IE 48B Time Series Analytics
Homework 2, due November 22nd, 2021

Instructions: Please solve the following exercises using R (http://www.r-project.org/) or Python (https://www.python.org/). You are expected to use GitHub Classroom and present your work as an html file (i.e. web page) on your progress journals. There are alternative ways to generate an html page for you work:

- A Jupyter Notebook including your codes and comments. This works for R and Python, to enable using R scripts in notebooks, please check:
  - https://docs.anaconda.com/anaconda/navigator/tutorials/r-lang/
  - https://medium.com/@kyleake/how-to-install-r-in-jupyter-with-irkernel-in-3-steps-917519326e41

  Things are little easier if you install Anaconda (https://www.anaconda.com/). Please export your work to an html file. Please provide your *. ipynb file in your repository and a link to this file in your html report will help us a lot.

- A Markdown html document. This can be created using RMarkdown for R and Python Markdown for Python

Note that html pages are just to describe how you approach to the exercises in the homework. They should include your codes. You are also required to provide your R/Python codes separately in the repository so that anybody can run it with minimal change in the code. This can be presented as the script file itself or your notebook file (the one with *.ipynb file extension).

The last and the most important thing to mention is that academic integrity is expected! Do not share your code (except the one in your progress journals). You are always free to discuss about tasks but your work must be implemented by yourself. As a fundamental principle for any educational institution, academic integrity is highly valued and seriously regarded at Boğaziçi University.

Task: Penalized regression approaches for time series representation

Background
Assuming you are familiar with linear regression, this task requires you to deal with penalized versions of the linear regression approaches. There are alternative penalization approaches for data with a specific structure such as time-series. Suppose we would like to perform regression given a time series data set with some assumptions specific to time series observations. A penalized regression approach with fused penalties (Tibshirani et al., 2005) solves the following optimization problem:

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \ \frac{1}{2} \sum_{i=1}^{n} (y_i - \beta_i)^2 + \lambda \sum_{i=1}^{n-1} |\beta_{i+1} - \beta_i|$$

The first part is the sum of squared errors when each time series observation i is represented by β_i, second part is the fused lasso penalty. As you can see from the equation, the representation for the temporally close observations (i.e. coefficient of observations at time i and time $i+1$) are motivated to be close to each other. This approach is so called 1D Fused Lasso. This is an alternative way to obtain an adaptive piecewise constant approximation. Note that λ is the only parameter of the algorithm. There are alternative strategies to determine the appropriate level of lambda (to be discussed).

Assume that we are interested in representing CBF training series using 1D Fused Lasso and regression trees. Our aim is to compare two alternative adaptive piecewise constant approximation in terms of their representation ability and classification performance.

Subtasks:

1- For each training series of CBF, obtain an appropriate representation using 1D Fused Lasso. This will require you to select λ parameter for each time series. For automatic choice of λ in this type of problems, there is a well known approach which leaves every k points out of the observations and learns a representation for the remaining observations and then evaluates the approximation in the left-out observations. This idea is illustrated for a sample time series with 10 observations for the case of k=2 in Figure 1. The representation is learned over yellow observations and tested on gray ones for each lambda setting.

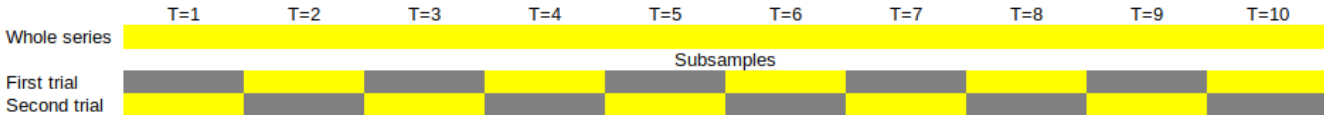

Figure 1. Subsampling in the evaluation of alternative lambda values.

Determine an alternative lambda sequence (fixed for each series) and select an appropriate k (k=10 should be OK) and obtain the best representation for each time series using the proposed strategy.

(Hint: If you are R used, this is already implemented in genlasso package. To the best of my knowledge, there is no readily available tool to perform this evaluation in Python unfortunately. There are some python modules to solve this optimization problem with examples. You may want to check cvxpy and related example https://www.cvxpy.org/examples/applications/l1_trend_filter.html Please note that this is not the exactly same problem, the formulation here is similar however the aim is to obtain a piecewise linear model).

2- Perform the similar task with regression trees. If you recall, regression trees require to set certain parameters of the tree learning process (analogous to lambda setting as in 1D Fused Lasso). You can fix complexity parameter to zero, minsplit to 20 and minbucket to 10 and only search for the best parameter of maximum depth in your analysis. While choosing the best maximum depth setting, you can follow the evaluation strategy illustrated in Figure 1.

3- One you have completed the previous tasks, you have the information of the best parameter settings for each approach. After you obtain the representation for each time series (using the best parameters), you can calculate the approximation error (i.e. mean squared error for each series – 30 MSE values for

CBF set). Compare the error rates using the boxplot of MSE values for each approach. Comment on your findings.

4- Assume that we would like to perform a comparison based on the accuracy of 1-NN classifier on the representations. Compare the accuracy of 1-NN classifier on the training data when you use the raw time series as input versus the representations. You can use Euclidean distance to calculate the distance between the time series.