Project 1. Temperature Sequence Analysis

Biological sequences such as DNA or Proteins are characterized to be relatively long sequences with a small size alphabet (e.g. 4 symbols for DNA and 20 for proteins). Many of the algorithms for biological sequence analysis were developed having these characteristics into account. Nevertheless, these algorithms can still be of use in other domains where the generated data has sequential nature. The goal of this project is to develop an analysis and methods development that allows the application of standard methods for biological sequence analysis to other domains. In particular, we propose the analysis of times series data we earth surface temperature data. This analysis may give us interesting hints on patterns of climate change. This dataset available from kaggle contains data from more than 3100 cities around the world with more than 3200 data points.

The following tasks need to be addressed during this project:

1) Perform the discretization (numeric to symbolic transformation) of the temperature dataset. Investigate different strategies for this processing step.
2) Adapt the algorithms for Dynamic Programming to be used in the generated datasets.
3) Perform clustering of the different cities based on their temperature data. Apply different methods for clustering (hierarchical, k-means, etc..).
4) Adapt the DP algorithms and other algorithms introduced in the AASB course to find temperature patterns shared among different cities.
5) Generate visualization of the clustering and patterns found.
6) Generate an iPython notebook or a Kaggle kernel to capture and document all the computation process.

Methods should be developed in the python language.

Data from here:

https://www.kaggle.com/berkeleyearth/climate-change-earth-surface-temperature-data

Step 1

Obtain statistics on the input dataset:

Number of time series representing different cities
Magnitude of the temperature data (minimum, average, maximum)
Number of time points per series
Other relevant statistics

Step 2

Explore different strategies for discretization of input data. Temperature data is in numerical format. To use bioinformatics algorithms that have an alphabet (discrete set of symbols) these data needs to be converted. Here we will explore and implement different functions to perform this step.

Equal Width Discretization. This method consists in dividing the temperature data in fixed intervals and assigns a symbol to each interval. The function to perform this task receives a parameter **K**, which represents the size of the alphabet. Then, for each series, k intervals are created for the values between min and max of the series. The size of each interval is given by (max – min)/k. Each of these intervals will correspond a symbol (can be a letter from a,b,c,d ....).

For more details please refer to the paper “Discretization of Temporal Data: a survey”

Three methods should be implemented for posterior comparison:

- Equal width distribution
- Equal Frequency distribution
- Symbolic Aggregate Approximation

Once these methods are implemented as functions each of the time series can be discretized.

Create visualization of the original time series and the discretized time series (sequences). Explore matplotlib.

Step 3

Adapt the dynamic programming algorithm introduced in the AASB classes for the analysis of these sequences. In principle minor modifications are required. Define the size of the alphabet and apply global alignment. The goal in this step is to obtain a similarity matrix for all the sequences (time series previously discretized). For that a matrix that contains a pairwise comparison of all the sequences should be obtained. If there are **n** sequences then the matrix should have **n x n** cells, where each cell M_{ij} contains the score of global alignment of **sequence *i*** with **sequence *j***.
Since matrix **M** contains scores of alignments, the more similar the sequences the higher the score. We will then convert this matrix in a distance matrix **D**. In

this case the more similar the sequences are the smaller is the distance between then. A possible transformation is: **D = M/max(M).**

Step 4

The goal of this step is to find motif sequences that are recurrent across different sequences. Given a motif of length **W,** a minimum number of sequences where the motif is present **P,** find all the motifs with a **frequency > P**. Adapt the methods introduced in AASB. Create a visualization for the occurrence of the motifs along the sequence.

Step 5
Clustering of the sequences based on their similarity matrix.
This step will be discussed later.