ML4QS Assignment 2

Haohui Zhang[1 [2722930]], Yilin Li[1 [2737659]], and Yiming Xu[1 [2696855]]

Vrije Universiteit, Amsterdam, Netherlands

1 Theoretical Part

C5Q2. Dynamic Time Warping (DTW) aims to measure the similarity between two time series series usually using the Euclidean distance. Although this metric is widely used, it has a weakness in sensitivity to distortion in the time axis [2]. If one time series has a lot of noise and distortion, for example, if a person speaks at a very noisy place while the other does not, using DTW to find the similarity between these two speech data will perform very badly. The lower bounding distance metrics LB_Kim and LB_Keogh proposed in [3] can limit the maximum difference in the time points of pairs and improve the DTW by remarkable speedup.

C5Q7. Subspace clustering algorithms can localize searches of relevant dimensions, allowing them to find clusters that exist in multiple potentially overlapping subspaces [4]. Consider a lymphoma microarray dataset with 4000 features, where each feature is the expression level of a specific gene. Understanding the differences between cancer subtypes at the genetic level is critical to discovering which treatments are most likely to be effective. However, the common clustering approach cannot handle such high-dimensional data, while, subspace clustering approaches can help to uncover patterns.

C6Q1. Best fitting function is fitted based on the training dataset. Whether it is a good function is determined by the test dataset based on its generalizability. If we only care about the training data, the model will easily overfit, and the out-of-sample error is very likely to increase. Therefore, a good function with the smallest in-sample error in the validation data will be chosen at last.

C6Q7. The ROC curve is affected by Θ_{cut}. The range of $(1+e^{-\theta^T x_j})^{-1}$ is (0,1). When $\Theta_{cut} = 0$, all samples will be assigned to class 1, therefore, the TP=1 and FP=1 which is the point on the upper right in the left panel of Fig.6.4. In contrary, when $\Theta_{cut} = 1$, all samples will be assigned to class 0, and the position of it in the figure is the lower left corner. In between, the TP rate and the FP rate increase with the decrease in the threshold Θ_{cut}. Because FP will result in cost, the minimum of the cost is always caused by a low FP rate and a high TP rate. When the minimum cost estimate is $\Theta_{cut} = 0$, the ROC would be an extremely curved line that nearly goes straight to the upper left corner. When the minimum of the cost estimate lies at $\Theta_{cut} = 1$, the ROC would be a extremely curved line nearly going straight to the lower right corner.

C7Q3. Traditionally, neural networks have only three types of layers: hidden layers, input layers, and output layers. The number of neurons in the input layer equals to the number of input variables. The number of neurons in the output

layer is equivalent to the number of outputs associated with each input. The hidden layers are required if and only if the data must be separated non-linearly. As for the numbers of the neurons in hidden layers, there is a guideline in [6]: The number of neurons should be between the size of the input layer and the output layer, and less than twice the size of the input layer. Or the number should be 2/3 the size of the input layer plus the size of the output layer.

C7Q6. Kernel functions in Support vector machines (SVMs) can help solve problems. 1) Linear Kernel Function: is one-dimensional and the most basic form of kernel in SVM. 2) Gaussian RBF Kernel Function: the radial basis function, and is used when there is no prior knowledge about the data. 3) Polynomial Kernel Function: is a general representation for a kernel of degree greater than 1 [7].

C7Q8. Classifiers that rely on pairwise distances between points, such as KNN, suffer from a problem known as the "curse of dimensionality". As the dimension of the dataset increases, the density of the high-dimensional space occupied by the training data becomes smaller, the neighbors may be so far apart, and we need to search a large amount of space to find the neighbors. The pairwise distance between the points increases as we add additional dimensions [5]. Therefore, KNN is not suitable for large dimensional datasets and performs best with a low number of features. As for model-based approach, a large number of features may affect model performance without causing a sharp drop.

C7Q13. Under the same number of samples N, the more features, the easier it is to overfitting. The reason is that the more features, the larger the hypothesis set, which will cause the $P(E_{in\_sample}(h) > E_{out\_of\_sample}(h) > \epsilon)$ to increase. What's more, when the noise data in the sample interfere too much, the model over-remembers the features of the noise which will lead to overfitting. If we apply effective feature selection and remove junk features, the redundant data will become less which declines the opportunity to make decisions based on noise. Besides, removing irrelevant features tends to reduce the difficulty of learning.

C8Q5. The " no free lunch theorem" states that all optimization algorithms perform equally well when the performance of all possible problems is averaged, which implies that there is no single best optimization algorithm [9]. In our context, there can be a lot of strategies to initialize the random reservoir, however there is no single strategy that has the best performance in all domains. The strategy should be selected depending on the dataset and domain.

C8Q6. We suppose the tracked data by the quantified self is a time series data, we can directly use it to train the RNN, or we can regard each time point is a isolated instance and use them to train the feed forward network. However, if the task aims to predict the future like predicting the mood, the athletic performance, etc., the notion of time will contain very valuable information. To take the temporal aspects into account more naturally, we need a temporal learning method like RNN rather than a feed forward network.

C8Q8. Three algorithms are introduced in the book to optimize parameters of the dynamical systems model, however none of them can guarantee to find the optimal parameter especially in a complex environment. All of them can easily

find a local optimum. However, in terms of evolutionary algorithms, a remarkable fact is that each algorithm emphasizes different characteristics that are most important for a successful evolutionary process. Therefore, it is impossible for these algorithms to locate the global optimum in a different environment and different datasets. This is proved in the experiments in [8]

C9Q2. Remember the *Markov Property* below, the reward is normally determined by the current state of the user, the action taken.

$$\mathbf{Pr}\{R_{t=1} = r, S_{t+1} = s' | S_t, A_t\} \tag{1}$$

Define our task is finite MDP, the expected reward is expressed:

$$r(s, a, s') = \mathbb{E}[R_{t+1} | S_{t+1} = s', S_t = s, A_t = a] \tag{2}$$

In this task, the goal is to make user more active, so if taking an action or activity a, the user's state (from s to s') become active, the immediate reward r will be assigned a positive numerical value, otherwise, the r will be assigned a negative value. When the level of active increases, r also increases. If the task is a deterministic process, the reward equals to the immediate expected reward. If the task is a stochastic process, which means take a in s leads to various s' based on *transition probability* $p(s'|s, a)$ and various r based on a probability distribution, and the reward will be the expected reward multiply transition probability:

$$R(s, a) = \sum_{s'} p(s'|s, a) r(s, a, s') \tag{3}$$

C9Q4. The Markov property is defined in 1, when both probabilities are equal for all rewards r and states s over all time points. Applying rl, the quantified self is regarded as the environment, and the agent is a kind of software which can observe the state of the self, provide an action to the self and observe the next state. However, in the quantified self setting, the individual should self-track any kind of biological, physical, behavioral, or environmental information [1]. Therefore, the amount of the sensory data may be too large and have too many variation, and some biological attributes like heart rate have delay and uncertainty, which means it is almost impossible for the agent to evaluate the probability of moving to the next state. Also, because the state may have too much variation, the data is self-tracked and the user cannot follow the action provided by the agent at all time point, the corresponding relations between s, a and s' cannot be found.

2 Practical Part

2.1 C5Q1: Clustering Accelerometer and Gyroscope Data

We have already clustered the phone's accelerometer data, so we clusterd the gyroscope sensor data and compared the results with the accelerometer data to determine the effect of clustering on other sensors' data.

The difference between Accelerometer and Gyroscope sensor data in K-means and Hierarchical clustering is illustrated in Figure **??**. The upper portion of the image represents the results of the accelerometer data, while the lower portion represents the gyroscope data. When the k value for the K-means algorithm is less than six, the silhouette score of the Accelerometer data continues to increase as the K value increases, whereas when k is greater than six, the silhouette score gradually decreases. The silhouette score of the Gyroscope data varies almost insignificantly as k increases, and reaches its maximum value when k equals 5. Consequently, the K-means algorithm assigns six clusters to Accelerometer data and five clusters to Gyroscope data.

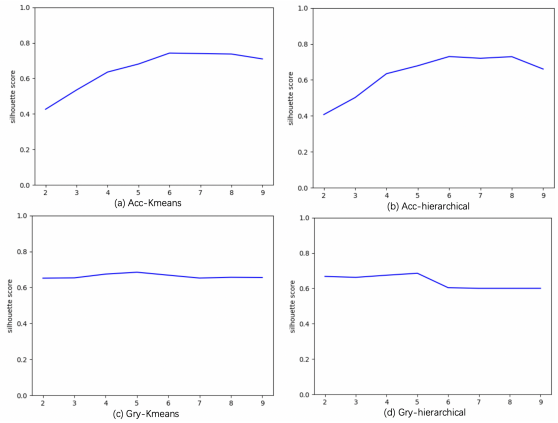

Fig. 1. Comparison of results on k-means and hierarchical clustering of gyroscope data and accelerometer data in the original data.

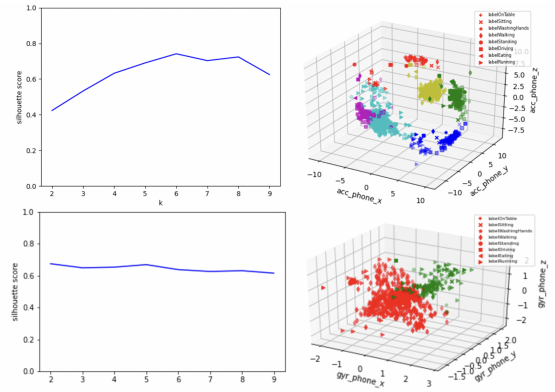

Fig. 2. Comparison of the results on K-medoids and the 3D distribution of gyroscope data and accelerometer data in the original data.

Comparing the results of the Hierarchical algorithm and the K-means algorithm yields comparable results. For the Accelerometer data, the results of the K-menas algorithm and the Hierarchical algorithm are very similar; the silhouette values increase as the K value increases, reaching a maximum when k equals 6, and then decreasing. Similar to K-menas, we obtained the best results for Gyroscope data when k equals 5.

Observing the 3D plot in Fig.1 reveals that the 3D distribution of the Accelerometer data presents a circular ring, whereas we do not observe this phenomenon in the distribution of the Gyroscope data. We hypothesize that this difference is due to the fact that different activities have significantly different accelerations, but for these activities, the gyroscope data changes differently, resulting in more concentrated data.

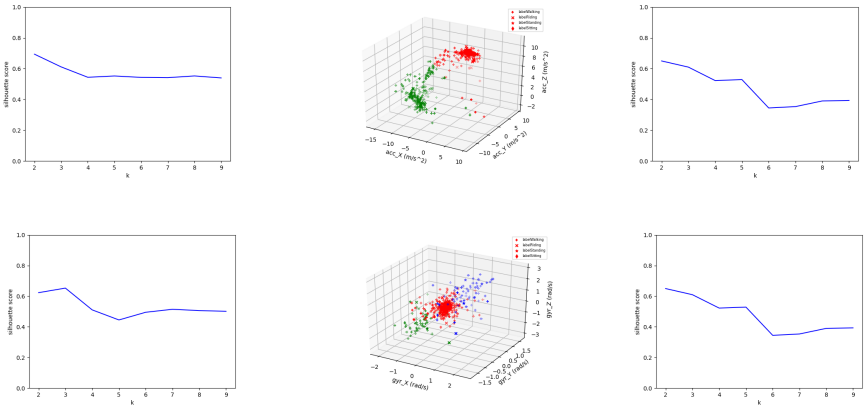

Fig. 3. Comparison of our collected Acceleration, Gyroscope data on three clustering methods, k-means, k-medoids, and hierarchical clustering.

2.2 C5Q2: Clustering on Owned Dataset

Next we focus on clustering the data we collected, we choose Acceleration, Gyroscope, and observe how these features perform on three methods.

As shown in Figure 3 K-means and K-medoids methods classified the collected Accelerometer data into three clusters. When K = 3, these two methods receive the highest score; as k increases, the score value decreases. The K-means and K-medoids methods achieved the highest score for the Gyroscope data at k=2, so the data were divided into two clusters. For the 3D distribution, we observed a similar structure to that of the crowdsourced data: the distribution of the Gyroscope is concentrated in the center, whereas the distribution of the Accelerometer is concentrated in the outer periphery, with no data distribution in the center. We believe this phenomenon confirms hypothesis on Section 2.1 .

2.3 C7Q1: Activity Recognition with Different Labeling Methods

Basically, the case which contains multiple labels or an unknown label will be dropped in order to ensure the performance of train. This is because if a set of features corresponding to multiple labels is used as input for training, it will average the typical parameters of activities and affect the performance of classification[10]. In this section, we intended to verify whether those multi-label or unknown-label case will impact the performance or not. Since it does not make sense to group cases with multiple labels into any one category of label (e.g., random assignment is undesirable because it would be treated as noise and then bring a negative impact on the classification), we believed that giving those cases an "Undefined" as the label of classification is a reasonable method. The feature matrix is expressed as [2883 rows * 535 columns]. Analysis of the confusion matrix (Right panel in Fig.4) shows that many cases with valid labels are classified as "Undefined" (e.g., up to 66.7% of data in "labelWashingHands" are misclassified as "Undefined "). This indicates that the inclusion of the "Undefined" dataset affects the classification model (i.e., Latent variable[11] or core parameters in various models, such as A^T, θ and etc.). Besides, Table.1 shows that the accuracy of classification after adding "Undefined" is significantly lower than that of normal experiments, regardless of whether NN, RF, DT or NB is used. The Naive Bayesian is most significantly affected, with a decrease of about 15% in accuracy for all results with different combinations of features. In conclusion, we summarized that adding multi-label or unknown-label cases can negatively affect the accuracy of most algorithms. All details are shown on Fig.4 & Table.1.

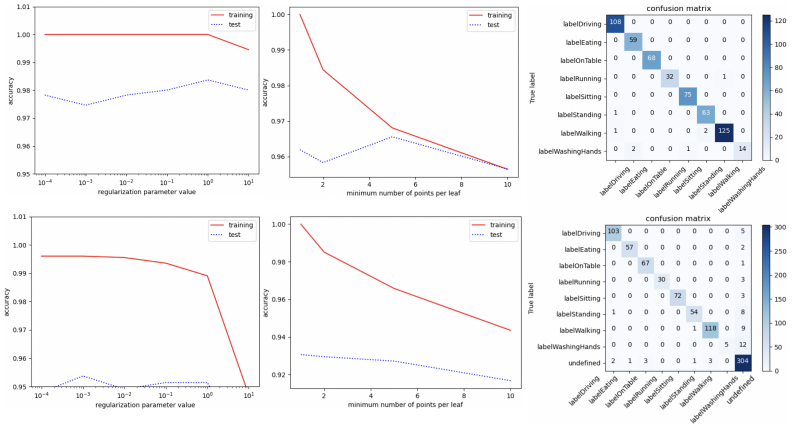

Fig. 4. Overview of comparison of classification. Left and middle panel: the regularization parameter value and growth of minimum number of points per leaf illustrates that the accuracy of "Undefined" is always lower than the previous experiment over the increase of leaves. Right panel: Confusion Matrix. "Undefined" is attributed to "Walking" five times and "Walking" is categories as "Undefined" nine times.

2.4 C7Q3: Implementing Activity Recognition on Owned Dataset.

According to the result in Table.1, the difference in accuracy between Crowdsignals and Owned dataset for activity recognition is not significant (even with different algorithms and feature sets). We believed that the activities in Owned dataset switch less frequently and last for a long time. Hence, the dataset has fewer noises and the results of feature extraction are more accurate (e.g., the Time Domain Metric in Assignment 1 can fit the pattern of activities well). The analysis of the classification results demonstrates that Owned dataset has higher accuracy on all algorithms. Next, considering that Random Forest (RF) generalizes better to small-scale datasets and is good at handling high-dimensional data (which fits the characteristics of our dataset), we believe it is reasonable to use RF for better accuracy[13]. We also found that the relevance of the features selected from the Crowdsignals data is stronger. This is because the Naive Bayesian algorithm does not perform well on the Crowdsignals dataset, and the most influential factor for it is the correlation between the different features[12]. This means that the selected features (Selected Set on Table.1) based on Crowdsignals are in-comprehensive and does not fully characterize the data. Details of result and comparison are shown on Fig.5 & Table.1.

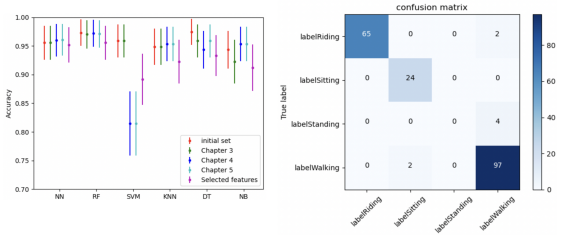

Fig. 5. Overview of classification. Left panel: Performance on various methods. Right panel: Confusion Matrix. High value on the diagonal shows the high accuracy.

2.5 C8Q3: Development of Dynamical Model for Prediction

In this section, we intended to implement a multi-objective genetic algorithm to predict the heart rate based on certain features of the Crowdsignals dataset. The whole process consists of two phases. 1) Feature selection: After implementing Pearson Correlation analyze on the set of features, we proposed to select `Acc_X, Acc_Y` and `Heart_rate` as the features to design objective function. This is because `Acc_x` and `Acc_Y` will affect the activity intensity of the subject and thus will influence the heart rate. 2) Design of multi-objective function: After selecting the valid features, NSGA-II[15], which is an advanced genetic algorithm, is imported to find a comprehensive spread of solutions and better convergence near the Pareto front. In detail, according to the principle of Markov Chain[14], the prediction of the heart rate can be represented as finding the optimized value of current state based on the parameters on previous state. Hence, the

value of heart rate can be contained in the objective function as input. Specifically, the top result was calculated using Formula $\{Acc\_X_n = Heart\_rate_{n-1} * Acc\_X_{n-1}; Acc\_Y_n = a * Heart\_rate_{n-1}; Heart\_rate_n = b * Heart\_rate_{n-1}\}$, which contains the value of `Heart_rate` in calculation. Furthermore, the bottom one utilized the Formula without the `Heart_rate`

Table 1. Performance in various machine learning algorithms. The result will be expressed as "Accuracy" ([0, 1]). Initial Set, Chatper 3-5 and selected set means different set of features will be collected and treated as input. *: Accuracy is lower than 0.700. [1]: Dataset dropped "Undefined"; [2]: Dataset included "Undefined"; [3]: Owned Dataset.

Type of Dataset and Method	Initial Set	Chatper 3	Chatper 4	Chatper 5	Selected Set
Neural Network[1]	0.966	0.970	0.973	0.971	0.874
Random Forest[1]	0.964	0.962	0.993	0.992	0.983
Naive Bayesian[1]	0.909	0.907	0.910	0.900	0.808
Neural Network[2]	0.931	0.932	0.934	0.935	0.700
Random Forest[2]	0.947	0.948	0.959	0.959	0.933
Naive Bayesian[2]	0.733	0.748	0.739	0.739	*
Neural Network[3]	0.954	0.953	0.955	0.955	0.949
Random Forest[3]	0.968	0.967	0.968	0.967	0.952
Naive Bayesian[3]	0.941	0.918	0.955	0.955	0.906

According to the results shown in Fig.6, the inclusion of optimization objectives (i.e., `Heart_rate`) into parameters can improve the accuracy of prediction. The top panel reflects that NSGA-II can fit the data set well. Meanwhile, by observing the curves in the bottom panel, we can conclude that the higher the correlation between parameters and labels of prediction, the higher the accuracy.

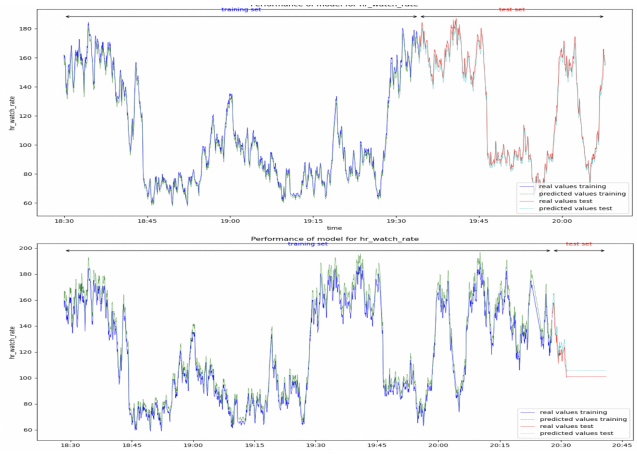

Fig. 6. Overview of comparison of prediction with various objective functions.

References

1. Hoogendoorn, M., & Funk, B. (2018). Machine learning for the quantified self. On the art of learning from sensory data.
2. Ratanamahatana, Chotirat A.: Everything you know about dynamic time warping is wrong. In: Third workshop on mining temporal and sequential data. vol. 32, (2004)
3. Keogh E., Ratanamahatana C.: Exact indexing of dynamic time warping. In: Knowledge and information systems. vol. 7 (3), pp. 358—386. (2005)
4. Parsons L., Haque E., Liu H.: Subspace clustering for high dimensional data: a review. In: Acm sigkdd explorations newsletter. vol. 6, pp. 90–105. (2004).
5. Farhan R.: k-Nearest Neighbors and the Curse of Dimensionality, https://towardsdatascience.com/k-nearest-neighbors-and-the-curse-of-dimensionality-7d64634015d9. Last accessed 19 June 2022.
6. Jeff H.: The Number of Hidden Layers, https://www.heatonresearch.com/2017/06/01/hidden-layers.html. Last accessed 19 June 2022.
7. TechVidvan: SVM Kernel Functions, https://techvidvan.com/tutorials/svm-kernel-functions/#: :text=A%20kernel%20is%20a%20function,number%20of%20dimension. Last accessed 19 June 2022.
8. Bäck T., Schwefel H.: An Overview of Evolutionary Algorithms for Parameter Optimization. In: Evolutionary Computation. vol. 1, pp. 1–23. (1993). https://doi.org/10.1162/evco.1993.1.1.1
9. Wikipedia, No free lunch theorem, https://en.wikipedia.org/wiki/No_free_lunch_theorem. Last accessed 19 June 2022.
10. Gupta, S., & Gupta, A. (2019). Dealing with noise problem in machine learning data-sets: A systematic review. Procedia Computer Science, 161, 466-474.
11. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.
12. Leung, K. M. (2007). Naive bayesian classifier. Polytechnic University Department of Computer Science/Finance and Risk Engineering, 2007, 123-156.
13. Biau, G., & Scornet, E. (2016). A random forest guided tour. Test, 25(2), 197-227.
14. Geyer, C. J. (1992). Practical markov chain monte carlo. Statistical science, 473-483.
15. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.