Maria Giustiniano

Ridge regression algorithm for median house price prediction

Academic year 2020/2021

Contents

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

Introduction

This project aims to implement the ridge regression algorithm with square loss from scratch and use it to catch how accurately it can make predictions in the field.
The ridge regression is a statistical tool for modelling the relationship between some explanatory variables and some real-valued outcome that differs from the conventional linear regression for the so-called penalty coefficient α. The method is part of the learning paradigm known as *Regularized Loss Minimization*, RLM for short, which tries to stabilize the linear regression algorithm by adding a **regularizer** α that prevents the output from shifting too much even when inputs slightly change.
Beyond focusing on the performance of the training error, the experiment also shows how the cross-validated risk estimate changes along with the penalty coefficient alpha and if its stability increases after having scaled features through the principal component analysis (PCA).
Specifically, section 1 introduces the dataset to the reader, whereas section 2 focuses on the algorithms' description. The third part defines all the steps performed throughout the experiment and, consequently, in the last section, the results of the analysis are shown and discussed.

1 Dataset

The dataset analyzed in this experiment contains 20640 records reporting an array of information about houses characteristics. Particularly, these are articulated by ten attributes, that are:

- **longitude**: A measure of how far west a house is; a higher value is farther west;
- **latitude**: A measure of how far north a house is; a higher value is farther north;
- **housingMedianAge**: Median age of a house within a block; a lower number is a newer building;
- **totalRooms**: Total number of rooms within a block;
- **totalBedrooms**: Total number of bedrooms within a block;
- **population**: Total number of people residing within a block;
- **households**: Total number of households, a group of people residing within a home unit, for a block;
- **medianHouseValue**: Median house value for households within a block (measured in US Dollars);
- **oceanProximity**: Distance of the house from the ocean or the sea.

Since the only categorical variable is **oceanProximity**, it is replaced by a list of numerical values that tries to enhance the distance from the ocean. Indeed:

$$\left\| \begin{array}{ccc} \text{ISLAND} & \rightarrow & 1 \\ \text{NEAR OCEAN} & \rightarrow & 2 \\ \text{NEAR BAY} & \rightarrow & 3 \\ <\text{1H OCEAN} & \rightarrow & 5 \\ \text{INLAND} & \rightarrow & 6 \end{array} \right\|$$

Deepening the investigation, it has emerged that the **totalBedrooms** feature has 207 missing values. To prevent any loss of information, they are substituted by the median value of the houses' number of total bedrooms accordingly to their distance from the ocean.

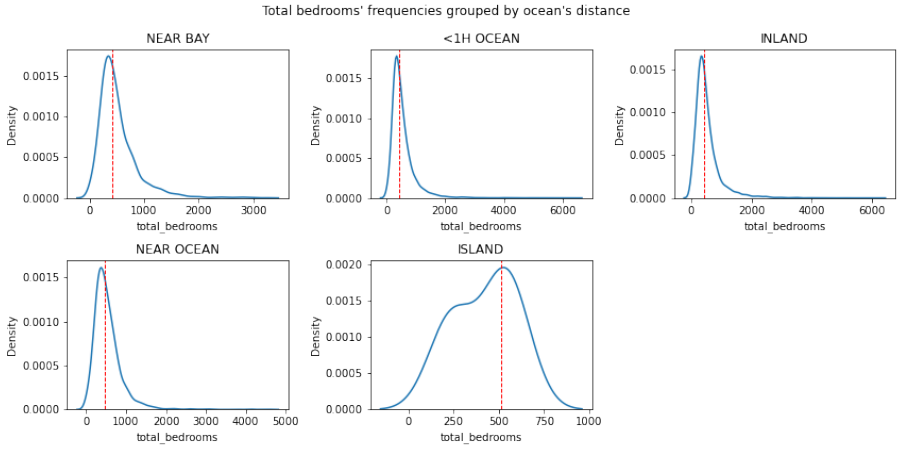

Figure 1: median number of houses' total bedrooms with respect to their distance from the ocean

Moreover, the attributes have different unit of measures and ranges, as shown below, which is why it a min-max normalization that shrinks all the interval in a range $\in [0, 1]$ is applied.

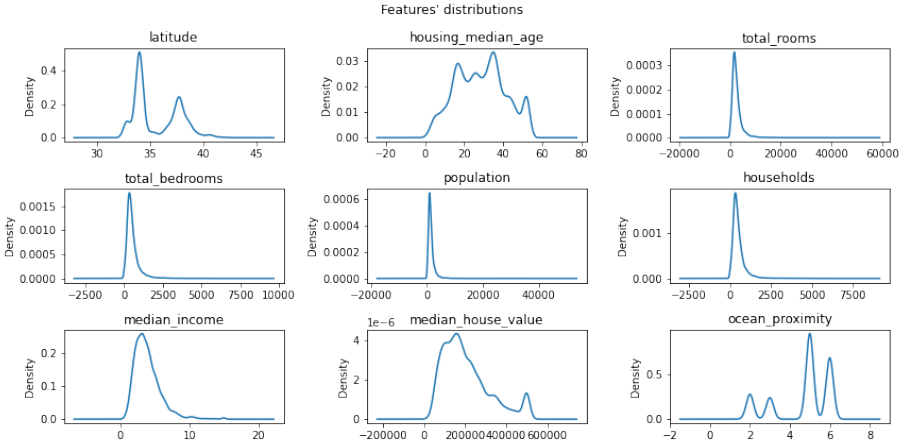

Figure 2: Attributes' distributions

2 Methodology

2.1 Ridge Regression

The ridge regression is a statistical tool for modelling the relationship between some explanatory variables and some real-valued outcome. It is a regularized version of the canonical linear regression, that is essentially used to prevent the predicted output from shifting too much even when inputs slightly change. Indeed, this tool is part of the learning paradigm known as *Regularized Loss Minimization*, RLM for short, obtained by introducing a **regularizer** α in the ERM practical. The RLM is a learning rule in which it is jointly minimized the empirical risk and a regularization function.

Given a learning problem, where the domain set $\mathbf{X}$ is a subset of $\mathbb{R}^d$, for some d, and the label set $\mathbf{Y}$ is the set of real numbers $\mathbb{R}$, the purpose of this procedure is to learn a linear function $h : \mathbb{R}^d \to \mathbb{R}$, which best approximates the relationship between the variables.
If in the canonical form, the best optimal predictor is ERM for the square loss, that is

$$\hat{\mathbf{w}} = \underset{w \in \mathbb{R}_d}{\operatorname{argmin}} \, ||S\mathbf{w} - \mathbf{y}||^2. \tag{1}$$

applying the RLM rule with Tikhonov regularization to linear regression with the squared loss [1], we obtain the following learning rule:

$$\hat{\mathbf{w}} = \underset{w \in \mathbb{R}_d}{\operatorname{argmin}} \, ||S\mathbf{w} - \mathbf{y}||^2 + \alpha||\mathbf{w}||^2. \tag{2}$$

Performing linear regression using equation (2) is called ridge regression. Its solution is obtained by comparing the gradient of the objective to zero,

$$\nabla(||S\mathbf{w} - \mathbf{y}||^2 + \alpha||\mathbf{w}||^2) = 0 \tag{3}$$

and the solution to ridge regression becomes

$$\hat{\mathbf{w}_\alpha} = (\alpha I + S^T S)^{-1} S^T \mathbf{y} \tag{4}$$

The singularity of $S^T S$, that could increases the variance error, is not a problem anymore. Indeed, if $\lambda_1 \geq ... \geq \lambda_d > 0$ are the eigenvalues of the invertible matrix $S^T S$, the eigenvalues of $\alpha I + S^T S$ are simply $\alpha + \lambda_1 \geq ... \geq \alpha + \lambda_d > 0$. This means that $\alpha I + S^T S$ is invertible for all $\alpha > 0$.

2.2 Dimensionality reduction

The principal component analysis (PCA) is an unsupervised machine learning technique to reduce dimensionality. It detects the directions along which the variance is maximized and projects the data onto them. The projection is mathematically expressed by a linear transformation which, in turn, is represented by a rotation matrix $W \in \mathcal{R}^{n \times d}$, where $n < d$ is the reduced number of dimensions.

PCA solves an optimization problem to find the best possible rotation matrix W and the inverse transformation $U \in \mathcal{R}^{d \times n}$. It computes the optimal transformations to map the data into a lower-dimensional space and to reconstruct it.

In mathematical terms, the optimization problem reads:

$$\underset{W \in \mathcal{R}^{n \times d}, U \in \mathcal{R}^{d \times n}}{\operatorname{argmin}} \sum_{i=1}^{m} ||x_i - UWx_i||^2, \tag{5}$$

where $||\ ||$ is the L_2 norm and m is the number of datapoints [1].
It has been shown in [1] that the optimal solution satisfies two conditions:

1. $W = U^T$
2. U is an orthonormal matrix

Hence, the optimization problem becomes

$$\underset{U^T \in \mathcal{R}^{n \times d}, U \in \mathcal{R}^{d \times n}}{\operatorname{argmin}} \sum_{i=1}^{m} ||x_i - UU^T x_i||^2. \tag{6}$$

After some simplifications that involve the computation of UU^T, it can be expressed as :

$$\underset{U \in \mathcal{R}^{d \times n} : U^T U = I}{\operatorname{argmax}} trace(U^T \sum_{i=1}^{m} x_i x_i^T U). \tag{7}$$

Finally, it has been proved that the solution of this optimization problem is the matrix U whose columns are the first n eigenvectors of the matrix $A = \sum_{i=1}^{m} x_i x_i^T$ corresponding to its n highest eigenvalues of the covariance matrix A [1].

2.3 Model selection and validation

In order to select the best possible parameter α and provide an accurate estimate of the risk of the predictor, two different split procedure are pursued:

1. Train-Validation-Test Split
2. k-fold cross validation

The *Train-Validation-Test Split* is a splitting strategy consisting of breaking the available examples into three sets: the **training**, the **validation** and the **test** sets. The former provides examples to train the algorithm; thus, the learned parameters are run on the validation set for model selection.
Having chosen the model that generates the lowest error, the procedure proceeds by computing the predictor's performance on the last set and use this measure as an estimate of the true error of the learned predictor.

Although simple to use and interpret, there are times when the procedure is not advisable, such as when the dataset is not balanced or when the dataset is too small. Indeed, there will not be enough data in the training dataset for the model to learn an effective mapping of inputs to outputs or, in the case of outliers in the training set, the resulting algorithm could be biased and ineffective when used in the field.

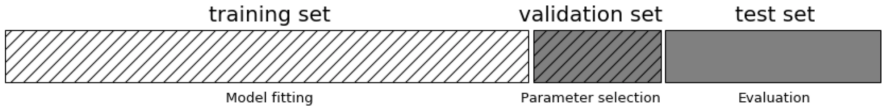

Figure 3: Train-Validation-Test Split

With *k-fold cross-validation*, the original training set with **m** examples is split into **K** subsets, known as folds, each of size **m/K**. The procedure consists of producing k different risk estimates, one for each of the k possible combinations where k-1 folds are retained together to train the algorithm **A** and the remaining is used to compute the error and so the performance of the model. For each fold, the rescaled error is measured

$$\hat{\ell}_{D_k}(h_k) = \frac{K}{m} \sum_{(x,y)\in D_k} \ell(y, h_k(x)) \tag{8}$$

Finally, the true risk estimate is obtained by averaging these errors

$$\hat{\ell}_S(A) = \frac{1}{K} \sum_{K=1}^{k} \hat{\ell}_{D_k}(h_k) \tag{9}$$

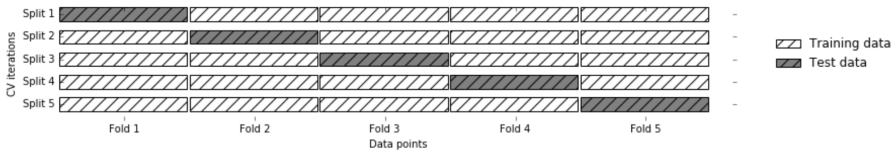

Figure 4: 5-fold cross-validation

To study the dependence of the cross-validated risk estimate on the parameter alpha of the ridge regression, the above cross-validation technique is run for each alpha. At the end, it is selected the model with the regularizer α that produces the smallest error.

3 Experiment

The analysis starts after the conclusion of the preprocessing phase. At first, the target variable **median_house_value** is separated by all others features so a random splitting in training and test set is performed. The split retains 70% of the records in the training set and the remaining 30% in the test set.

The first strategy exploits the train-validation-test paradigm to find the optimal configuration of the weights' vector. Hence, the already extracted training set is further splitted into train and validation sets. Therefore, given a list of possible values of alpha and the quadratic loss, the **ridge()** algorithm is fitted on the training and validated on the validation set, where the mean squared error (MSE) for each α is computed.
Hence, the loop produces for each regularizer the corresponding loss and gives the possibility to retrieve the coefficient associated with the smallest MSE.

Best alpha	1.0
MSE on train set	0.023112246626737616
MSE on test set	0.022801830398737656
R squared on the Train set	0.5942175171947703
R squared on the Test set	0.5912917252847574

The best coefficient is run again on the original training (train plus validation) and successively on the test set for evaluating its performance.

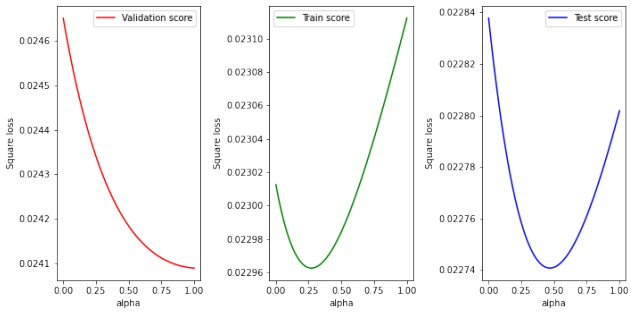

Figure 5: Train-validation-test split results

As shown, this technique does not perform very well. Indeed, the best optimal alpha, according to the results on the validation set, seems to be associated with high values of the loss both in the training and in the test set.
Indeed, to give a more accurate estimate of the risk, a 5-folds cross-validation is produced.

Best alpha	0.0
Train score	0.02283090763783867
CV risk estimate	0.02289458127301779

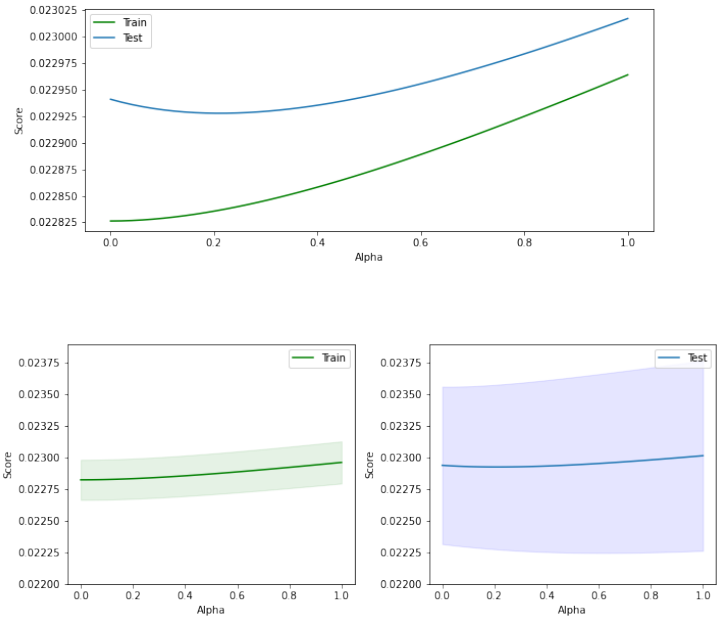

Figure 6: Cross validated risk estimate on alpha

The model is fit and evaluated for each alpha. In this way it was possible to study the dependence of the cross-validated risk estimate on the parameter α of the ridge regression but also to detect which of those parameters was linked to the lowest loss.
The cross-validated risk appears lower than the one computed with the training-validation-test method; moreover, both the training and the test error grow accordingly.
Taking into consideration the standard error too, it is possible to see that the risk estimate is more unstable than the training error one, but after all, balanced enough.

Additionally, a dimensionality reduction through PCA technique is carried out for the exact purpose of attempting a reduction in the risk estimate.

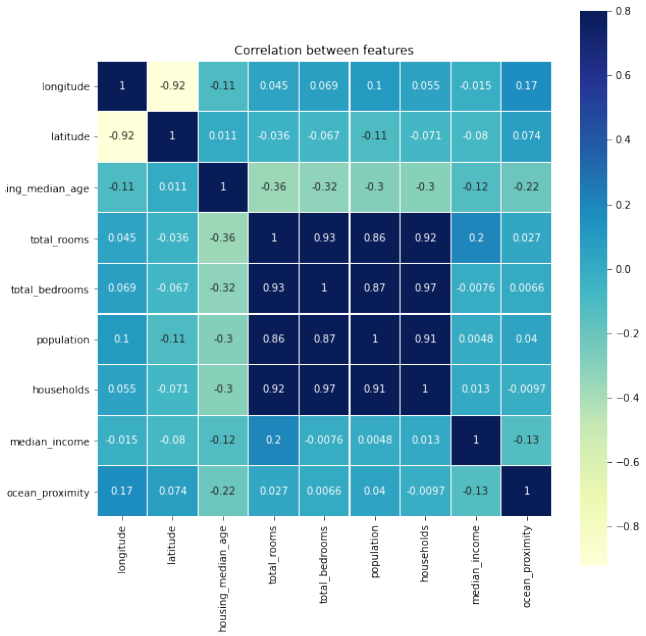

Figure 7: Correlation map

The correlation map displays that, except three out nine variables, all the features are uncorrelated.
The study of the cumulative variance explained by the principal components, obtained through the eigendecomposition of the features' covariance matrix, suggests that projecting the data from a 9-dimensional to a 5-dimensional space will preserve much of the original variance.

Explained variance taking five components 99.08030802546011

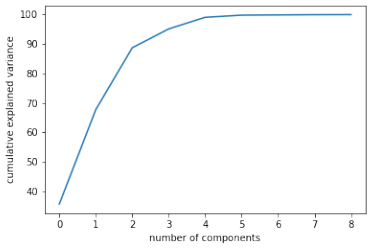

Figure 8: Variance explained

Once the data are projected onto the new space, a **train-validation-test validation** approach is accomplished.

Best alpha	1.0
MSE on train set	0.02410181573674267
MSE on test set	0.023496859957695918
R squared on the Train set	0.5768436194145075
R squared on the Test set	0.5788337634917583

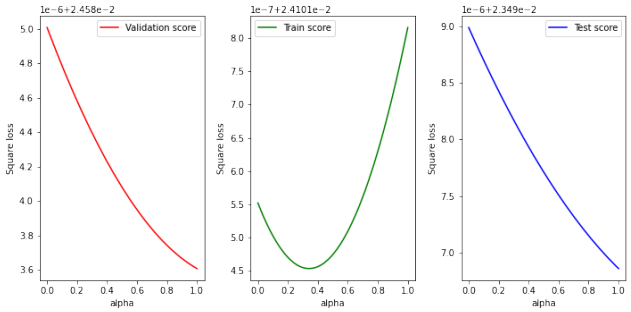

Figure 9: Train-validation-test split results after PCA

Again, the performance seems not to produce coherent results. Hence, a 5-folds cross-validation is performed for each alpha to end up with the following results:

Best alpha	0.0
Train score	0.023916073247367907
CV risk estimate	0.023931753117948717

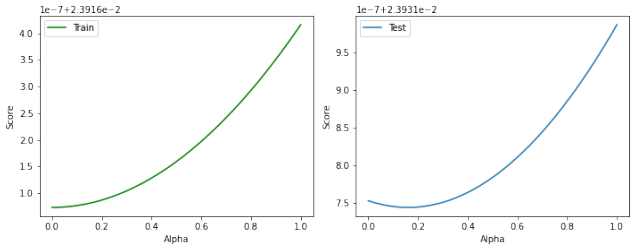

Figure 10: Cross-validated risk estimate after PCA

4 Conclusion

Overall the risk estimate appears to be more stable after the dimensionality reduction, despite a slight worsening of both the train and the test error.

	Before PCA	After PCA
Best alpha	0.0	0.0
Train score	0.02283090763783867	0.023916073247367907
CV risk estimate	0.02289458127301779	0.023931753117948717

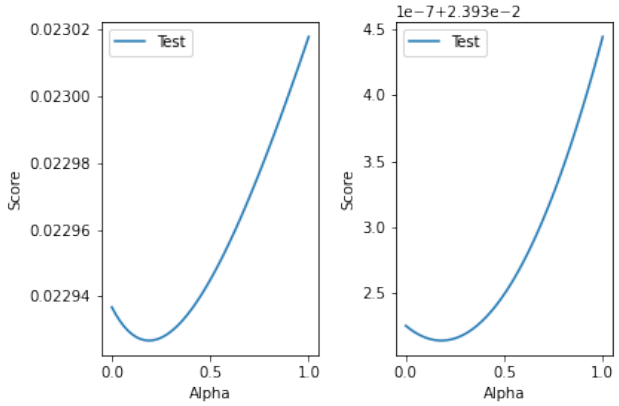

Figure 11: Risk estimate before and after PCA

Finally, selecting the best alpha, the model **ridge (alpha= best alpha)** is run once again on the entire dataset to compute both the quadratic loss and the goodness-of-fit metric R^2.

	Before PCA	After PCA
MSE on the entire dataset	0.02283704751736542	0.5965688411541219
R^2 on the entire dataset	0.023917637952508546	0.5774795148672578

References

[1] Shai shalev-Shwartz, Shai Ben-Dadid. 2014. *Understanding machine learning*