Data Mining Homework 2

Jennifer Probst

26 October 2020

Exercise 1: Time Series

Pair-of-classes	Manhattan	DTW,w=0	DTW,w=10	DTW,w=25	DTW,w=inf
abnormal:abnormal	67.77	67.77	38.65	26.48	25.37
abnormal:normal	67.52	67.52	34.20	26.94	26.35
normal:normal	45.65	45.65	24.42	22.17	21.87

b) In general one can say that abnormal(an):normal(n) cannot be distinguished very well from an:an. They separate quite well from n:n with the Manhattan distance and for low values of w; this gets less clear for higher values of w. I would probably take w = 10, because differences between all classes can be seen the clearest and this w is roughly 10% of the sample, which seems an adequate choice.

c) The choice of w defines the number of elements from one seq with which one element form the other can be compared. If we choose w=0 we only compare values at the same position in the sequence and therefore it makes sense that we end up with the same value as the Manhattan distance. Increasing w seems to lower the values and making the classes less separable.

d) It is not a metric, as it does not fulfil the 'identity of indiscernibles'. If we for example take x=(2,2,2) and y=(2,2), then DTW(x,y)=0, but x $\neq y$.

e) The runtime is reduced, because only elements with $|m - n| \leq w$ get computed. This happens n times, so it has a runtime of O(w*n). The runtime of normal DTW is $O(n^2)$, because a nxn matrix has to be calculated and for each entry previous entries can be used.

Exercise 2: Shortest path kernel

b)

Pair-of-classes	SP
mutagenic:mutagenic	5309.92
mutagenic:non-mutagenic	2706.78
non-mutagenic:non-mutagenic	1433.28

c) The runtime complexity of FW is $O(n^3)$, where n is the number of nodes in the graph, because we have to go through node triplets from n values, so n*n*n loops. For SP the runtime is $O(n * (n/2) * m * (m/2)) = O((nm)^2/4)$, which for n=m is $O(n^4)$. To improve the runtime we could use an alternative algorithm the get the SP score. We can create two lists of counts of shortest paths with the same length, one vector for S1 and one for S2. This happens in $O(n^2)$ for each count list. We then compute the sum of multiplications of the corresponding elements (counts of the same length) in O(n), which gives us an runtime of $O(n^2)$ for this improved algorithm.