PTML 9: 10/06/2022

TABLE DES MATIÈRES

1 MANIFOLDS

In the previous TP we saw an example of data that were lying in a linear subspace of dimension 3, inside $\mathbb{R}^{30}$. We could test this with a PCA.

However, in some situations the data also lie on subsets of $\mathbb{R}^d$, that are not linear subspace but also have a lower dimension, in a sense that is slightly different than the linear case. This subsets are called **manifolds** (variété, sous-variété) and the process of looking for such manifolds is called nonlinear dimensionality reduction or manifold learning.

https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction#Locally-linear_embedding

To properly define the dimension of a manifold, some mathematical background is necessary, however the intuition is that locally, this subsets can be perfectly described by a number p of parameters, that corresponds to this dimension.

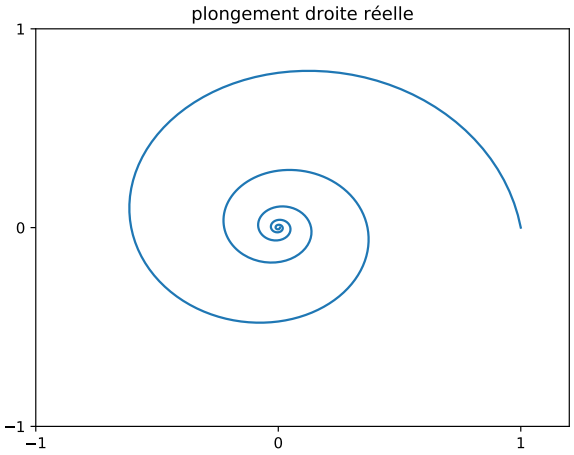


FIGURE 1 – 1 dimensional manifold, in $\mathbb{R}^2$. Locally, it is possible to parametrize this spiral with 1 number.

The goal of manifold learning is to project the data in a smaller subspace trying to preserve information on the dataset. Then, an algorithm might run more easily on the projected data.

You can find resources on this topic in these pages

https://scikit-learn.org/stable/modules/manifold.html

https://scikit-learn.org/stable/auto_examples/manifold/plot_compare_methods.html

In figure 2, you can find an example of manifold learning of the MNIST dataset with the t SNE method. The data are initially in $\mathbb{R}^{28\times 28}$.

https://lvdmaaten.github.io/tsne/

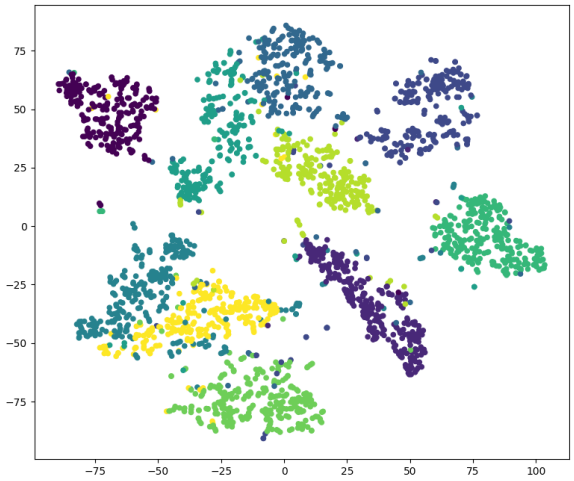


FIGURE 2 – Projection of the MNIST data to $\mathbb{R}^2$ with t-SNE. The colors correspond to the class of the sample, but this is just for visualization : the class is unknown by the dimensionality reduction method.

Some disadvantages of non linear manifold include the following facts :

— It is hard to determine a good output dimension (whereas in PCA we can use explained variance) and it is hard to interpret the embedded dimensions (whereas in PCA we know what they mean).

— They depend on the number of neighbors chosen (if relevant, as several of them involve a nearest neighbor search, often in the geodesic sense)

— They are often computationally slower.

You can experiment with scikit and try to reduce the dimension of some datasets with a nonlinear method.

2 SPARSE MODELS

In this exercise we will compare the performance of Ridge regression (linear model with L2 regularization) and Lasso regression (linear model with L1 regularization) for some specific datasets. The number of samples n and the number of features d will vary, but the optimal predictor is always **sparse** : it has a small number of non null components. We will see that depending on n and d, the difference in quality between Ridge and Lasso will vary.

https://scikit-learn.org/stable/modules/linear_model.html#

Files :

— **generate _sparse.py** : generates the dataset. n and d, and the number of non-null components of the best estimator θ^* can be set there.

— **lasso.py** : loads the dataset and learns with the Lasso estimator.

— **ridge.py** : loads the dataset and learns with the Ridge regression estimator.

Exercice 1 : Use the files in order to compare the performance of both estimators, for several values of n and d. For instance, you can observe some results like in the following figures. Check the sparsity of the estimators returned by the Lasso.

2.1 $n = 80,\ d = 100$

In this setting the performance of the Ridge regressor is very bad compared to that of the Lasso.

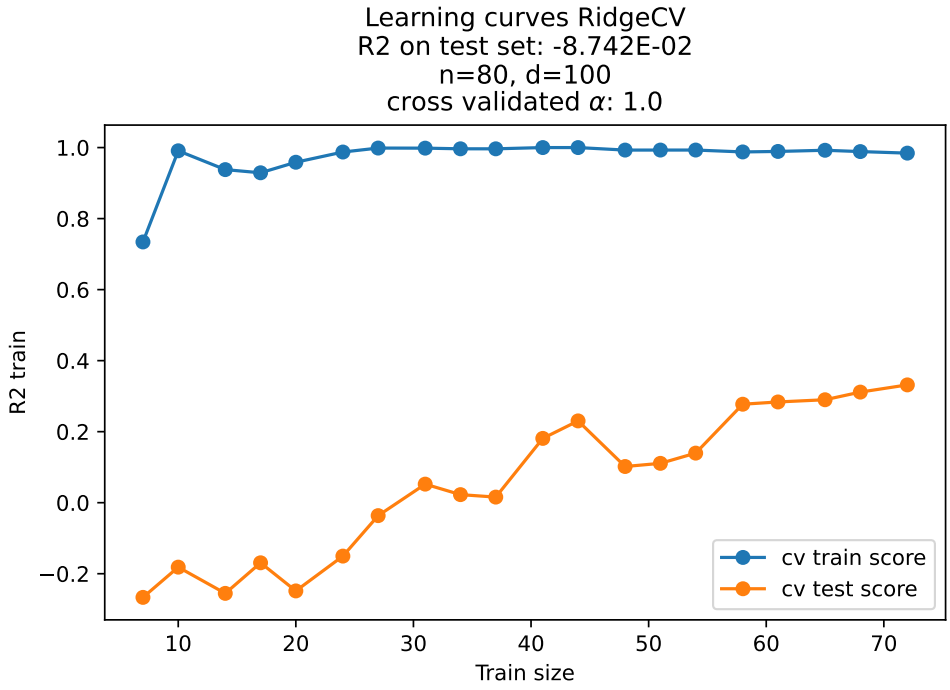


FIGURE 3

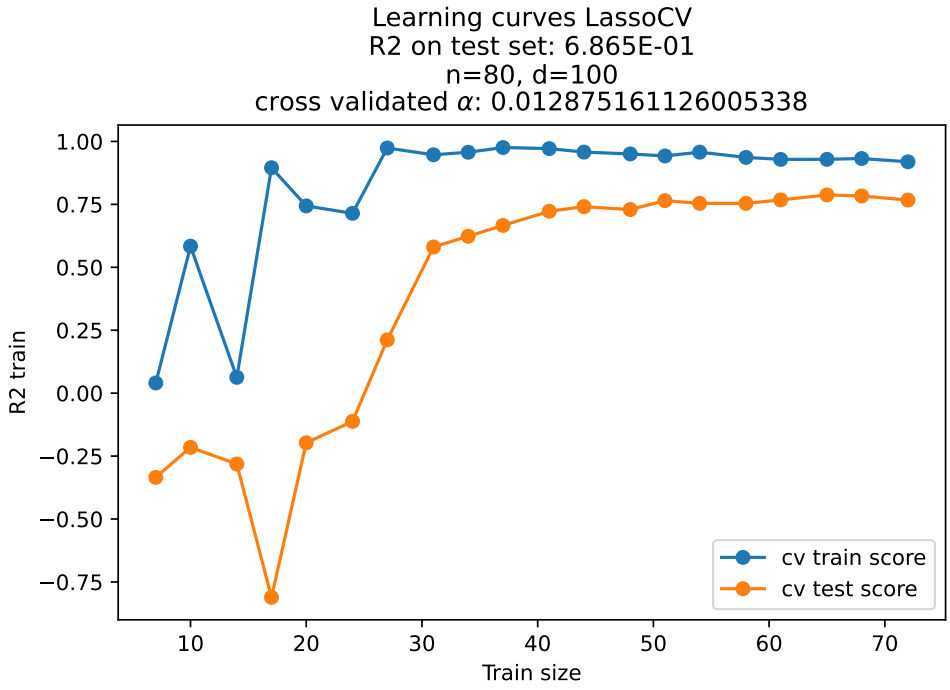
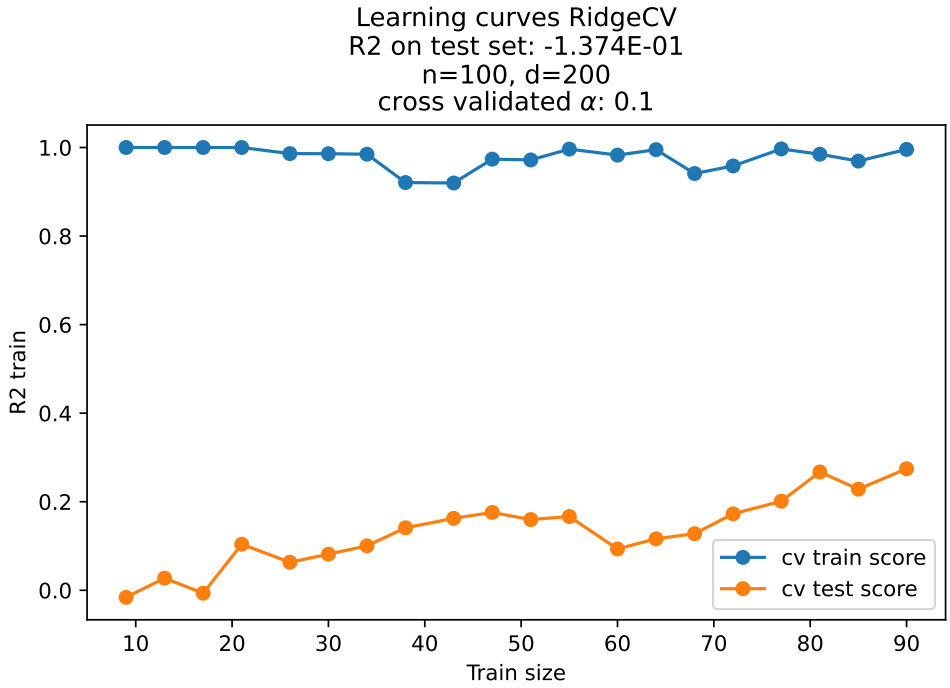
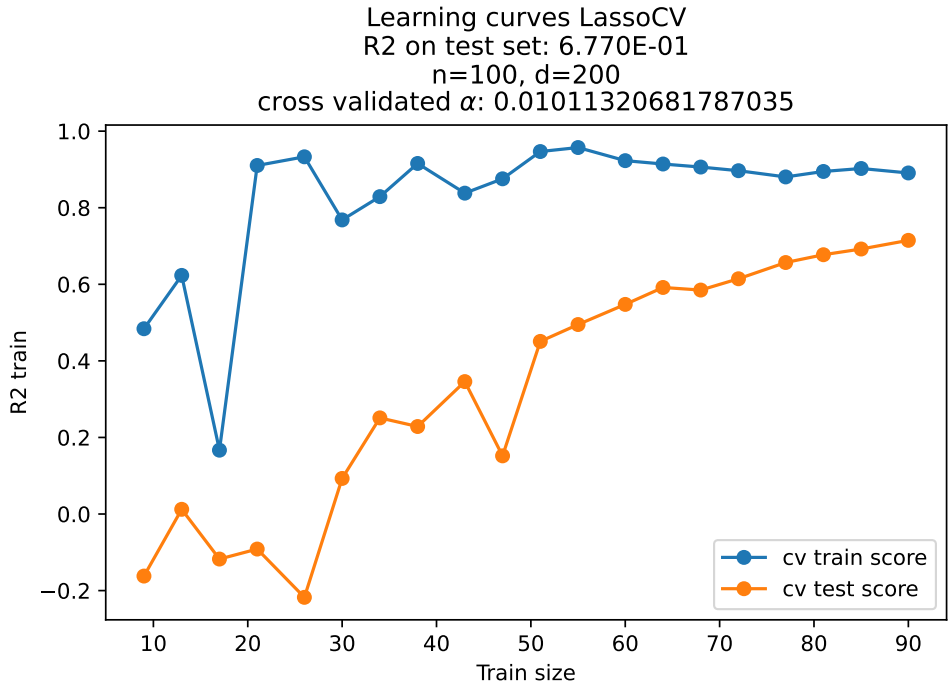


FIGURE 4

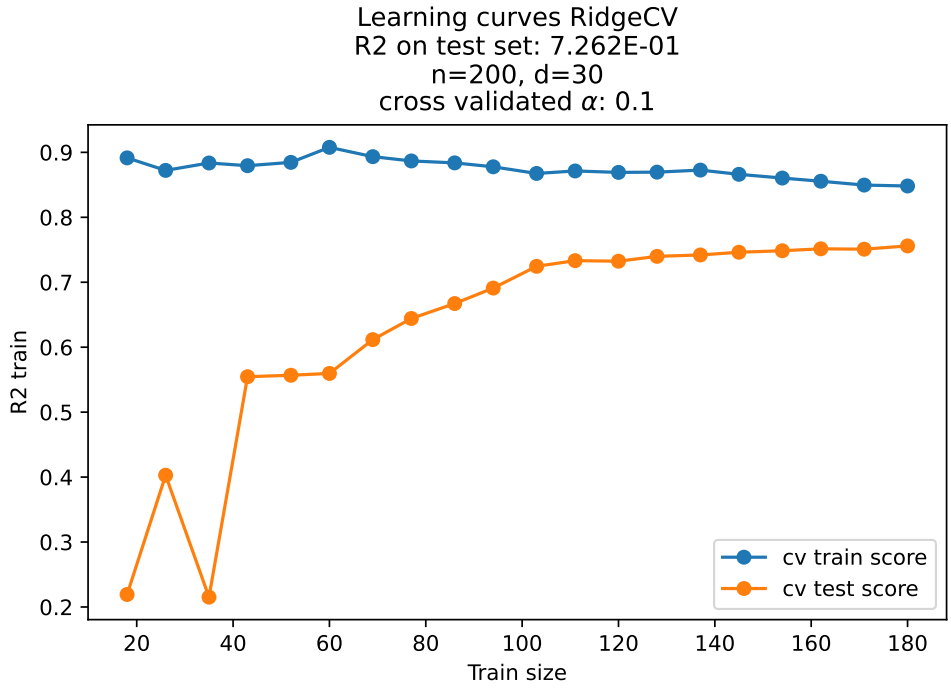
2.2 $n = 100,\ d = 200$

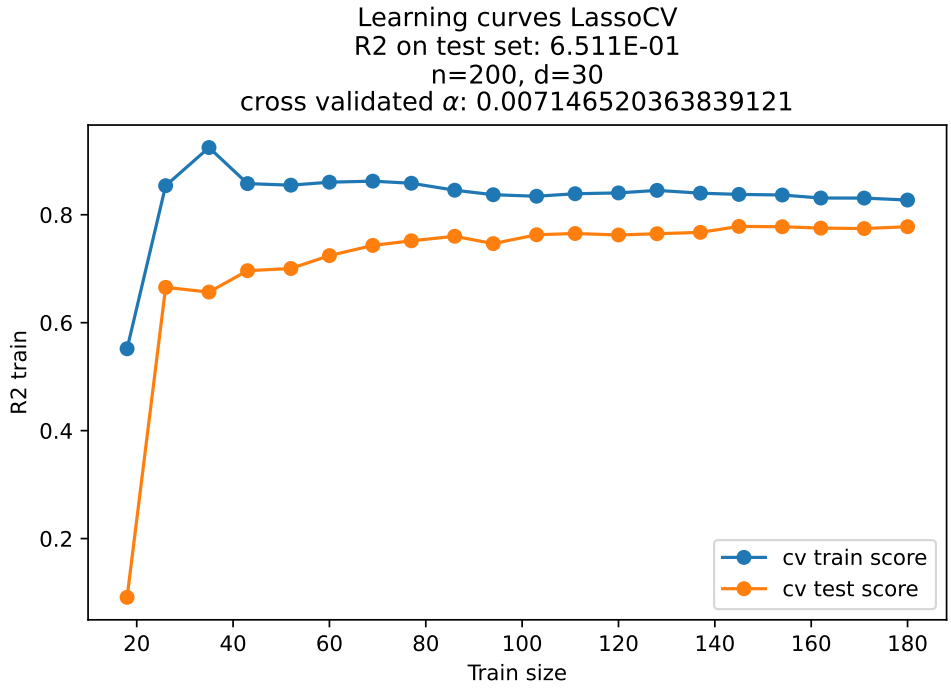




2.3 $n = 200,\ d = 30$

In this setting, the performance is similar for both models.





2.4 Conclusion

In some situations, especially when d is larger than n, and when the Bayes estimator is sparse, the Lasso estimator often has a better performance than the Ridge estimator.

Note that situations where $d \geqslant n$, or even $d >> n$ are not unusual in machine learning.

However, roughly speaking, solving the optimization problem is often easier for Ridge than for Lasso. Indeed, whereas Ridge regression leads to a strongly convex, differentiable optimization problem, the Lasso regression does not lead to a differentiable problem, which raises additional difficulties and different solvers, such as coordinate descent (which is used by scikit).

The problem of finding a sparse estimator, which means an estimator than depends only on a small number of variables, is called **variable selection.** The lasso is often thought of as an automatic variable selector.

Finally, note that Elastic Net is another method that combines both regularizations.