ECOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

CS-433 MACHINE LEARNING

Project 2 —— Machine Learning in Chemistry

Authors:
Pengkang Guo, Shiling Liang, Xiaolu Fang

Hosting Lab:
Computational Molecular Design Laboratory

Supervisors:
Alberto Fabrizio, Raimon Fabregat

2020

CS-433 Machine Learning Project 2 —— Machine Learning in Chemistry

Pengkang Guo, Shiling Liang, Xiaolu Fang
Ecole Polytechnique Fédérale de Lausanne (EPFL), 1015 Lausanne, Switzerland

***Abstract*—The dramatic development of machine learning has led to its widespread use in many fields, including the field of chemistry. A widely used molecular representation in quantum chemistry, physics-based molecular representation, suffers from the problem of redundant vectors with a large number of features. In this project, seven feature selection algorithms are applied to reduce the dimensionality of a molecular representation dataset with a huge number of features, and the prediction accuracy is compared.**

I. Introduction

In recent years, physics-based molecular representations have been widely used in quantum chemistry. However, it suffers from the problem of lengthy vectors with a huge number of features. The presence of a large number of redundant and trivial features can lead to low computational efficiency and reduced prediction accuracy, especially for those molecular properties that are sensitive to noise. In this case, dimensionality reduction becomes very essential. Dimensionality reduction (or dimension reduction) refers to adopting some mapping methods to map data in the original high-dimensional space to a lower-dimensional space so that the low-dimensional representation can account for some meaningful properties of the original data.

For the purpose of improving computational efficiency and prediction accuracy of molecular properties, a framework for dimensionality reduction is developed in this report. Several commonly used dimension reduction methods, both supervised and unsupervised, are tried, and the performance of each dimension reduction method is compared in combination with several different regression algorithms.

II. Data

In this project, the dataset used to perform these feature selection algorithms contains a molecular representation matrix $\mathbf{X}$ and a vector $\mathbf{Y}$ representing a molecular property, where $\mathbf{Y}$ are scalar target values. The matrix $\mathbf{X}$ contains 754 records and 27827 features, and the vector $\mathbf{Y}$ contains 754 records.

III. Framework

In this project, a total of seven feature selection algorithms were implemented, which are PCA (Principal Component Analysis), Isomap (Isometric mapping), MDS (Multidimensional Scaling), Feature Importance, UMAP (Uniform Manifold Approximation and Projection), MLKR (Metric Learning for Kernel Regression), and RFE (Recursive Feature Elimination).

We first split the data into training/validation/test sets, and the corresponding numbers of samples are 554, 100, and 100 respectively. Since cross-validation is too time-consuming, we use a fixed validation set. Then we optimize the hyperparameters of each feature selection algorithm based on its linear regression

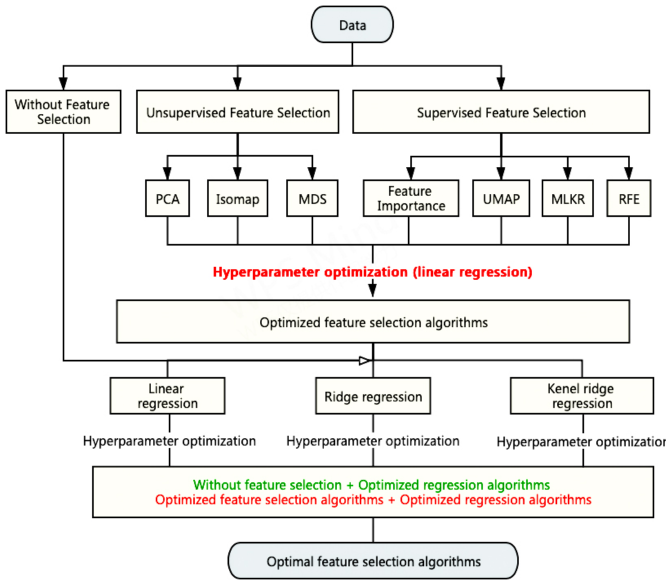
Data
Without Feature Selection
Unsupervised Feature Selection
Supervised Feature Selection
PCA
Isomap
MDS
Feature Importance
UMAP
MLKR
RFE
Hyperparameter optimization (linear regression)
Optimized feature selection algorithms
Linear regression
Ridge regression
Kenel ridge regression
Hyperparameter optimization
Hyperparameter optimization
Hyperparameter optimization
Without feature selection + Optimized regression algorithms
Optimized feature selection algorithms + Optimized regression algorithms
Optimal feature selection algorithms

Figure 1. Framework

MAE loss on the validation set. Linear regression is chosen because it does not introduce additional hyperparameters.

After hyperparameter optimization, we fix hyperparameters and retrain these models on the large dataset that consists of the training set and validation set to obtain the final models. Then we process the raw data $\mathbf{X}$ using the final models, whose output becomes the new data $\mathbf{T}$, which implies the generation of new molecular representations with reduced dimensionality.

We perform three regression algorithms, Linear Regression, Ridge Regression (RR), and Kernel Ridge Regression (KRR) on the new data $\mathbf{T}$, and they are optimized separately. In the end, the performance of the different regression algorithms after optimization is evaluated using the mean absolute error.

The overall framework of the project is shown in Figure 1.

IV. Data Preprocessing

Data normalization is important in some feature selection algorithms, which will be specified in the corresponding algorithm sections below.

V. Without Feature Selection

To evaluate the effect of feature selection, we perform the regression task on the raw data without feature selection and use its results as the baseline. The learning curve shown in Figure 2 is obtained. The minimal loss is 0.33981, given by kernel ridge regression with 654 training samples.

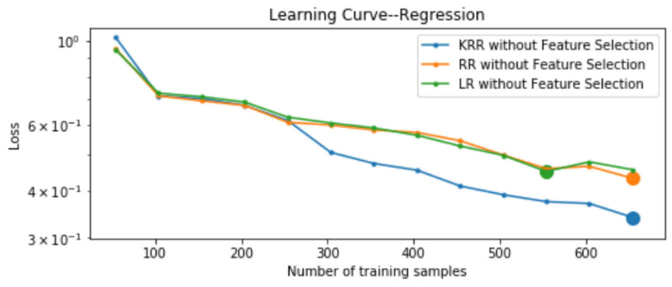

Figure 2. The loss v.s. number of samples for data set without feature selection

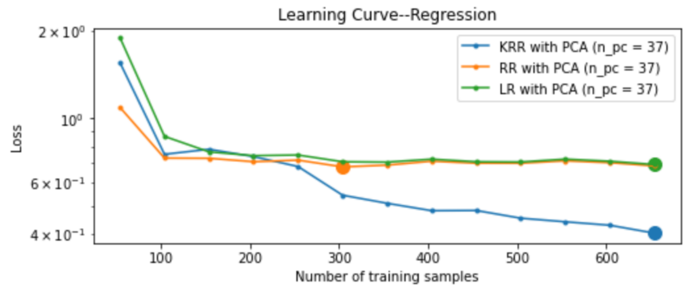

Figure 3. The loss v.s. number of samples for data set with feature selected by PCA

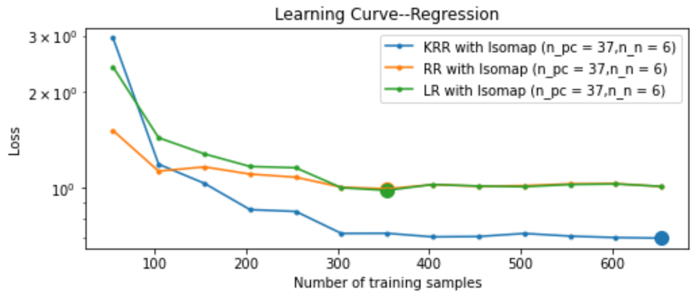

Figure 4. The loss v.s. number of samples for data set with feature selected by Isomap

VI. Unsupervised Feature Selection

In order to reduce the dimensionality of the data and select the most useful features to build predictive models, we need to use feature selection. Feature selection methods can be divided into two types: supervised and unsupervised. We first use unsupervised feature selection, which has the advantage of reducing the risk of overfitting.

A. PCA

Principal component analysis (PCA) is a popular dimensionality reduction algorithm and it can be used as an unsupervised feature selection algorithm. The idea of PCA is to generate a low-dimensional projection of high-dimensional input that maintains as much information as possible. In other words, PCA is to find a linear transformation that can map the input from high-dimensional feature space into low-dimensional latent space, $\mathbf{T} = \mathbf{X}\mathbf{P}_{\mathbf{XT}}$, and the projection $\mathbf{T}$ has the maximum variance. To construct the transformation matrix $\mathbf{P}$, we have to take a subset of the left singular vectors of $\mathbf{X}$. If the dimension of the potential space is k, we need to take the first k left singular vectors, which are arranged in descending order of singular values. These chosen left singular vectors are called principal components.

The algorithm is implemented with *PCA* class of *scikit-learn* library[1]. First, we optimize the number of principal components $n_{components}$ based on the MAE on validation set and get the optimal result as $n_{components} = 37$. Then we fix $n_{components}$ and retrain PCA using the whole training+validation set to get the final model. After that, we perform the final PCA on all datasets to get the low-dimensional representation. Finally, the regression task is performed using the low-dimensional representation, and the learning curve shown in Figure 3 is obtained. The minimal loss is 0.40241, given by kernel ridge regression with 654 training samples.

B. Manifold Learning

1) Isomap

Isometric mapping (Isomap)[2] is a kind of manifold learning, used for non-linear dimensionality reduction, which is an unsupervised algorithm. Manifold learning algorithms can be divided into two categories: those that maintain the global geometric framework, such as Isomap, and those that maintain the local geometric framework, such as Locally Linear Embedding (LLE). The main advantage of Isomap is the use of geodesic distance instead of the original Euclidean distance, which allows the geodesic distance relationship between any pair of data points in high-dimensional space to remain constant in low-dimensional space.

The algorithm is implemented with *manifold* class of *scikit-learn* library.We first fix the number of neighbors ($n_n = 25$) and optimize the number of coordinates (n_{pc}) for the manifold using linear regression, which eventually results in an optimization of $n_{pc} = 37$. We then optimize the number of neighbors n_n with the optimal $n_{pc} = 37$, still using linear regression, and obtain the optimized result $n_n = 6$. The optimized isomap is then applied to the original data for dimensionality reduction, after which the model is retrained and generate the learning curve shown in Figure 4. The minimal loss is 0.69533, given by kernel ridge regression with 654 training samples.

2) MDS

Multidimensional Scaling (MDS)[3] is another kind of manifold learning, the key point of which is to make the distance between samples in the high-dimensional space (original space) be maintained in the low-dimensional space. MDS includes classical MDS and no-classical MDS, the difference is that the distance criteria of classical MDS and no-classical MDS are European distance and non-European distance respectively. In this project, Euclidean distance (classical MDS) is applied.

The algorithm is implemented with *manifold* class of *scikit-learn* library. We first optimize the number of dimensions n_{pc} in which to immerse the dissimilarities using linear regression and get the optimal result as $n_{pc} = 2$. Then we use the optimized MDS to perform the dimensionality reduction. After that, the regression task is performed using the reduced-dimensional data,

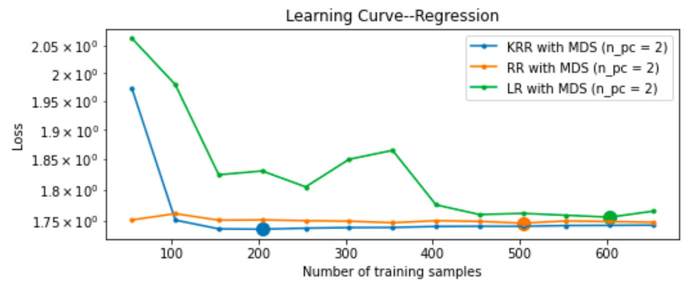

Figure 5. The loss v.s. number of samples for data set with feature selected by MDS

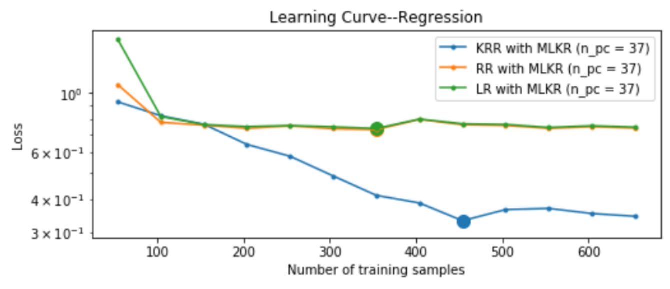

Figure 6. The loss v.s. number of samples for data set with feature selected by MLKR

and the learning curve shown in Figure 5 is obtained. The minimal loss is 1.73633, given by kernel ridge regression with 204 training samples.

VII. Supervised Feature Selection

We find that the regression loss (MAE) without feature selection is lower than that using the above three unsupervised feature selection algorithms. The reason is that although unsupervised feature selection can select features that maximize variance and retain the most information, however, these selected features are often not the most relevant to the target. To overcome this, we need to use both input and target in training, in other words, we need to use supervised feature selection algorithms.

A. MLKR

Metric Learning for Kernel Regression (MLKR)[4] is a supervised metric learning algorithm. We know the main shortcoming of unsupervised feature selection algorithms is that they can not catch the input-target relationship well. MLKR can be regarded as the supervised version of PCA. It learns a distance function by directly minimizing the leave-one-out kernel regression error and it can reduce the dimensionality of the input. Therefore, MLKR can be used as a supervised feature selection algorithm.

The algorithm is implemented with *MLKR* class of *metric-learn* library[5].The optimization process of MLKR is basically the same as that of PCA. What needs to be mentioned is that, because the MLKR calculation is very time-consuming, we just train MLKR models with $n_{components}$ in the range from 1 to 30. And we find that with the increase of $n_{components}$, the regression loss on the validation set generally decreases with a strong oscillation. To fairly compare the results of PCA and MLKR, we manually set $n_{components} = 37$. Then we retrain the final model, get the low-dimensional representation, and perform the regression task in the same steps as PCA. The learning curve shown in Figure 6 is obtained. The minimal loss is 0.32092, given by kernel ridge regression with 454 training samples.

B. UMAP

Uniform Manifold Approximation and Projection (UMAP)[6] is a newly-developed non-linear dimension reduction technique. Comparing to other non-linear dimensionality reduction techniques such as t-SNE, UMAP is much faster and can preserve a more global structure of data. In this project, we use umap-learn[7], a well-developed Python package which provides a supervised version of UMAP for regression task.

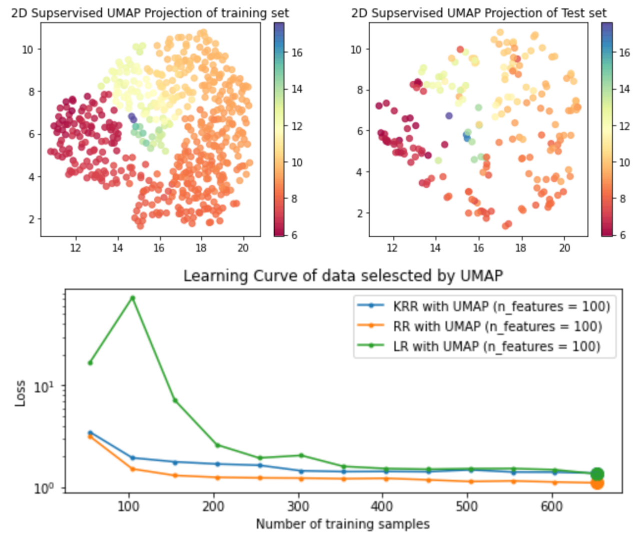

Figure 7. (Above) 2D supervised UMAP projection. (Below) Learning curve for regression task on data transformed by UMAP.

For visualization purpose, we can use UMAP to project the original 27827-dimension feature space to a 2D space (see Figure 7).

This embedding is trained on the training set and then be used to project the test set. As we can see in Figure 7, the test set dropped in the same shape as the training set in the reduced space. After that, we can take the low-dimensional feature space mapped by UMAP to the regression task. For the three regression tasks, the learning curves are shown in Figure 7.

For a UMAP which embeds data to a 100-dimension space, the optimal loss, 0.89471, is obtained under regression with 654 training samples. Although a 2D UMAP does a good embedding and can map the test set to the right places as shown in Figure 7, the overall loss is much higher than that with other dimension reduction methods.

C. Feature Importance

Importance selection is a model-based feature selection method to reduce the dimension of feature space. For a regression task, we can obtain a list of coefficients for each feature. For a regression model, $y_i = f\left(\sum_{j=0}^{n} c_j x_{ij}\right)$, where $\{x_{ij}\}$ is the set of features for the i-th data and $\{c_j\}$ are the corresponding regression coefficients. The magnitude of c_j can reflect the importance of the j-th feature. Therefore, we can use the absolute

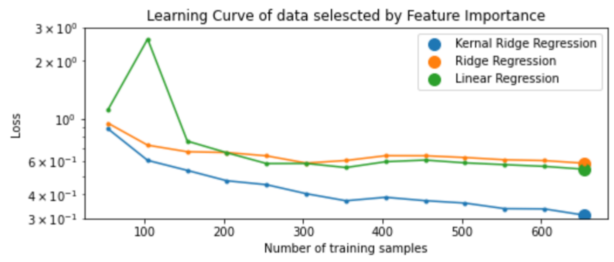

Figure 8. Learning curve for regression task on data selected by importance score

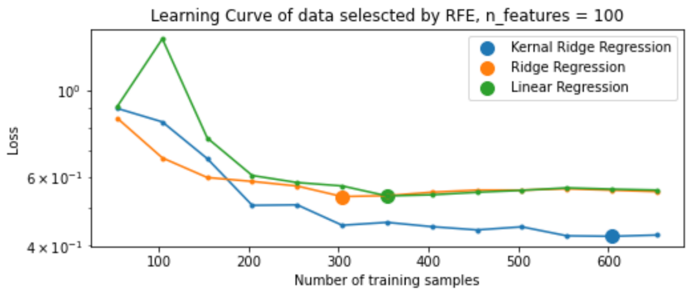

Figure 9. The loss v.s. number of samples for data set with feature selected by Recursive feature elimination (RFE) method.

value of the coefficients as a criterion to select the most important features.

The procedure starts with normalizing all data. Then train the data by linear regression to get the coefficients. With the magnitude of coefficients as the importance scores, we can remove the unimportant features. Here the number of selected features is fixed, thus the features with the largest 100 importance score are retained. We suppose the importance of different regression methods is the same. And using linear regression to extract the importance. The algorithm is implemented with *SelectFromModel* class from *scikit-learn* package. With the importance selector, we get 100 features retained from the original 27827 features. And then we do the regression task with the most important 100 feature, and obtain the learning curve as shown in Figure 8. The minimal loss, 0.31187, is given by kernel ridge regression with 654 training samples.

D. RFE

Recursive feature elimination (RFE)[8] is a variant of feature importance selection. Unlike the direct importance selection method mentioned in the last section, the RFE eliminate the features recursively: for each step, a specific number of unimportant features are dropped, then taking a regression task to find the importance score of the retained features, and repeat this procedure recursively until the desired number of features is obtained.

Setting the maximal number of features we want to keep as 100, and at each step, one feature is eliminated. The regression result on the final 100 features is shown in Figure 9. The minimal loss is 0.42244, obtained by kernel ridge regression with 604 training samples.

VIII. Results and Discussions

Table I lists the minimum losses for the above seven optimized feature selection methods with their corresponding regression algorithms which performs best.

Methods	Regression Algorithms	Minimum loss	Number of features
PCA	KRR	0.40241	37
MDS	KRR	1.73633	2
Isomap	KRR	0.69533	37
MLKR	KRR	0.32092	37
UMAP	RR	0.89471	100
Feature Importance	KRR	0.31187	100
RFE	KRR	0.42244	100
Without feature selection	**KRR**	**0.33981**	**27827**

Table I
Comparison of seven feature selection methods

Compared with unsupervised feature selection algorithms, we expect the inclusion of the information of the target vectors will make it easier for supervised feature selection algorithms to catch the relationships between the input and the target property and obtain better performance. As shown in Table I, the only two algorithms that have better performance than the basic case without feature selection are supervised feature selection algorithms. But we also notice that UMAP and RFE do not perform well. For UMAP, the problem could be that the manifold learned by UMAP is not good for linear regression models. While for RFE, the reason for poor performance needs further investigation.

Even using low-dimensional representation does not always result in smaller regression loss than using raw data, We can expect that the running time with feature selection will be much less than that without feature selection because the low-dimensional representations have much smaller numbers of features, and using it can greatly reduce the amount of computation. Here we just use LR, RR, and KRR, so the effect is not so significant. If researchers want to train some deep learning models with a large number of hidden layers and need to use gradient descent, the time saved by using low-dimensional representation will be considerable. Besides, fewer features mean fewer parameters and simpler models. So, when the number of samples is insufficient, using feature selection will help reduce overfitting.

IX. Conclusions

In this project, seven feature selection algorithms are experimented with to reduce the dimensionality of the original molecular representation dataset. Among these algorithms, MLKR and Feature Importance algorithms perform the best accuracy, achieving a minimum loss of 0.32092 and 0.31187, respectively, using kernel ridge regression, which is lower than 0.33981 in the case of no feature selection. Compared with the raw data with a dimension of 27827, a low-dimensional representation with a dimension of no more than 100 can achieve higher accuracy, which demonstrates the benefits of feature selection.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[2] J. Tenenbaum, V. Silva, and J. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, pp. 2319–2323, 01 2000.

[3] W. S. Torgerson, “Multidimensional scaling: I. theory and method,” *Psychometrika*, vol. 17, pp. 401–419, 12 1952.

[4] K. Q. Weinberger and G. Tesauro, “Metric learning for kernel regression,” in *Artificial Intelligence and Statistics*, 2007, pp. 612–619.

[5] W. de Vazelhes, C. Carey, Y. Tang, N. Vauquier, and A. Bellet, “metric-learn: Metric Learning Algorithms in Python,” *Journal of Machine Learning Research*, vol. 21, no. 138, pp. 1–6, 2020.

[6] L. McInnes, J. Healy, and J. Melville, “UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction,” *ArXiv e-prints*, Feb. 2018.

[7] L. McInnes, J. Healy, N. Saul, and L. Grossberger, “Umap: Uniform manifold approximation and projection,” *The Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.

[8] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Machine Learning*, vol. 46, pp. 389–422, 01 2002.