ML4QS Assignment 2 - Group 54

Hong Huo$^{1[2726433]}$, Matteo De Rizzo$^{2[2749303, and]}$ Seth van der Bijl$^{3[2649279]}$

^{1} Vrije Universiteit Amsterdam, 1081 HV Amsterdam, Netherlands
^{2} h2.huo@student.vu.nl
m.de.rizzo@student.vu.nl
seth.vander.bijl@student.vu.nl

1 Theory

1.1 Chapter 5

Ex. 2
The datasets may have different number of points. Only series of equal length can be used with Euclidean Distance. As a result, when points are lacking, ED cannot be computed (unless also cutting the other sequence, thus loosing more information). Euclidean Distance does not allow time-shifting or time-warping opposed to all algorithms which are based on dynamic time warping. An alternative can be cross correlation between the two time series. We can use Pearson correlation coefficient, and handle shifts in the patterns using a lag.

Ex. 7
A dataset with sound recording features and periodic distribution. As it is unknown which sound events to expect in an audio recording, many generic audio features are utilized. This result in high dimensional audio data. Clustering this data is challenging because of two reasons. First, high dimensional data suffers from the curse of dimensionality. Second, the sound events reside in a different combination of relevant features[4]. Besides, periodic sound events are better clustered when combined with certain features detecting this periodicity while non-periodic events could be clustered with other features.

1.2 Chapter 6

Ex. 1
In order to get best fitting function, we minimize the in-sample error with respect to the training data and obtain the best fitting model h_d . For each h_d we then calculate the in-sample error with respect to the validation data. Finally, we choose h_d with smallest error in the validation data as our best model. A good fit is when both the training data error and the test data are minimal. As the algorithm learns, the mistake in the training data for the modal is decreasing over time, and so is the error on the test dataset. If we train for too long, the training dataset performance may continue to decline due to the model being overfitting and learning the irrelevant detail and noise in the training dataset.

As a result, the separation of the test data from training or validation data is essential. Best fitting function is not necessarily a good function because the predictions on the test dataset are not necessarily good.

Ex. 7
In a ROC curve the true positive rate is plotted in function of the false positive rate for different cut-off points. Each point on the ROC curve represents a False Positive Rate/True Positive Rate pair corresponding to a particular decision threshold. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (1.0 False Positive Rate, 1.0 True Positive Rate)[2]. Therefore the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test. When the cost estimate lies at either cut = 0 or 1, the cost estimate is the highest.

1.3 Chapter 7

Ex. 3
Based on the data, draw an expected decision boundary to separate the classes. Express the decision boundary as a set of lines. Note that the combination of such lines must yield to the decision boundary. The number of selected lines represents the number of hidden neurons in the first hidden layer. To connect the lines created by the previous layer, a new hidden layer is added. Note that a new hidden layer is added each time you need to create connections among the lines in the previous hidden layer. The number of hidden neurons in each new hidden layer equals the number of connections to be made.

Ex. 6
The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example, linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid. By using a kernel function we essentially map our inputs to a different (usually higher dimensional) feature space in which the problem is linearly separable.

If $K_1(x, x^{'})$ and $K_2(x, x^{'})$ are kernels, so are cK_1 for $c > 0$, $K_1 + K_2$ and K_1K_2.

Ex. 8
First among several criteria for diminished performance is the fact that the more dimensional distances are combined the more the total distance will average out.[3] Furthermore, the curse of dimensionality comes into play[5]. The high dimensionality requires more rows in the dataset but since nearest neighbour (NN) approaches normally consider all items in the data set, the calculations become long-running when you have more data or less powerful when you do

[3] Like the many throws of two dice converging towards 7 after enough trials.

not because of this curse of dimensionality. Since NN approaches require a point to be close on every axis, each added column essentially doubles this search space.

Ex. 13
When you have a lot of features, those features and combinations of features can start to fit rows that are so unique that only this particular item matches these particular rows. This causes a higher probability of classifying by memory instead of pattern recognition and makes the classifier classify by memory[3] on noise. Also, this process causes more features to become present that might not be related to the problem and it increases model complexity allowing it to fit too precisely on the training data and not generalize well to the test data.

1.4 Chapter 8

Ex. 5
In the late 1990s, there were a lot of discussions regarding optimization algorithms being better than others, nonetheless, when an algorithm performed better on one task, it would perform worse on another. This is when the "No Free Lunch" Theorem was established, which confirms this phenomenon. Indeed, according to this, no model will always perform better than another model[**kuhn**] and it is therefore recommended to try a variety of different approaches before committing to a model to use.

Ex. 6
An example of a case where Recurrent neural network (RNN) works better than its feed forward counterpart (FFNN) is when analysing text data from facebook posts as n-grams. Having a RNN in this case allows the model to remember sequences of n-grams, and therefore have a higher accuracy in it's predictions. This same concept can be applied to many aspect of the quantified-self, such as hearth-beat patterns, or any other form of data that relies on a temporal (or rather sequential) scale.

Ex. 8
A model performance depends greatly on the optimization of it's parameters. In order to optimize them, we use a variety of techniques of tuning. Few examples are Genetic Algorithms or Simulated Annealing, which allow for optimal parameters to be found. One tool that can also be used is the Dynamic Parameter Estimation Tuning Key Features (DPET) Result Analyzer, that is able to compare results from a variety of sets and therefore find the best possible model parameters with the advantage of being very fast [**etap**].

1.5 Chapter 9

Ex. 2
Since the user is the agent the reward function should be based on desirable

actions/outcomes taken by the user. The actions/steps the system should take should lead to better desired behaviour by the user whatever the steps taken by the system are or to desirable states to bring the user in. An example of an rudimentary reward function could be $R(s, a, s\prime) = 1 * s_1 - 0.5 * s_2 - 1 * s_3 + 0.5 * s_4$ Where $s_1 = useristraining$, $s_2 = userisovertraining$, $s_3 = userisnottraining$ and $s_4 = useristrainingwrongtraining$. The actions taken by the reinforcement learning RL in a few previous episodes could be rewarded according to this reward function via a Q-function process.

Optimizes actions of the user

Ex. 4
The Markov property states that any necessary information for the current state is encompassed in the previous state and there is no need for a history of states.[1] An example of a setting in the quantified self where this property does not hold might be proposed in the following way in a setting for Bruce. Having the Markov property present here yields certain consequences. For example, when yesterday Bruce was depressed and today Bruce is happy, this simple divergence need not be cause to take action to inform the therapist to give extra attention to Bruce. We do however not see the previous states of the days the week before, where it is apparent that Bruce was depressed on all of those days. If that knowledge would have been present the algorithm could capitalize much more on the current happiness and recommended the therapist to pay extra attention to Bruce.

2 Practical

2.1 Chapter 5

Ex. 1
The following plots where generated for clustering with different methods.

As can be seen the k-means clustering for the gyroscope (GYR) data differs from the accelerometer (ACC) data mainly in the fact that ACC data had a notably stronger in-group coherence and stronger separation between groups. For the GYR the in-group coherence was much more loose making the clusters bleed significantly into each other. As expected this translates to much higher silhouette scores of around 0.85 for ACC as compared to scores of around 0.5 for GYR data. While the data is inherently less clusterable it is noted that the clusters for ACC do not necessarily better correspond to the different activities than those for GYR.

Similar patterns can be seen for the k-medoids GYR data. With most silhouette values around 0.5 the clustering factor is less strong than with ACC data with silhouette values around 0.8. Again both for the GYR and ACC clustering the divergence from the actual activities is notable with ACC data being generally closer together in-group.

The main difference in dendrograms for ACC and GYR data is the number of subgroups and their structure. ACC data has four similar sized subgroups while

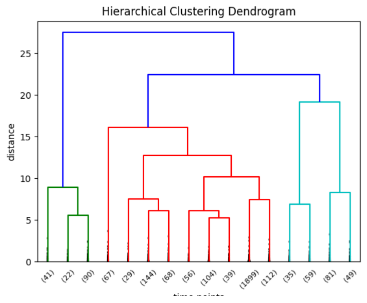
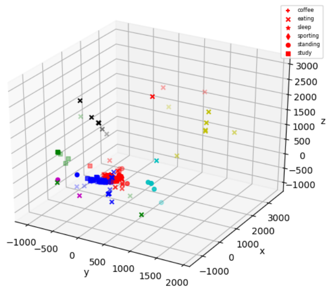


Fig. 1. Hierarchical clustering for phone gyroscope data

GYR data has one much larger subgroup besides two other subgroups making up three subgroups in total. With a top silhouette score of 0.73 the ACC data again appears more separable than GYR data with 0.67.

Ex. 2 A personal dataset was collected over 24 hours with a frequency of 100Hz (10ms) and subsequently resampled using episodes of 30 seconds. A few interesting clustering results of features are selected. The clustering for the magnetometer and the orientation of the device for the dataset is plotted below.



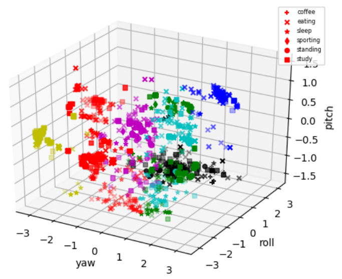


Fig. 2. K-means clustering for phone magnetometer data

Fig. 3. K-means clustering for phone orientation data

A few interesting patterns can be noted. Whereas the CrowdSignals (CS) data was aptly clustered using $k = 5$ for the KMeans clustering the personal data was best clustered using k=9. Since the number of unique labels in the personal data was 4 it is interested that these activities are possibly made up from multiple clusters or intersections of clusters.

Notable is the clustering factor for the magnetometer which are generally very well defined clusters for the different types of outliers and two less well defined clusters for the majority of values where the phone is in the pocket in a sitting position. A common core with small pockets of strong cohesive outliers is striking compared to the original dataset which had more generally dispersed points. It might be hypothesized that the nature of the activities (mostly sitting and some sporting) is cause to this divergence.

For the orientation in terms of yawn, roll and pitch it is noted that the locations appear very random in general with a few notable tight clusters. Sleep is most clearly separable from the other points in these terms but the clustering approach has split up sleep in several clusters. Standing also appears reasonably well-defined but is spit by the clusters.

Better than expected clustering results are noted for the magnetometer while worse than expected results are noted for device orientation meter.

2.2 Chapter 7

Ex. 3

The machine learning models implemented in chapter 7 were implemented for the personally collected dataset. In simply predicting whether a user was training or not mean squared errors of around 0.25 where obtained. In contrast to the results obtained in the book these results exhibit small standard deviations over several runs possibly indicating a low amount of variance and a high bias. Additionally a naive bayes was implemented which nevertheless performed worst. The Neural Net (mlp) performed best after the random forest similarly to in the book chapter but a larger divergence was present. The low errors are hypothesized to be due to the clear patterns in the data (frequencies of pull-ups) and the short time span in which data for a relatively simple task was collected.

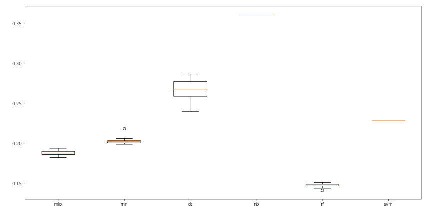


Fig. 4.

Ex. 1

In the case presented in the book, the cases with labels 'none' and 'multiple' were removed, and in this assignment we experimented with the repercussions of leaving said labels in.

Keeping the unknown labels and cases with multiple labels led to a big decrease in accuracy, where instead of it reaching 0.95, it stopped improving around the value of 0.91 (See figure 4 and 5).

The second images in these pictures show the impact of the regularization parameter upon the performance for both the training and the independent test set. The difference here is abyssal, as we can see a plummet in performance when the new labels were kept.

Finally, in the third images we see that performance on the training set increases when we decrease the minimum number of example per leaf. The same holds for the test set up to a certain extent. There is however a breaking point: the test set performance drops when a value smaller than 5 is selected, corresponding to over fitting.

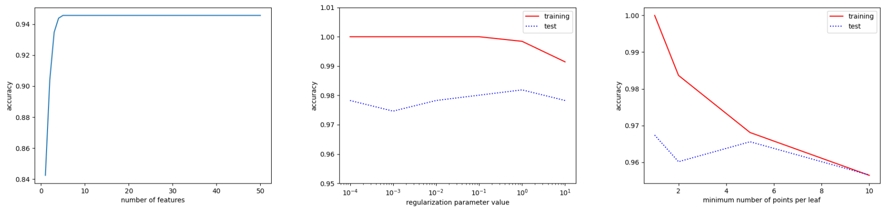


Fig. 5. Original labeling

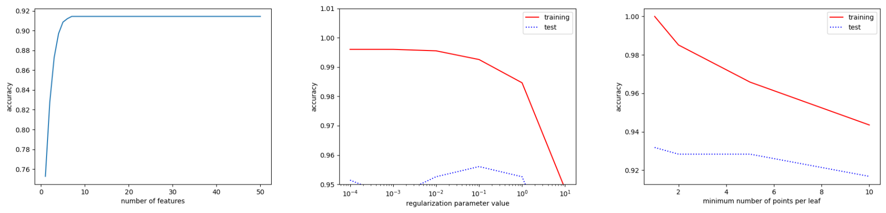


Fig. 6. With labels 'none' and 'multiple'

It is pretty clear that keeping cases with either two labels or unknown label damages the performance in a great way. It is therefore better to ignore these cases.

2.3 Chapter 8

Ex. 2

Preprocessing A personal dataset was collected with SensorLogger[4] where the activity of pull-ups was annotated with the weight and the number of pull-ups that where done. This data was preprocessed according the same steps as the CrowdSignals data where the adding of frequencies was of paramount importance regarding the nature of the movement of pull-ups. After this preprocessing a dataset was obtained with 962 columns and 75620 rows with the sampling frequency of 200Hz for about 11 minutes. After resampling this data to 1 second, a final dataset with 540 rows was yielded.

We modified the neural network employed in chapter 8 of the book in such a way to make it regress the weight the user was currently doing pull-ups with. Preliminary data exploration showed correlations of 30% of several frequencies for several sensors with training providing hope for the results that could be obtained with a neural network.

Experimental Setup Three runs where averaged where for each run a gridsearch was done finding the most appropriate hyperparameters for the present neural network. 100 hidden neurons for 250 iterations with output bias provided the best results after search.

Results With scores of 0.771, 0.609, 0.855 and 0.585 for respectively the training mean squared standard error, standard deviation of the squared standard error, test mean squared standard error and standard deviation of the squared test standard error the results where particularly reinforcing. A few considerations can me made explaining the subtleties of these apparently strong results. The train and test data where split at a certain point in time (the original code) the fact that the algorithm scores better on the test data could be due to the fact that the sporting activities in the latter 20% of the recording time where more stereotypical while the training in the start of the assignment was more noisy. Besides this was the data collected for a short period with only episodes of doing nothing and periods of training. While regression was the goal this constituted a fairly simple problem and not a complicated multiple classification problem.

[4] This dataset was another dataset collected besides the dataset from Q5.

References

1. R. Bhatia, Noise Pollution: Managing the Challenge of Urban Sounds, https://earthjournalism.net/resources/noise-pollution-managing-the-challenge-of-urban-sounds, [On-line; accessed 28 January 2020], 2014
2. Liao, T., Clustering Time Series Data — A Survey, Pattern Recognition 38, 2005, doi= 10.1016/j.patcog.2005.01.025
3. Hanley, J.A. and Mcneil, Barbara, The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve, Radiology 143, 1982, doi= 10.1148/radiology.143.1.7063747
4. Verleysen, Michel and François, Damien, The Curse of Dimensionality in Data Mining and Time Series Prediction, 2005, doi= 10.1007/11494669 93
5. Hawkins, Douglas M., The Problem of Overfitting, Journal of Chemical Information and Computer Sciences 44, 2004, doi=10.1021/ci0342472
6. Gábor, A., Banga, J.R. Robust and efficient parameter estimation in dynamic models of biological systems. BMC Syst Biol 9, 74 (2015). https://doi.org/10.1186/s12918-015-0219-2
7. Dynamic parameter estimation amp; tuning. ETAP. (n.d.). Retrieved June 19, 2022, from https://etap.com/product/dynamic-parameter-estimation-tuning
8. Blumenthal, R. M., An extended markov property, Transactions of the American Mathematical Society 85, doi= 10.2307/1992961
9. Kuhn, M., Johnson, K. (2013). Applied predictive modeling. New York: Springer.