exactly 5 markers? Through observation, we found that most rows of the first class have 5 markers, the second class with 11 markers, the third with 5 markers, the forth with 6 markers and the fifth with 11 markers. To make a new matrix with the largest size, we choose 5 markers to be the predictors. Then the pruning process begins. First, eliminate different classes' rest columns respectively. For instance, eliminate the last 21 columns of the first class and the last 3 columns of the fifth class. Then, eliminate the new data' s rows which have missing values. The next step is, for each row with more than 15 columns, we randomly choose 5 points. Now, every class have 15 predictors. We further randomly choose the subset of each class with 3,000 records.

The second pruned data set is: cutting the training data to be a 62,824*10 matrix and the test data to be a 12,000*10 matrix. The "User" is also took out for convenience. This idea is simple. Because each row has at least 3 markers, we remain the first 9 predictors and eliminate the rest. Finally, we can get all the records with only 9 predictors.

There are both advantages and disadvantages of these two ways of getting the pruned data. The first one has more observations and is under the precondition that all the markers are totally unordered. However, we cannot use it to manipulate test data, since there are records with less than 5 markers that we cannot predict. The second one is small and easy to get. It is under the precondition that we consider these markers are not totally unordered, but most of them have certain order. We tend to use the second pruned data, since it can be used to predict all the observations. And we also use the first pruned data to make comparison with the second one.

4 Methods&Results

4.1 Renly' s Approach Using Raw Data

In order to simplify the time complexity, we construct a point.mat function in advance with k rows and 3 colomns where k is the number of point in the each point set. Function distance.type describes the similarity between two point sets. It will return a matrix with m rows and n colomns. Entry with coordinate i and j represents the distance from the ith point in set1 to the jth point tn set2. In order to get the different linkage, we just need to use function mean(), max() and min(). Centroid will be discussed later. Another parameter "type" represents the definition of distance with Euclidean, Manhattan, Chebyshev, Correlation, Cosine. Since the time complxity of this program is quite high, we choose sample to represent the population. Here we choose 800 observation from training data as the sample. We take the last 10 observations as the test data and rest of them as the training data. For each distance type and linkage, we find the optimal K is 10. Each time we use the random sample 10 times and take the mean of error rate as the error rate for the certain K. For the centroid part, we certralize each observation with one coordinate (x, y, z). Then take x, y and z as the predictor using function knn() to construct model.

Results of the optimal K is attached. Among the model, Linkage with Single and distance with "Manhattan", "Euclidean" and "Chebyshev" all performed well in train data with error rate about 9%. Considering the occasion due to random sample, we choose the best 4 models with lowest training error. And the test error for these four model we also use the random process. We use 15000 observations for training and 200 for testing randomly 5 times to approxiame the population. The results are showing below:

type	linkage	distance	k	Testing Error rate
1	Single	Manhattan	2	4.3%
2	Single	Euclidean	1	3.7%
3	Single	Chebyshev	1	4.5%

4.2 The Idea of Classification Tree Using Characteristic Data

4.2.1 Feature Explanation

Num: the number of known points.

Avg_x: the average x position of known points from x0 to x11.

Avg_y: the average y position of known points from y0 to y11.

Avg_z: the average z position of known points from z0 to z11.

Max.y.avg: the average of the maximum y position of known points from y0 to y11.

Min.z.avg: the average of the minimum z position of known points from z0 to z11.

4.2.2 First Step

We use the **num** feature to separate 134 and 25. We count the number of known points only including x0 to x11 (or y0 to y11, or z0 to z11) for each training data after elimination, and then take the average for each gesture. It shows that there are **more than 90%** accuracy to separate 134 and 25 in both training data and test data. The total accuracy is **43.5%** and **43.8%** for training data and test data, respectively.

4.2.3 Second Step

25 Classification

For this part, we consider the variable **min.z.avg** as the criteria. If the min.z.avg is less than a boundary, trian.pred will be assigned 5, else 2. This boundary is decided by the training data set. From the outcome, we can see the accuracy of classifing 2 and 5 is **91.77%**. The test accuracy of classifing 2 and 5 is **93.9%**. The final accuracy is **52%** for both training data and test data.

134 Classification

For this part, we consider the variable **avg.y** to separate 1 from 34. Boundaries are also decided by the features in training data.

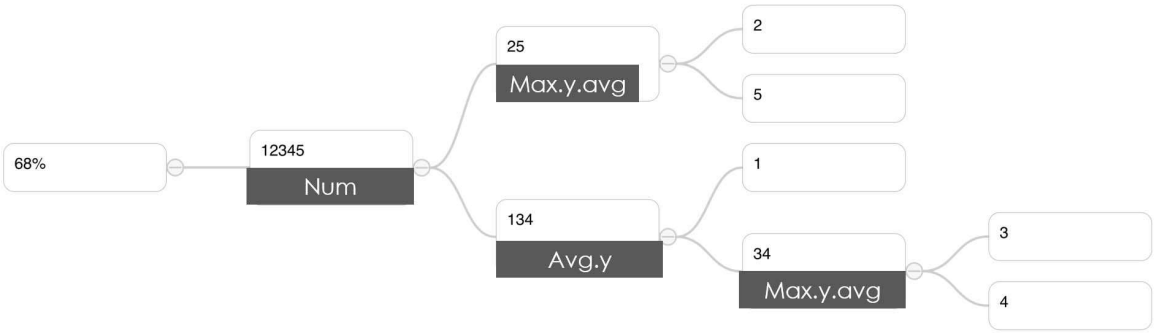
4.2.4 Third Step

34 Classification

We consider **max.y.avg** to separate 3 and 4. It only gives **80.47%** accuracy in the training data and **80.69%** in the test data, which leads to the compared low accuracies in final. Training accuracy is **76.3%**, and the test accuracy is **76.6%**.

4.3 Other Versions

4.3.1 Version One

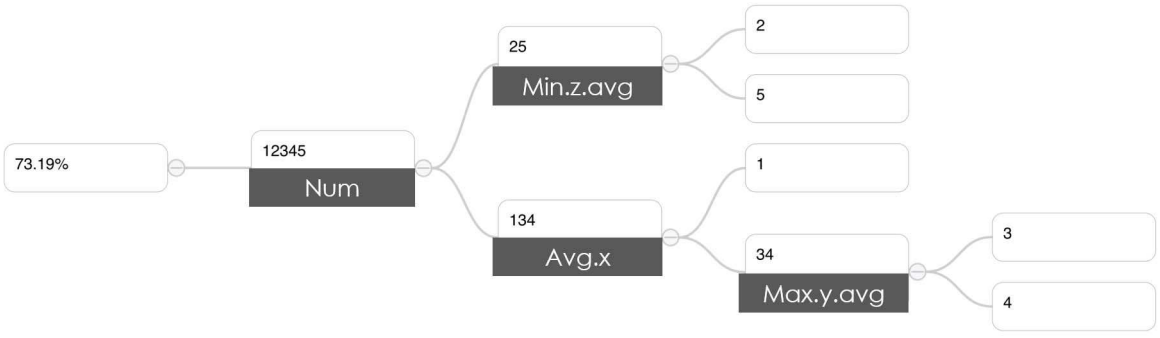


Mind Mapping 1

If we change the the criteria of the classification of 2 and 5 to max.y.avg in step 3, we will get the following results.

For the training part, we get the accuracy of 65.6% to classify 2 and 5. And finally get the total accuracy of 68.3%. For the test part, we get the accuracy of 65.3% to classify 2 and 5. And finally we get the total accuracy of 68.4%.

4.3.2 Version Two



Mind Mapping 2

If we change the criteria of separation 1 with 34 to avg.x in step 3, we will get the following results.

For the training part, the accuracy of classifying 1, 3 and 4 is 70.23%. The final accuracy is 70.5%. For the test part, accuracy of 71.35% is reached. The final accuracy is 71%.

4.3.3 Unmentioned Versions

Actually, we have tried to change another 3 criteria aiming to better separate 2 and 5, or 3 and 4, or 1 from 34, but the outcomes are not good as what we have discussed above. Here we just skip the explanation.

4.4 SVM, LDA, QDA, and KNN Using Pruned Data

Because we have the pruned data with equal sizes. We can perform many classification methods on it.

4.4.1 SVM

We use the second data set with 3 markers. Both kernel = "linear" and kernel = "radial" are used and we found that the linear one is better. And we tried to use the cross-validation to find the optimal cost, but the time complexity is too high. So we tried a few cost values and found that cost = 10 is the best. Unfortunately, because the training data is too large, we failed to get the result of the training error on the whole training data. Instead, we randomly chose 9999 records and get the training error rate 55.95%, which is pretty bad. But the test error rate of the whole test set is much better, say, 27.075%. Furthermore, in order to reduce the time of testing, we took one fourth of the training data. Not only do the training speed become much faster, but the error rate is only a little higher – 30.975%.

4.4.2 LDA & QDA

Still, we first use the second pruned data with 3 markers. LDA' s training error is 29.78%. LDA' s test error is 48.43%, which is high. QDA' s training error is 7.66%, which is pretty good, but its test error is a little high – 36.31%. These two methods may have the problem of overfitting.

4.4.3 KNN

By changing the value of k, we found k=1 is the best for prediction. Again, we took out user 6 as our testing user. The training error is 33.07%. The test error of the second pruned data is 3.925%, which is the best among all the methods. Because the test result on the second data set is pretty good, we also use the first pruned data to make comparison. And we found that, although the number of predictors increase, the test error is higher – 17.8%. Based on this strange result, we guess the data may have some certain order rather than totally random.

5 Conclusion&Summary

We have tried and compared numerous classification methods on this hand gesture data set. Some approaches are classical and some are home-made.

First, Renly' s approach gives various ways to solve this problem. With the different combinations of linkage and distance types, these method has infinite potential. However, it also requires much on time to train these models. Therefore, the future work of Renly' s approach is to find more effective algorithms and data structure make it possible and realistic for bigger data.

Second, classification tree method is inspired by the idea of tree, and uses different boundary conditions in each step. Because of the elimination of lowest three Ys, it loses some information from the data. In addition, the accuracy after each step is the conditional probability, and to get the final total accuracy we need to multiply them together. Thus, it does not give so high accuracy.

Third, when there is a great number of missing values, we can prune the data and use the limited information to make predictions. Even though the precondition is that all the points are unordered, we can still assume that they have certain order, and try to use some methods that we have learnt.

All the methods presented in this project have advantages and disadvantages. They can be applied to different kinds of data set and make comparatively accurate predictions. We believe that more secrets and inner features of a data set need to be explored. Only then can we use our knowledge and innovative ideas, and become excellent statisticians.

6 References

Andrew Gardner, C. A. D., Jinko Kanno, Rastko Selmic (Oct 2014). "3D hand posture recognition from small unlabeled point sets." 2014 IEEE International Conference on Systems, Man and Cybernetics (SMC): 164-169.

Andrew Gardner, J. K., Christian A. Duncan, Rastko Selmic (June 2014). "Measuring distance between unordered sets of different sizes." 2014 IEEE Conference on Computer Vision and Pattern Recognition(CVPR): 137-143

Gardner, A. (2017). "Datasets for Motion-Capture-Based Hand Gesture Recognition." from `http://120.198.248.33/cache/www2.latech.edu/~jkanno/datadescription-1.pdf?ich_args2=888-11101106009497_23f373d5d9e8155883995a61e294432d_10001002_9c89652edec6f9d29738518939a83798_af4cf71ee3d8eb7a02c1636c26264d58.`