Mini-Project (ML for Time Series) - MVA 2025/2026

Allison Zhuang allison.c.zhuang@gmail.com
Felipe Vicentin mva@a.vicentin.xyz

December 14, 2025

1 Introduction and contributions

In supervised and unsupervised learning scenarios, one of the most common problems is feature selection. Simply put, given a dataset with d features, one wants to select $r < d$ features that will be used by the learning algorithm. Most feature selection methods either filter a subset from the original features or transform them into a new representation space, and then filter from this new representation. A notable example of the latter strategy is PCA, which finds orthogonal components that maximize the variance in the data.

The paper on which this project is based, Laplacian Score for Feature Selection (LS), proposes a new feature selection method that filters features from the original representation of the data. LS was originally mainly applied to tabular classification and image clustering tasks, but in this project we extend it to Time Series data. To this end, we modify the distance function used in the original paper (Euclidean distance) to the Dynamic Time Warping (DTW) distance, which is better suited for time series.

Finally, to test our results, we compare the LS feature selection algorithm to other types of feature selection on 5 different classification tasks. We found no original source code from the authors, so we coded the entire algorithm ourselves. We used external libraries to define the models used to train the classification algorithm and to import the metrics to quantify their performance. New experiments were performed to measure the effect of the t hyperparameter and to test other distance functions (L2, L1, DTW). Our main improvement to the paper is the addition of the DTW distance to work with time series. The repartition of our work is as follows: **Allison**: First implementation of LS, feature extraction, data analysis/generating figures, write report (sections 2.3, 3, and 4). **Felipe**: Find datasets, optimize feature extraction, experiments class, write report (sections 1, 2.1, and 2.2).

2 Method

Let $X = (\mathbf{x}_1^\top, \ldots, \mathbf{x}_n^\top) \in \mathbb{R}^{n \times d}$ be our data set, where each $\mathbf{x}_i \in \mathbb{R}^d$ is a data point. The main idea is to model the similarity between data points as an undirected graph. Then, an edge between $\mathbf{x}_i$ and $\mathbf{x}_j$ symbolizes some similarity among these points. The weight of the edge translates to how similar the two nodes are, i.e., high weight implies high similarity. This graph structure is then used to weight the contribution of each feature. Let $S \in \mathbb{R}^{n \times n}$ be the adjacency matrix of said graph. As such, S_{ij} encodes the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$ ($S_{ij} = 0$ if there is no edge $i \to j$).

Once with the graph, we want to optimize two conditions to get the relevance of feature r. Firstly, we maximize its variance, as this increases its representative power. Here, we model the distribution of the data using the graph. In particular, given the graph's degree matrix $D = \operatorname{diag}(S\mathbf{1}_n)$, we say that $d = (D_{ii} \mid i = 1, \dots, n)$ is the discrete probability distribution of the nodes. Then, we can compute the variance of the r-th feature $\mathbf{f}_r = X\hat{\mathbf{e}}_r$ as follows.

$$\begin{aligned} \mathbb{E}[\mathbf{f}_r] &= \frac{1}{\sum_i d_i} \sum_i \mathbf{f}_{r,i} d_i = \frac{\mathbf{f}_r^\top D \mathbf{1}_n}{\mathbf{1}_n^\top D \mathbf{1}_n}, \\ \operatorname{Var}(\mathbf{f}_r) &= (\mathbf{f}_r - \mathbb{E}[\mathbf{f}_r]\mathbf{1}_n)^\top D (\mathbf{f}_r - \mathbb{E}[\mathbf{f}_r]\mathbf{1}_n) = \tilde{\mathbf{f}_r}^\top D \tilde{\mathbf{f}_r}. \end{aligned}$$

Secondly, we wish that the features respect the underlying graph structure. One can guaranty this by minimizing the differences of a feature, weighted by the similarity of the nodes. Rigorously, we minimize the following.

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} (X_{i,r} - X_{j,r})^2 S_{ij} &= \sum_{i=1}^{n} \sum_{j=1}^{n} (\mathbf{f}_{r,i}^2 + \mathbf{f}_{r,j}^2 - 2\mathbf{f}_{r,i}\mathbf{f}_{r,j}) S_{ij} \\ &= 2\mathbf{f}_r^\top D \mathbf{f}_r - 2\mathbf{f}_r^\top S \mathbf{f}_r = 2\mathbf{f}_r^\top L \mathbf{f}_r, \end{aligned}$$

where L is the Laplacian of the graph. Since $L\mathbf{1}_n = 0$, for all $\alpha \in \mathbb{R}$, we have

$$(\mathbf{f}_r - \alpha\mathbf{1}_n)^\top L (\mathbf{f}_r - \alpha\mathbf{1}_n) = \mathbf{f}_r^\top L \mathbf{f}_r + \alpha^2 \mathbf{1}_n^\top L \mathbf{1}_n - 2\alpha \mathbf{f}_r^\top L \mathbf{1}_n = \mathbf{f}_r^\top L \mathbf{f}_r.$$

In particular, we can take $\alpha = \mathbb{E}[\mathbf{f}_r]$ and immediately see that the above is equal to $2\tilde{\mathbf{f}_r}^\top L \tilde{\mathbf{f}_r}$.

Thus, we join these two goals into a single feature score that maximizes the feature variance and minimizes the contrast to the graph structure:

$$L_r = \frac{\tilde{\mathbf{f}_r}^\top L \tilde{\mathbf{f}_r}}{\tilde{\mathbf{f}_r}^\top D \tilde{\mathbf{f}_r}}.$$

Now, how do we generate the undirected graph that models the similarity within our dataset? There are many ways to deal with this problem and the authors propose two strategies. The first one consists in running a Nearest Neighbors algorithm on the dataset and connecting the K nearest neighbors. If label information is also available, it is possible to connect nodes that are in the same class. Then, one can compute the similarity between two nodes using the euclidean distance:

$$S_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{t}},$$

using a suitable hyperparameter $t > 0$. The second approach they present is useful in the context of classification tasks. It requires knowledge of the labels of each data point y_i. Then, we connect nodes in the graph if they are from the same class. The corresponding similarity is constant per class and equal to n_i^{-1}, where n_i is the number of elements in class i. They go on to prove that this instance of the graph yields a Laplacian score L_r that is equivalent to the Fisher score, another common feature selection score.

2.1 Suggested improvement

In this project, we are mostly interested in working with time series data. Following what has been done in class, we extract many features for each series and, then, try to perform some kind of prediction. The Laplacian score, as is, works with this set of extracted features, but can we do better? Our hypothesis is that we may be able to improve the feature selection if the graph structure is built over the similarity of the time series themselves. To this end, we propose using the DTW distance instead of the Euclidean distance. This translates in both the Nearest Neighbors algorithm and in the similarity matrix S. Specifically, we will use the following metric as a similarity criterion:

$$S_{ij} = e^{-\frac{\mathrm{DTW}(\mathbf{s}_i, \mathbf{s}_j)^2}{t}},$$

where $\mathbf{s}_k$, $k = 1, \dots, n$, is the time series signal.

2.2 Implementation details

In order to speed up the runtime of the Laplacian algorithm, we vectorized our operations:

$$\mathbf{L}_r = \frac{\mathrm{diag}(\tilde{\mathbf{F}}^T \mathbf{L} \tilde{\mathbf{F}})}{\mathrm{diag}(\tilde{\mathbf{F}}^T \mathbf{D} \tilde{\mathbf{F}})}, \qquad \tilde{\mathbf{F}} = \mathbf{F} - \mathbf{M}\mathbf{1}^T, \qquad \mathbf{M} = \frac{\mathbf{FD1}}{\mathbf{1}^T\mathbf{D1}}$$

Note that $\mathbf{F} \in \mathbb{R}^{n \times m}$ (features × samples). The r-th row of $\mathbf{F}$ is the transposed feature vector, f_r^T, containing all m sample values for the r-th feature. $\tilde{\mathbf{F}}$ takes the place of $\tilde{f}_r$ as the mean-removed data matrix. We take the diagonals on the numerator and denominator because they contain all the terms where $r = r'$. This has the same effect as performing the individual dot product $\tilde{f}_r^T A \tilde{f}_r$ for every feature r separately. Finally, we return the negative of the Laplacian score so that greater Laplacian scores translate to better features.

3 Data

To show the versatility of our approach across different kinds of datasets, we used 2 datasets spanning 2 different classes of data.

First, we used heartbeat data from aeon[1]. To reduce computational cost, we truncated our heartbeat data by reducing each patient heartbeat sample to 2000 points. Our classification labels are 0 and 1 for normal and abnormal heartbeats, respectively. Examples are pictured in the Appendix (Figure 1).

Secondly, we used historical data of the closing stock prices of Amazon (AMZN) from Kaggle[2]. This data is characterized by high volatility and an upward exponential trend. Our date range was May 15, 1997 to August 02, 2017.

We processed our stock price data into fixed-day windows to generate classification labels for a prediction task. The prediction windows were 40 days long. The prediction task was to classify whether the price would be higher or lower a week (5 days) after the last day of the window. The 40-day window represents approximately 2 months' worth of trading days. An example of our AMZN stock data is visualized in the Appendix (Figure 2).

We also denoised our heartbeat data using dictionary learning–surprisingly, accuracy in our downstream classification tasks actually suffered. It's possible that we should have added a sparsity component, as we set ours to 0 when running the experiments. When denoising our heartbeat

[1] https://timeseriesclassification.com/description.php?Dataset=AbnormalHeartbeat

[2] https://www.kaggle.com/datasets/praxitelisk/financial-time-series-datasets

data, we used 300 atoms. When denoising our stock data, we used about 50 atoms–our dictionary learning denoting scores indicate that the Mean Squared Error (MSE) and PSNR functions plateau at that point, which is visualized in the Appendix (Figure 3).

4 Results

To test the effectiveness of our feature selection methods, we ran about 180 random forest classification experiments. Our variables were as follows:

1. Dataset: Amazon stocks or heartbeat
2. Feature selection method: PCA (baseline), Laplacian score using the following 4 distance metrics: the Euclidean norm (as implemented in the paper), DTW distance (better for time series), DTW distance lower bound (computationally cheaper), and Fisher score.
3. $k \in \{5, 10, 15\}$ (hyperparameter for k-nearest neighbors)
4. $t \in \{0.1, 1, 10\}$ (hyperparameter for our distance kernel)
5. y (hyperparameter to determine if we should connect nodes in our Laplacian graph based on having the same labels) [3]
6. Raw or denoised data (to verify the effectiveness of our dictionary learning technique)

4.1 Stock Market Data (Irregular Time Series)

We observed the following phenomena in our Amazon data, which we find to be revealing about the workings of Laplacian score as a whole:

1. Classification accuracy generally improved as the number of selected features (r) increased. This shows that our features (mostly autocorrelations and Fourier features) were well-suited to the Laplacian score filter method and the irregular but cyclical stock data. 4, 5, 6, 7
2. The DTW distance metrics (DTW LS and Lower-bound DTW LS) consistently and significantly outperformed the standard Euclidean norm. We think this is because stock data involves inherent temporal misalignments. The Euclidean norm's "lock-step" comparison, $d_{\text{Euclidean}}(X, Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$, is highly sensitive to these shifts, resulting in a large distance (low similarity) for misaligned patterns. Meanwhile, the DTW distance is an elastic metric that correctly identifies true pattern similarity by optimally aligning the series, leading to a much better local structure graph and superior LS scores. 4, 5, 6, 7
3. Lower-bound DTW distance was almost as effective as the full DTW distance itself (this is good news, because lower-bound DTW is significantly cheaper to compute!). 4, 5, 6, 7
4. The performance gap between the Euclidean norm and the DTW distance seemed to grow for higher values of k (nearest neighbors) and higher values of t (kernel bandwidth). This seems to indicate that the Euclidean norm and DTW distance end up choosing different features. Increasing the neighborhood size forces the LS to capture a more global structure. DTW accurately defines this larger region of shape-similar patterns, while Euclidean incorporates more noisy, misaligned samples, compounding its error and widening the gap.

[3] Note: although we ran experiments to test the effect of y, this effect was ultimately negligible, which is why we will not be expanding upon its effects in this section

Meanwhile, a large t makes the similarity non-irrelevant for moderately distant signals. While the Euclidean distance might be huge for these signals (making the similarity close to 0), the DTW distance allows for moderate similarities in these cases, creating a better structural representation. 7

5. A sudden, dramatic jump in accuracy was observed at $r \approx 30$. The LS may have chosen the first 30 r features for the purpose of preserving the graph structure rather than identifying the most useful features, choosing higher-variance but more explanatory features later on. 4, 5, 6, 7

6. A sudden drop in accuracy for the Euclidean norm was observed $r \approx 35$. The additional features were probably good at preserving graph structure but worse for classification. These may have been incorrectly chosen by the flawed Euclidean metric, complicating and weakening the feature set. 7

4.2 Heartbeat Data (Regular Time Series)

1. Surprisingly, increases in r had seemingly no effect on the classification accuracy. Different experiments peaked in accuracy at $r = 5$ and $r = 30$. We think the volatile peak is caused by signal spreading: the discriminative information is "smeared" across many global features (Fourier/autocorrelation coefficients). This requires the selection of many coefficients to reconstruct the signal, leading to an inconsistent peak location. 9, 10

2. The Euclidean norm was unexpectedly competitive with DTW distance metrics at high r. We may have seen better performance by the Euclidean norm because heartbeat data is highly regular and well-aligned, diminishing DTW's primary benefit (elastic alignment). For instance, it would be surprising to see a heartbeat pattern in one sample repeated, but twice as fast or twice as slow, in a different sample. Biologically, human heartbeats are very narrowly constrained. The Euclidean norm excels at capturing magnitude differences at specific, aligned points, making it an effective and competitive metric for this regular time series type. 9, 10

3. All 3 accuracy curves (i.e., from DTW, DTW lower bound, and PCA) tracked the most closely with each other when k was high (≥ 15) or t was low (≤ 0.1). We think this is beacuse high k forces LS to capture more global variance, which is independent of the subtle differences between DTW and Euclidean norm. Low t creates a nearly binary similarity matrix, making the specific distance calculation irrelevant. 9, 10

4. DTW and lower-bound DTW did not track each other as closely in this case as they did with the stock data. 9, 10

4.3 Synthesis

Stock price data is irregular, but exhibits trends over the course of a month or a year; meanwhile, heartbeat data's identifying features are highly localized and morphological, in momentary perturbations or differences in amplitudes. Stock price data played well with our autocorrelation and Fourier features, while heartbeat data did not; stock price data benefitted from our use of DTW distance, while heartbeat data was better-suited to the Euclidean norm. These opposite classes of time series data created a fascinating and revelatory contrast. The experiments demonstrate that the precise nature of the time series data must dictate the choice of distance metric for LS feature selection.

Appendices

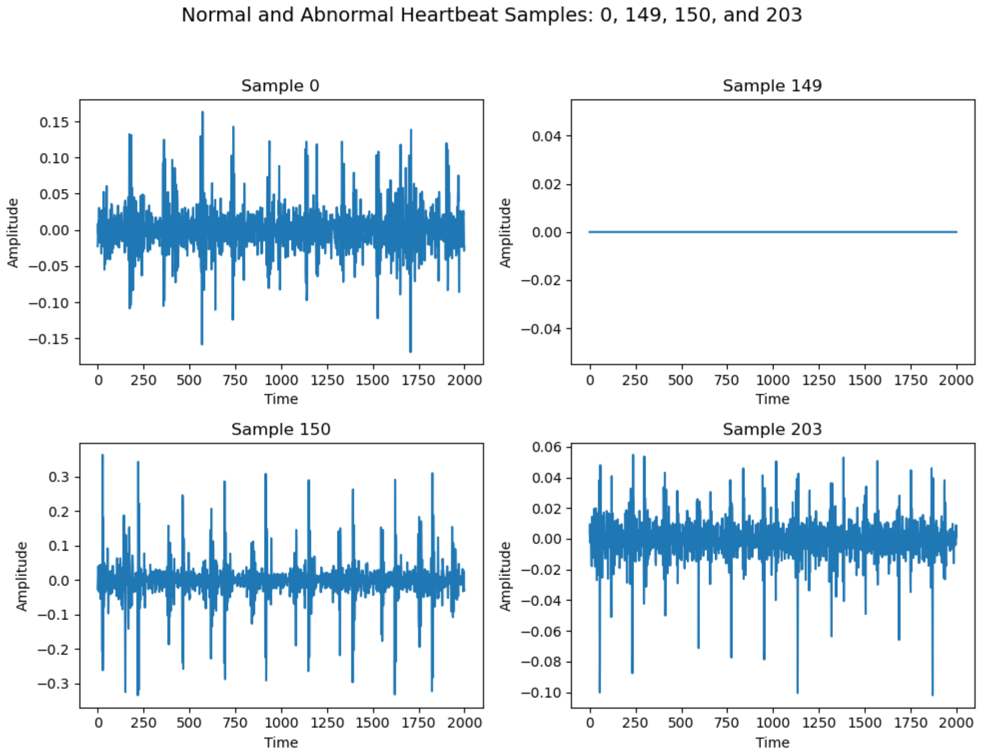
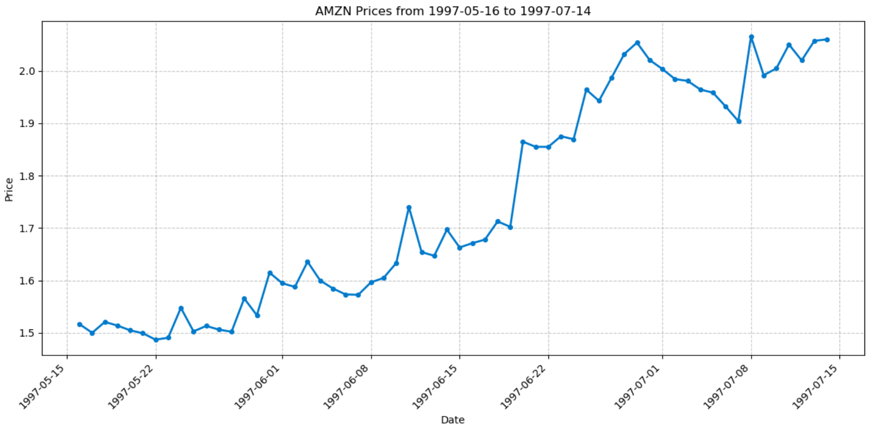
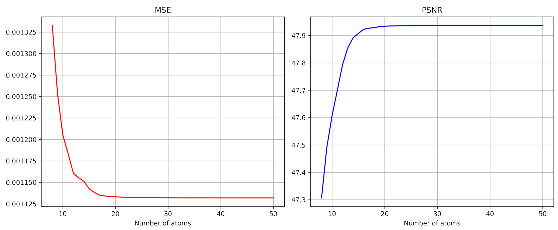


Figure 1: Samples 0, 149, 151, and 203 of the dataset. Sample 0 is normal (label 0), and the rest are abnormal (label 1).



(a) Amazon (AMZN) stock price data.

Figure 2: Visualization of the historical stock price data for AMZN.



(a) Denoising scores for the dataset.

Figure 3: Visualization of MSE/PSNR plateauing against the number of dictionary atoms.

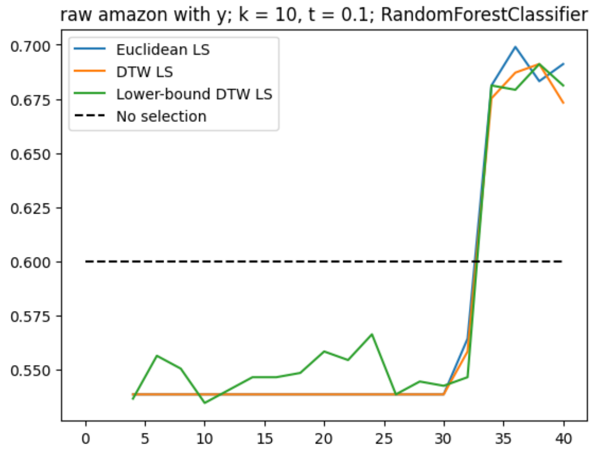


Figure 4: Lower value of k, to compare with Figure 6 below

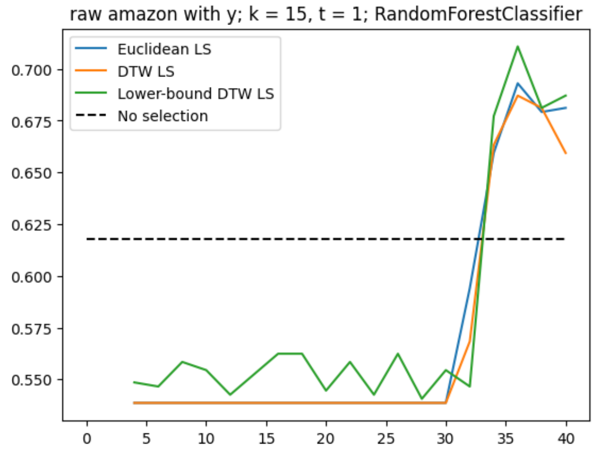


Figure 5: Compare behavior over changes in t with figures 6 and 7

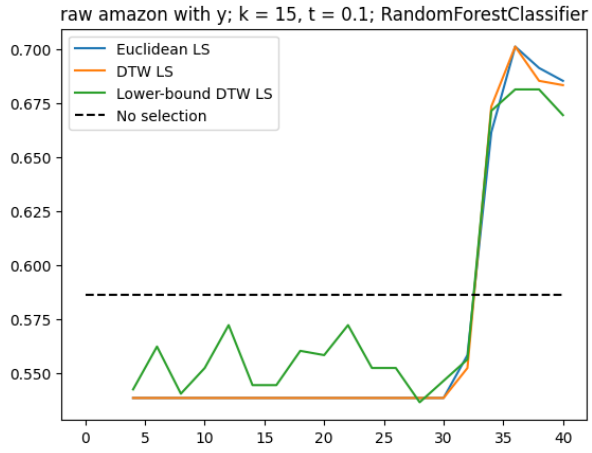


Figure 6: Higher value of k, to compare with Figure 4 above; smaller t, to compare with Figures 5 and 7

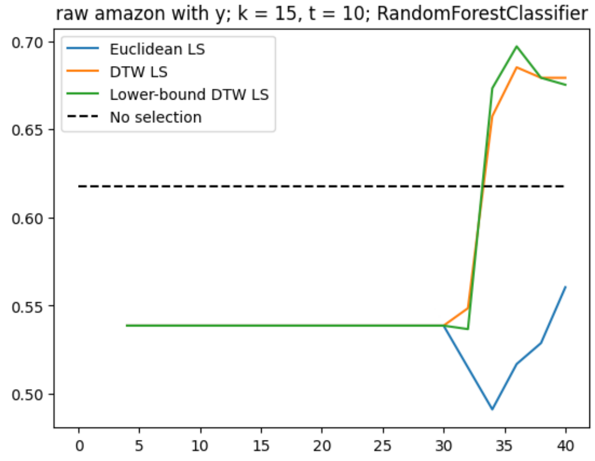


Figure 7: Compare behavior over changes in t with figures 5 and 6

Figure 8: All figures show a positive trend of accuracy versus r (features used), the closeness of the DTW and lower-bound DTW (especially for higher r), and a spike in accuracy at $r \approx 30$; Figure 7 shows the greatest performance difference between the DTW and Euclidean distance metrics, also showing the drop in classification accuracy at $r \approx 35$.

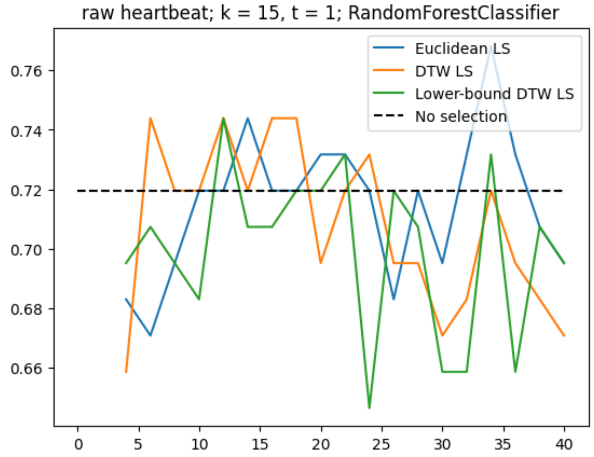


Figure 9: All 3 metrics track closely with each other for a high value of k

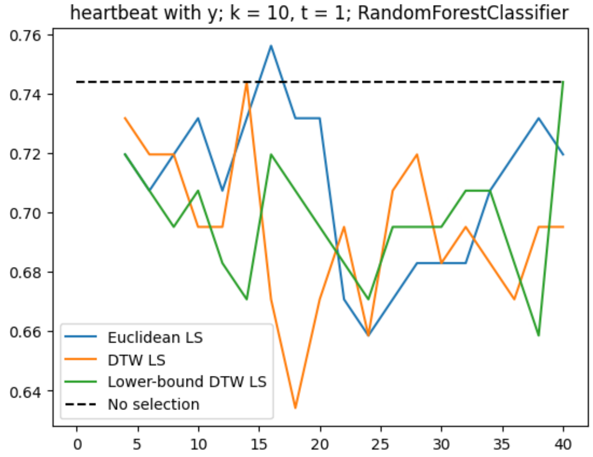


Figure 10: Metrics do not track closely with each other for a lower value of k

Figure 11: We can also observe in both figures that our classification accuracy does not increase with r, that the Euclidean LS has a higher maximum accuracy than the DTW LS, and that DTW and DTW lower bound do not track as closely this time.