Housing Prices

Ridge and Lasso regressions for the prediction of the median house price:

a machine learning implementation

Anna Olena Zhab'yak[1]

University of Milan, Data science and Economics

and

Michele Maione[2]

University of Milan, Computer Science

(October 2020)

Key words: *machine learning, ridge regression, lasso regression, hedonic model*

Summary – This paper is trying to answer the question which the best regression model is to predict the median house price. The hedonic theory is exploited and models as the Ridge and the Lasso regression are used on a cross-sectional dataset of housing prices available at this link: https://www.dropbox.com/s/zxv6ujxl8kmijfb/cal-housing.csv. Applying the nested cross-validation we estimate the hypermeters and generate the best hyperparameter, then evaluate the model via K-fold cross-validation. PCA is implemented to improve the risk estimator. In Section 1 we introduce the problem of housing prices in the U.S. and the approach used in this work. Section 2 is dedicated to the literature about regressions in predicting the price of real estates and in Section 3 the theoretical notations are clarified to simplify the understanding of the concepts. Our experiment is described in section 4 and Section 5 provides the consequential critical comments and evaluations.

1. Introduction and description of the problem

The Hedonic theory identifies the attributes as implicitly embodied in goods and their observable market prices, so extending this concept to the housing prices we can see the attributes as the house's characteristics that are determinant for the final value. The Hedonic model exploits the consumer theory and Her willingness to pay depending on the utility gained from the bundle of aggregated attributes.

Our work starts from a real problem of housing prices in the United States, where the economical purposes and the low mortgage rates incentive a solid and hot real estate market[3]. Indeed, the U.S. is one of the most stable and secure countries for real estate investment in recent years[4]. It is estimated that household wealth is nearly 50% invested in real estate and the owner-occupied

[1] annaolena.zhab@studenti.unimi.it – 960298.
[2] michele.maione@studenti.unimi.it – 931468.
[3] Santarelli, 2020.
[4] Source: International Investor Survey.

housing rate in July 2019 was about 63.5%[5]. However, the U.S. real estate market was not always as reliable as today, indeed the sudden bubble of the housing market of 2006-2007 preceding the Great Recession and its subsequent burs is clear evidence of the system weaknesses. The speculation on the housing prices and their extremely high values is due to the lack of information caused by the manipulations of major players in the real estate sector[6]. For these reasons, the task of predicting the value of a house becomes crucial, as the constructed house price model can influence economic growth and improve the efficiency of the real estate market. An accurate prediction model is significant and helps to fill up an information gap for the prospective homeowners, policy-makers and other real estate market participants, such as mortgage lenders and insurers[7]. Modelling house prices presents some issues, for example, the median value might be extremely influenced by the value of the sold properties in the area with similar characteristics[8] or the prediction could become wrong due to exogenous factors influencing the prices. Indeed, the economic health reflects in the market according to the supply and demand law so any shock will affect the current prices. Moreover, working on a large dataset, like the one used in this work, can lead to the so-called multicollinearity of the features which tend to overfit when it comes to implementing the algorithm predicting the value. The classic OLS regression has the desired property of being unbiased, but it can suffer from overfitting and have a huge variance in those cases where features are highly correlated. To pull down the variance and obtain more biased estimator a regularization technique is necessary. The focus of this paper is therefore on two regularization techniques, the Ridge and Lasso regression. The Ridge regression[9] is a useful tool for improving prediction in regression tasks with highly correlated predictors. Lasso regression is also used to handle high dimensional databases where the features are correlated, and this technique shrinks some of them to zero, performing a feature selection with a consequent dimension reduction. Both methods act on the coefficients by introducing a penalty on them to make more effort to the most informative ones, this way minimizing overfitting of the data and solving the multicollinearity problem. The impact of each attribute on the predicted price is given by the value of the coefficient, higher coefficients mean higher influence. The penalty is the tool through which we perform the regularization, also called tuning parameter, which controls the bias-variance trade-off, and its selection is crucial. For choosing the regularization parameter in practice, nested-cross-validation is widely used.

2. Most important related works

Many works have been developed to predict the median house value with models of different complexity [see Manjula et al., 2017]. The concept of hedonic prices was developed by Rosen (1974), however the first implementing the hedonic model to the house sector was Lancaster (1966). Griliches (1971) provided the reading of a commodity, such as a house, as an aggregation of individual components or attributes. Timothy Oladunni & Sharad Sharma (2016) and Limsombunchai et al. (2004) have showed that the price of a property is predictable exploiting the hedonic theory, comparing the hedonic regression in comparison with other algorithms. Dubin (1998) has developed a work to predicted house prices using MLS data, even though exploiting different algorithms for the prediction, such as kriging algorithm to create an accurate spatial interpolation of house prices. Others as Xin and Khalid (2018) have used ridge and lasso regression to deal with multicollinearity of features on a time series database for predicting the housing price. Hoerl and Kennard (1970) firstly introduced the Ridge regression as biased estimator for non-orthogonal problems. The asymptotic properties of ridge have been widely studied, [see for e.g. Dobriban and Wager (2018), Dicker (2016)]. For the validation approach we refer to the cross-validation which biased estimation of the error is known [Hastie et al. (2009)],

[5] Source: United States Census Bureau.
[6] Oladunni, Timothy & Sharma, Sharad, 2016.
[7] Limsombunchai et al. (2004).
[8] The so-called sales comparison approach.
[9] Introduced by Hoerl and Kennard (1970).

since it uses a smaller amount of data than the entire dataset[10]. However, we can apply a bias-control, see Liu and Dobriban (2020), for example via k-fold cross validation, see Ray (2018), since there is an inverse relation between the k size and bias, if the first grows the latter goes down. Mishra et al. (2017) have clearly explained the intuition behind the PCA and the underlying algebra to rich these results. PCA was introduced by Pearson (1901) and Hotelling (1933) and it is largely used in a lot of fields. Gupta and Kabundi (2010) have implemented lasso, ridge and PCA to predict housing prices on a time series dataset, so they could control for the economic stochastic shocks.

3. Notation and relevant definitions - Regression

The goal of the regression is to generate a prediction $\hat{y} = f(w, x)$ such that the loss function $\ell(y, \hat{y})$ is small for most data points $x \in \mathcal{X}$, where $\hat{y} \in \mathcal{Y}$ is the prediction from the labels set $\mathcal{Y} \subseteq \mathbb{R}$, $w \in \mathbb{R}^d$ is the coefficient vector and $\mathcal{X} = \mathbb{R}^d$ the data domain; the prediction mistakes are a function of the difference $|y - \hat{y}|$.

3.1. Hedonic model

Following the hedonic theory, the housing price can be written as a function $f(.)$ in the following way:

$$P_i = f(s) \tag{1}$$

where s is the vector of all the objective attributes and P_i is the price of the i^{th} element of the data matrix $\mathcal{X}$. In this case, the price (our target variable) is a function of:

- longitude,
- latitude,
- housingMedianAge,
- totalRooms,
- totalBedrooms,
- population,
- households,
- medianIncome,
- medianHouseValue,
- oceanProximity.

3.2. Loss function

With loss function we denote the measure of how different the prediction of a hypothesis is from the true outcome. We use a nonnegative loss function to measure the discrepancy $\ell(y, \hat{y})$ between the predicted label $\hat{y}$ and the true label y. In the regression task we define the quadratic loss that is the squared distance between y and $\hat{y}$

$$\ell(y, \hat{y}) = (y - \hat{y})^2 \tag{2}$$

when $\hat{y} = y$ then $\ell(y, \hat{y}) = 0$ otherwise If $\hat{y} - y = c, \forall\, c \in \mathbb{R}^+$ and c is large then also $\ell(y, \hat{y})$ tend to be large. The mean of the squared error (MSE) will be used in the experiment.

3.3. Test error and training error

The split of dataset into two separate subsets is necessary in order to have some fresh data to estimate the predictive power of the algorithm. The dataset is divided in n elements for the test, and m elements for the training. Indeed, the validation is given by the test error which is:

$$\frac{1}{n}\sum_{t=1}^{n} \ell\left(y'_t, f(x'_t)\right) \tag{3}$$

The validation is done over a fitted predictor in the training set, and its power is given by the training error:

$$\hat{\ell}(f) = \frac{1}{m}\sum_{t=1}^{m} \ell\left(y_t, f(x_t)\right) \tag{4}$$

Total error is given by three elements:

- variance,
- bias,
- irreducible error.

The main idea is to derive a trade-off between the bias and variance, on order to optimize them both. More complex models present high variance and low bias since they fit good the true data but generalize worst.

[10] In other words, the algorithm has not enough data to train on and can be approximated.

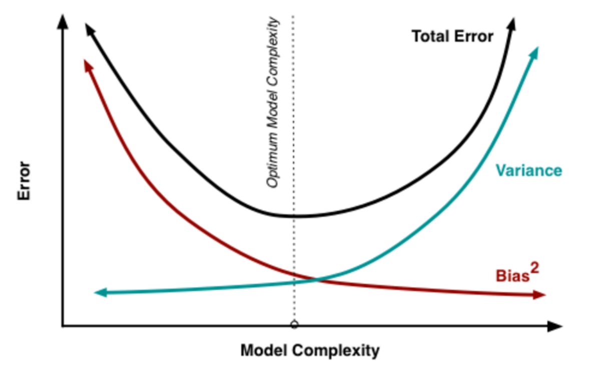

Figure 1 - the optimal choose for trade-off. Source: researchgate.net

3.4. Empirical Risk Minimization (ERM)

The Empirical Risk Minimization is a learning algorithm which returns some predictors f ∈ $\mathcal{F}$ given a set of predictors, that minimize the training error, given a non-negative real-valued loss function $\hat{\ell}$:

$$\hat{f} \in \underset{f \in \mathcal{F}}{\operatorname{argmin}} \hat{\ell}(f) \tag{5}$$

3.5. Statistical risk, Bayes optimal predictor and Bayes optimal risk

We use statistical learning to introduce the notion of expectations in estimating the loss since we need to assume the independence between the variables and the predictor we generate is based on this assumption. Let $h : \mathcal{X} \to \mathcal{Y}$ be the predictor that maps data points to labels. **The statistical risk** is then defined as the expectation of the loss function among D, the distribution from where the random sample of data points and labels were drawn:

$$\ell_D(h) = \mathbb{E}[\ell(Y, h(x))] \tag{6}$$

where $h(x)$ is the predicted $\hat{y}$. We then define the **Bayes optimal predictor** as the function f^* which minimize the overall training error $\ell_D(h)$, given the conditional probability among all predictors given that our data point is x:

$$f^*(x) = \underset{\hat{y} \in Y}{\operatorname{argmin}} \mathbb{E}[\ell(Y, \hat{y}) \mid X = x] \tag{7}$$

The **Bayes optimal risk** is the expectation over the loss function of the Bayes optimal predictor and following the same logic as before we have that the Bayes risk is smaller than the other risks:

$$\mathbb{E}\left[\ell\left(Y, f^*(x)\right)\right] \leq \mathbb{E}[\ell(Y, h(x))] \tag{8}$$

Coming to our regression problem with the squared loss, the Bayes optimal predictor is:

$$f^*(x) = \underset{\hat{y} \in Y}{\operatorname{argmin}} \mathbb{E}[(Y - \hat{y})^2 \mid X = x] \tag{9}$$

minimizing this quantity[11], we have:

$$f^*(x) = \mathbb{E}[Y \mid X = x] \tag{10}$$

and the Bayes risk becomes the expectation of (6):

$$\mathbb{E}\left[\left(Y - f^*(X)\right)^2 \,\middle|\, X = x\right] = \operatorname{Var}[Y \mid X = x] \tag{11}$$

3.6. Regressions – Linear, Ridge, Lasso

Ridge and Lasso regression modify the standard linear regression by introducing a positive constant as regularization parameter. Indeed, the objective function to minimize under these solutions is **RSS[12] + penalty**, and the penalty differs for the two methods. Starting from the classical linear model we have:

$$y_i = \bar{x}_i^\top w \tag{12}$$

let be the data domain $\mathcal{X} = \mathbb{R}^d$ and $x = (1, x_1, \dots, x_d)$[13] a row vector of $\mathcal{X}$. The linear predictor is a linear function $h : \mathbb{R}^d \to \mathbb{R}$, and for an activation function $f : \mathbb{R} \to \mathbb{R}$ we can write as follows:

$$h(x) = f(w^\top x) \tag{13}$$

where $w \in \mathbb{R}^d$ and $w^\top x = \sum_{i=1}^d w_i x_i$.

The Bayes optimal risk is given by

$$f^*(x) = \mathbb{E}[y \mid X = x] \tag{14}$$

and it is also an empirical risk minimization to $(x_1, y_1) \cdots (x_m, y_m)$ is

[11] By taking the derivative w.r.t $\hat{y}$, since $f^*(x)$ is differentiable.

[12] Sum of the squared residuals used for the classical OLS.

[13] Add one extra feature to stabilize the prediction.

$$\hat{w} = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \frac{1}{m}\sum_{t=1}^{m}(w^\top x_t - y_t)^2 \qquad (15)$$

Since we can rewrite these terms in vector notation, we have

$$\hat{w} = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \|v - y\|^2 \qquad (16)$$

for $v = (w^\top x_1, \dots, w^\top x_m)$ the vector of predictions and $y = (y_1, \dots, y_m)$ the vector of real labels and for $v, y \in \mathbb{R}^m$.

In matrix notation we have S the design matrix $S \in \mathbb{R}^{m\times d}$ with d features and m observations x_i that are rows of $S^\top$, and therefore the vector becomes $v = Sw$. Applying the ERM we derive

$$\hat{w} = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \|Sw - y\|^2 \qquad (17)$$

The solution to the ERM is the minimization of this convex function $F(w) = \|Sw - y\|^2$ using the Euclidean norm. To solve the problem in linear regression we can use the closed form solution, or the gradient descend.

If $S^\top S$ is a non-singular matrix[14], and the conditions of the general position holds, the solution of the ERM is the closed form:

$$\nabla F(w) = 2S^\top(Sw - y) = 0 \qquad (18)$$

$$\hat{w} = (S^\top S)^{-1} S^\top y \qquad (19)$$

In some cases, the linear regression performs well on the training data, having a low bias, but it gives a non-accurate estimate on different data. The reason why it occurs it is because of multicollinearity of the prediction vectors (as known as non-orthogonality)[15] . More in general with d large or n small, the risk that the model can overfit[16] the data is high. The OLS estimator $\hat{w}$ therefore is unbised but have a huge variance and it is not stable. To overcome this problem, Ridge and Lasso regression help to prevent over-fitting which results from simple linear regression. We introduce a regularized parameter α which adds some bias[17] whereas pushing the variance down. This also controls the model complexity, indeed the value of α has a direct relation with the complexity. This occurs to find the best trade-off between bias and variance to get to that sweet spot for having good predictive performance[18]. The two methods work similarly but lead to different results, this happens because of the divergent formulas.

3.6.1. Ridge solution

Ridge regression uses the penalty multiplied by the square of the magnitude of the coefficients, also known as L2 regularization.

The ERM functional of Ridge regression is

$$\hat{w}_\alpha = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \|Sw - y\|^2 + \alpha\|w\|^2 : \forall\alpha > 0 (20)$$

for $\alpha \to 0$, $\hat{w}_\alpha \to \hat{w}$ so the solution leads the linear regression, for $\alpha \to \infty$ the coefficient tend to a zero vector and the line becomes flatter, shrinking the linear regression solution towards to zero.

To optimize the objective function, we take the gradient as before and solve for w to find a suitable value:

$$\nabla F(w) = \|Sw - y\|^2 + \alpha\|w\|^2 \qquad (21)$$

$$2S^\top Sw - S^\top y + 2\alpha w = 0 \qquad (22)$$

$$(S^\top S + \alpha I)w = S^\top y \text{ [19]} \qquad (23)$$

The new estimated parameter becomes:

$$\hat{w}_\alpha = (S^\top S + \alpha I)^{-1} S^\top y \qquad (24)$$

This is the so called closed-form solution and α is the one measuring the stability of the procedure.

3.6.2. Lasso solution

Least Absolute Shrinkage and Selection Operator, or simply Lasso, is slightly different from the previous because the penalty is multiplied by the absolute value of the magnitude of coefficients, also known as L1 regularization

[14] This happens if the data points span $m \geq d$.
[15] Hoerl and Kennard, 2010.
[16] Overfitting: the algorithm performs very good on training data but cannot be generalized to a new bunch of data.
[17] Bias is how well the fit correspond to the true value.
[18] See graphic 1 in this paper.
[19] Adding the identity matrix fixes the invertibility problem, always compute inverse, and this is more stable solution.

$$\hat{w}_{Lasso} = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \|Sw - y\|^2 + \alpha|w| \,:\, \forall \alpha > 0 \quad (25)$$

For $\alpha \to \infty$, $\hat{w}_{Lasso} = 0$. The Lasso procedure encourages simple, sparse models[20], indeed some coefficients can become zero and be eliminated from the model, this way performing a feature selection. The shrinkage amount is given by the value of tuning parameter α. If α increase, we have some parameters go straightway to zero.

The optimization of a non-differentiable function as Lasso solution is done by a proximal gradient descend approach[21].

The first step is to take the gradient descend for current $w^{(k)}$ vector and form a new vector $z^{(k)}$:

$$z^{(k)} = w^{(k)} - \eta X^{\top}\left(Xw^{(k)} - y\right) \quad (26)$$

Where η is the step size and k is the moment we are considering. Then solve the proximal regularize problem for $w^{(k+1)}$ as follows:

$$w^{(k+1)} = \underset{w\in\mathbb{R}^d}{\operatorname{argmin}} \left\|z^{(k)} - w\right\|_2^2 + \eta\alpha|w| : \alpha,\eta > 0 \quad (27)$$

This is a scaler minimization problem indeed we can rewrite it as:

$$\min_{w_i} \sum_{i=1}^{m} \left(z_i^{(k)} - w_i\right)^2 + \alpha\eta|w_i| \quad (28)$$

Since we have an absolute value for $|w_i|$ we consider two cases:

Case 1 $w_i \geq 0$

$$\min_{w_i} \left(z_i^{(k)} - w_i\right)^2 + \alpha\eta w_i \quad (29)$$

differentiate with respect to w and solve:

$$-2(z_i - w_i)^2 + \alpha\eta = 0 \quad (30)$$

since we have the non-negativity constraint over w_i:

$$w_i = \begin{cases} z_i - \frac{\alpha\eta}{2}, & z_i > \frac{\alpha\eta}{2} \\ 0, & else \end{cases} \quad (31)$$

Case 2 $w_i < 0$

$$\min_{w_i} \left(z_i^{(k)} - w_i\right)^2 - \alpha\eta w_i \quad (32)$$

differentiate with respect to w and solve for it:

$$-2(z_i - w_i)^2 - \alpha\eta = 0 \quad (33)$$

$$w_i = \begin{cases} 0, & z_i + \frac{\alpha\eta}{2} > 0 \\ z_i + \frac{\alpha\eta}{2}, & else \end{cases} \quad (34)$$

This solution is also known as the “soft threshold” operation. More compactly we can rewrite the values for w_i as follows:

$$w_i = \begin{cases} 0, & \frac{-\alpha\eta}{2} < z_i < \frac{\alpha\eta}{2} \\ z_i - \frac{\alpha\eta}{2}, & z_i > \frac{\alpha\eta}{2} \\ z_i + \frac{\alpha\eta}{2}, & z_i < \frac{-\alpha\eta}{2} \end{cases} \quad (35)$$

The common point of these two methods is that adding the regularization parameter to the cost function the algorithm is forced to pick the lowest weights, indeed the goal is to ensure a small coefficient through this regularization parameter. The main difference is that many coefficients are exactly zeroed under lasso, which is never the case in ridge regression where there is not any elimination of coefficients. Moreover, Lasso arbitrarily selects any one feature among the highly correlated ones, leading to a higher variance then Ridge regression.

3.7. Cross-validation

The performance of an algorithm must be evaluated on a new data order to see if it works even on a not trained data. The most common approach used is to split the dataset into three subsets, training, validation set and test set. The first set will be used to train the algorithm and the second to evaluate its performance, finally the best output according to the validation set will be evaluated on the final test set to estimate the risk.

[20] Stephanie Glen. “Lasso Regression: Simple Definition” from “StatisticsHowTo.com: Elementary Statistics for the rest of us!”.

[21] This method requires a convex function. Lasso lies in this category.

Cross-validation (CV) is one of the techniques used to test the effectiveness of a machine learning model and it is also a re-sampling procedure used to evaluate a model if we have a limited data. The CV risk estimate therefore becomes $\mathbb{E}[\ell_D(A(s))]$, where $A(s) = \hat{h}$ is our estimated predictor on the training set. This approach however can be biased[22], therefore a K-Fold cross-validation is largely used for evaluating the accuracy of model.

This approach splits the training set into K-subsets and each fold at each interaction is used for testing while the remaining are used to training. This ensures that every observation from the original dataset has the chance of appearing in training and test set, this way decreasing the bias of the CV[23].

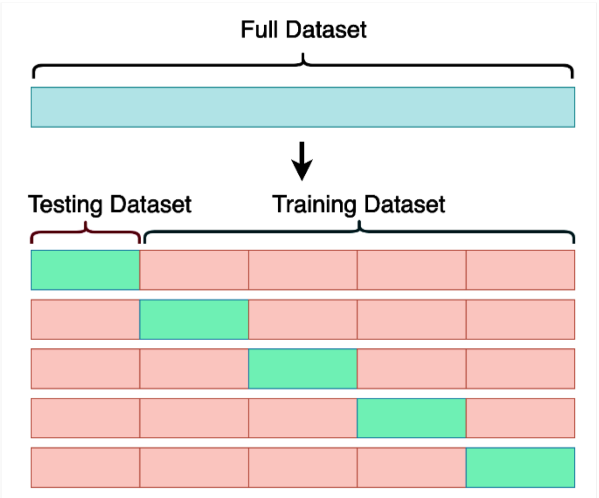

Figure 2 - K-Fold. Source: mlfromscratch.com

We have D_k the testing part while $S^{(k)}$ the train part. The number of subsets obtained is k and we get a value for $h_k = A(s^k)$ computed on the training part. The test loss according to k-fold is

$$\hat{\ell}_{D_k}(h_k) = \frac{K}{m} \sum_{(x,y)\in D_k} \ell(y, h_k(x)) \qquad (36)$$

After all interactions (form 1 until k^{th} interaction) we collect the obtained values, and the CV risk estimate is the average of all errors:

$$\frac{1}{K}\sum_{k=1}^{K} \hat{\ell}_{D_k}(h_k) \qquad (37)$$

The complete algorithm works like this:

```
1. Shuffle the dataset randomly,
2. Split the dataset into k groups,
3. For each unique group:
    1. Take the group as a hold out
       or test data set,
    2. Take the remaining groups as
       a training data set,
    3. Fit a model on the training
       set and evaluate it on the
       test set,
    4. Retain the evaluation score
       and discard the model.
4. Summarize the skill of the model
   using the sample of model evalua-
   tion scores[24].
```

However, this approach cannot be used to tune the hyperparameter that is involved into the estimation of the parameter such that to obtain $\min_{\theta\epsilon\Theta} \ell_D\left(h_S^{(\theta)}\right)$ where $h_S^{(\theta)} = A_\theta(S)$. The choice of it must be done before the choice of the training set on which the algorithm will be learned. To do this we use another variation of CV that is the Nested cross-validation, which puts together the previous two techniques in such a way to solve the hyperparameter choice problem. The set is split into k-fold, one for testing and the rest for training. After two loops are made:

- An internal cross validation on the training part of the initial split is also split in k-folds and run the validation for a grid of all $\theta \in \Theta$ choosing the best one.
- External cross validation is done in order to avoid the dependency over the training set, and it takes the best output from the inner loop and run it on the entire training

[22] Because we could be just lucky with the train set we have chosen and the risk estimated on the test set suffer of this distortion.

[23] As $k \to N$, leads to "Leave-one-out cross-validation", the leave one out approach, where the number of sets equal the number of observations. On the contrary for smaller values of k we have the CV approach.

[24] Brownlee J., 3 August 2020. Source: machinelearningmastery.com.

set. The risk is therefore evaluated on the remaining test set of the current external fold.

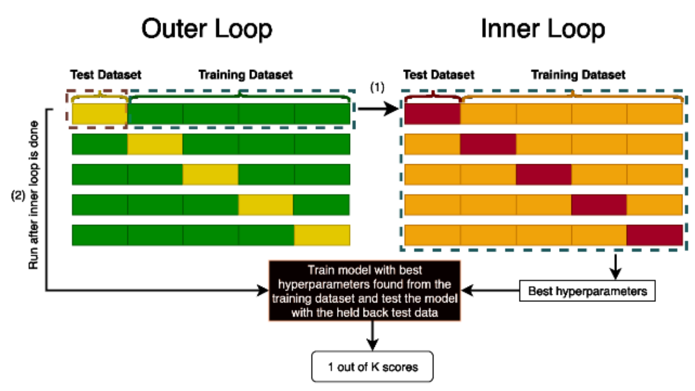
Outer Loop
Inner Loop
Test Dataset
Training Dataset
(1)
Test Dataset
Training Dataset
(2)
Run after inner loop is done
Train model with best hyperparameters found from the training dataset and test the model with the held back test data
Best hyperparameters
1 out of K scores

Figure 3 - nested cross-validation. Source: mlfromscratch.com

This procedure is run on each fold and at the end we pick up the expected value of the average risk as follows

$$\mathbb{E}\left[\min_{\theta\epsilon\Theta}\ell_D(A_\theta)\right] \qquad (38)$$

3.8. Principal component analysis

Principal Component Analysis is a technique used for dimensionality reduction. Its goal is to reduce the number of features through a combination of the original data variables, in this way keeping most of the original information. The standardization is needed before the implementation if PCA, which finds the eigenvalues and eigenvectors of the correlation matrix[25]. The selection of the principal components is based on the variance and they will be then independent one from the other. The feature that cause more variance is the first principal component, and so on until we reach a suitable number of explained variance by the principal components. A good way is to plot the variance against principal components and ignore the principal components with diminishing values. This way we reduce the variance, and we can improve the stability of the regression by solving the multicollinearity. It is a way to identifying patterns in data and expressing the data in such a way as to highlight the similarities and differences.

4. Proof of a technical result

The demonstration of our experiment and all the critical considerations will be described here. This material is also available on GitHub at this url: https://github.com/mikymaione/HousingPrices we have created a Jupiter Notebook to illustrate the procedure followed step by step at this url: https://github.com/mikymaione/HousingPrices/blob/master/SourceCode/HousingPrices/main.ipynb

4.1. Data pre-processing

Before performing the analysis and implementing the regression, pre-procession of data is necessary. The dataset presents features that cannot be compared in a linear Euclidian space, therefore geometry is not working properly on this row data. Indeed, to learn the algorithm, we need to encode the features and raise them to a homogeneous level, so we can compare them[26]. The dataset contains 20,640 observations and 10 features for each house including the median house value which is the target value that we are trying to predict. Firstly, we create the two constants of the designed matrix and the target variable: X, y. Even though among hedonic model literature[27] there is a concern about the statistical insignificance of some features such as household size, we use two approaches:

- keep all the features in regressions to have as much information as possible,
- use some unsupervised techniques to decide which feature to drop such as PCA.

4.1.1. Missing values

The missing values must be handled to avoid errors in the execution of the code, so they are filled with the mean value of the corresponding column.

[25] The eigenvectors of the Covariance matrix are the directions of the axes where there is the most variance (most information).

[26] For example, longitude and households are features with real numbers however they have different interpretations.

[27]See Berna and Craig, 2016.

4.1.2. Categorical features

There is a categorical feature which represents the distance from the ocean, we transform the elements of the column into columns dummies and assign it to the data set[28].

4.1.3. Standardization

The standardization is done by subtracting the mean and divided by the variance. In this way we have μ = 0 and σ = 1. This procedure is needed as both L1 and L2 assume that all features are centred around 0 and have unit variance. In general, this is important for those algorithms that use Euclidian norm and for PCA implementation.

4.1.4. Correlation matrix

We explore the symmetric correlation matrix between the features to see graphically how they move together. The used scoring is Pearson's coefficient r and $r \in [-1;\ 1]$, if the value is closer to $|1|$ there is more correlation, and the sign gives the direction. Darker and lighter colours on this map are the two extremes.

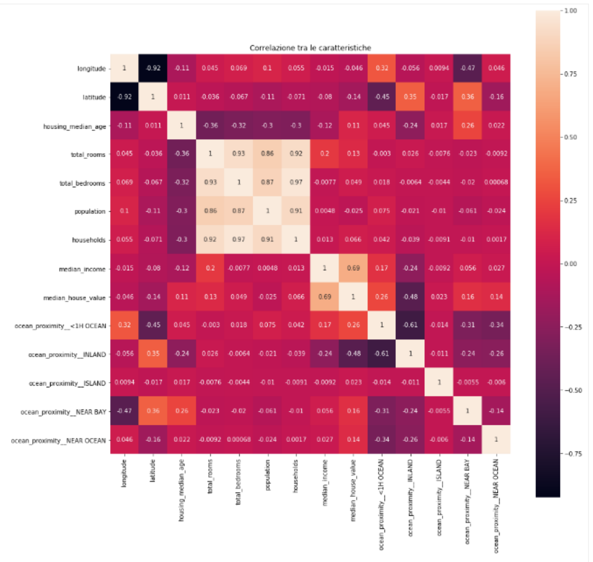

Figure 4 – Correlation matrix between features – Pearson coefficient

We have enough evidence that there exists statistical relationship between the variables[29]. Therefore, our dataset is suitable for decomposition into its principal components to increase convergence speed and eliminate collinearity by finding the core components of the datasets.

4.2. Model tuning

Model tunning consists in the choice of the parameters to use into regression. In our case is the α hyperparameter for both Ridge and Lasso regression. The train-test split is done 80% - 20%[30].

4.2.1. Scoring

Mean squared error is implemented as scoring, and it takes bigger values more than proportionally if the error in prediction increases.

4.2.2. Choosing the set of parameters alpha

In order to obtain a reasonable amount of information to determine a certain $f^* \in \mathcal{F}$ where f^* is the function that minimize the training error, we need to set different values of the tuning parameter to find out the best one. A larger value of α leads to a high bias but a low variance. On the other hand, for small values of α the variance increases, and bias go down. We perform the analysis on α with the relative mean squared error on the training data, comparing the Lasso and Ridge solutions. Nested cross validation gives us the same best value and the non-nested cross validation. We are plotting the parameter using the non-nested CV and then comparing the two methods.

Ridge regression

We use a logarithmic range $ln|F|$. Training size m is bigger than the $ln|F|$ in order to avoid underfitting.

[28] This is called hot encoder technique; we do not worry about adding extra dimensions as the dummy variable sets to zero the features that do not belong to the given observation.

[29] See the lighter square 4x4 in the middle.

[30] 80/20 rule: following the Pareto principle.

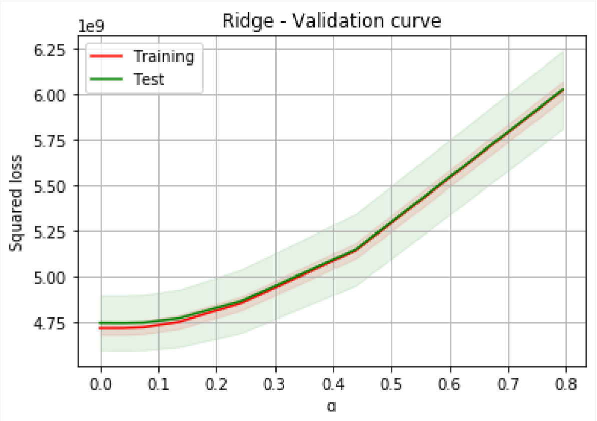

Figure 5 - Ridge regression: validation curve with α ∈ (0; 0.8)

As we can see the optimal value for the hyper-parameter that optimize the squared loss is in between the range (0; 0.1], after that the squared lost increases. The generated best value of the penalized term is **0.04079**, according to nested and non-nested validation curve.

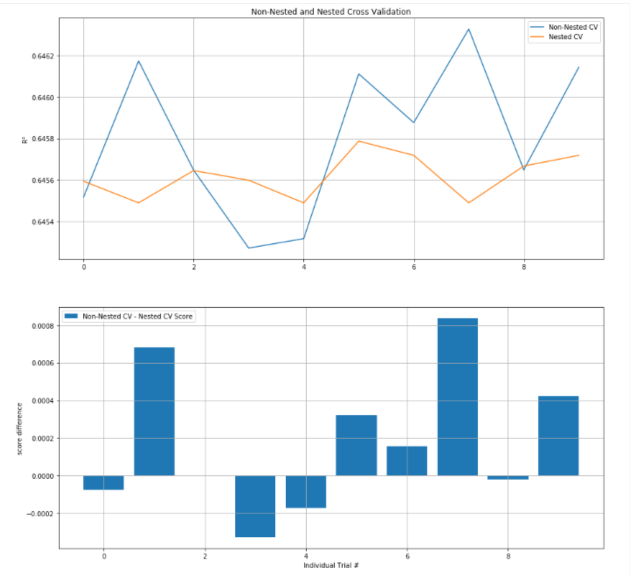

Figure 6 - Ridge: nested cross-validation (red) vs non-nested cross-validation (blue) of the best α

This graph shows how the nested cross-validation has a quite stable result and it is independent from the individual trial. On the other hand, the non-nested cross-validation is influenced by the individual trial interaction with upper and lower bounds that have a huge variance.

Lasso regression

For Lasso implementation we have these values:

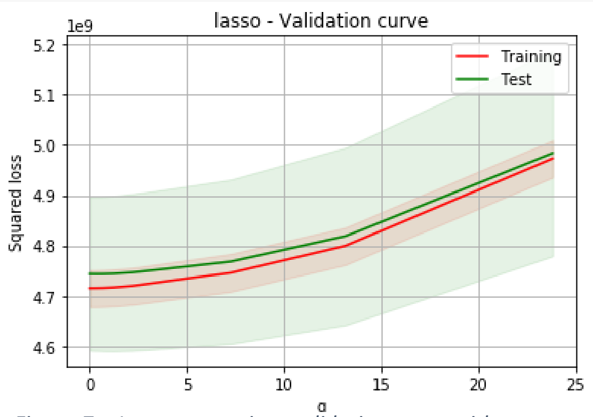

Figure 7 – Lasso regression: validation curve with α ∈ (0; 24)

In lasso regression this increase is smoother, and the loss is more stable for α in between (0; 1]. The values of used alpha are linear, and the best alpha according to the nested-cross validation and non-nested the validation curve is **0.37212**.

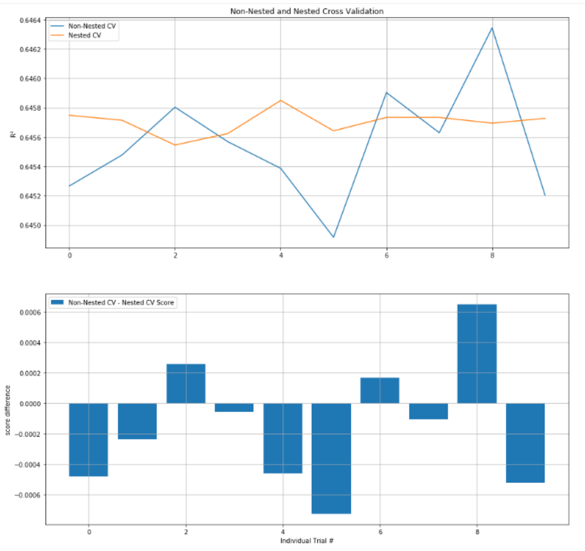

Figure 8 – Lasso: nested cross-validation (red) vs non-nested cross-validation (blue) of the best α

This graph shows the path for the nested and non-nested cross-validation, and again the latter performs worst given the variation of the individual trials. The nested cross-validation still being more stable and therefore not dependent on the trials.

4.3. Learning algorithm

Once we have determined the hyperparameter, the optimization of the learning algorithm is done. For Ridge regression we use the Cholesky method, that is the closed form. For Lasso regression we apply the proximate gradient descend[31]. This procedure is internal to the Ridge and Lasso function, and the learning algorithm is the output.

4.3.1. Ridge learning algorithm

We fit the best α to plot the learning curve performance. Training error becomes larger when iterations are increased, and the test error is higher as we could always expect better performances on the training set. As we see the overfitting disappears as we increase the training size, it starts going down from almost 2,000 training size, and it is improving with the training size growth exhibiting a stable squared loss (4.86 in Figure 9).

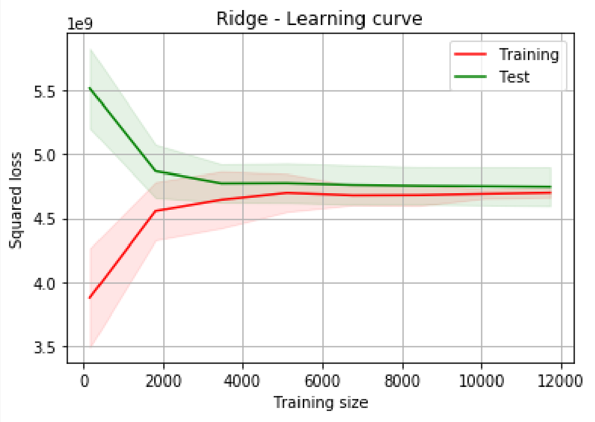

Figure 9 - Ridge regression: learning curve with different training set sizes

In this plot (Figure 10) we can visualize how the predicted value differs from the real values. This is done with fit and predict functions.

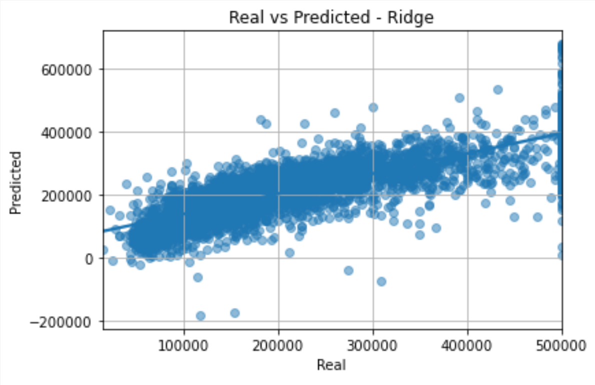

Figure 10 - Ridge regression: scatter plot prediction vs test

The prediction is more consistent with lower prices and becomes sparser for higher values, this can be caused by the presence of the outliers. The R2 is around 63%.

Here (Figure 11) we can see the magnitude of each coefficient, and its power on the prediction of the target variable.

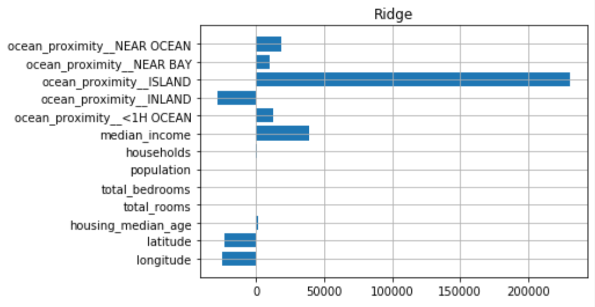

Figure 11 - Ridge: coefficients magnitude

Looking at the coefficients of Ridge regression we can conclude that *household size* is statistically insignificant as the literature suggests[32]. On the other hand, the driven force is the house location in *island*, which presents a direct and positive impact on the predicted prices. Also, the *median income* causes higher prices. On the other hand, the *inland* location has a negative impact on the prices, so as for *latitude* and *longitude.*

4.3.2. Lasso learning algorithm

The learning curve of Lasso regression is showed below (Figure 12).

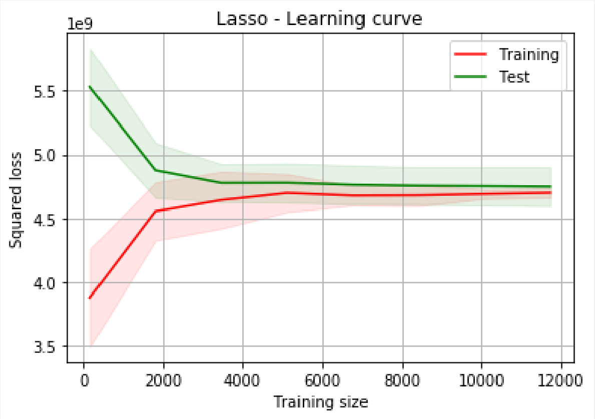

Figure 12 - Lasso regression: learning curve with different training set sizes

Similarly to the previous Ridge regression, the error is going down after a training size of 5,000 and

[31] This is possible because we take an approximation of the gradient since Lasso function is not derivable.

[32] Berna and Craig, 2016.

becomes stable around 4,87. The variance is contained and overfitting disappears as the training size increases.

The coefficients magnitude is shown in this graph.

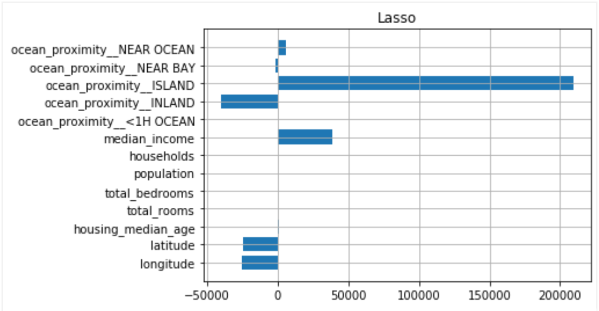

Figure 13 - Lasso: coefficients magnitude

Island location has a huge power in the prediction, it means that it drives the predicted values. *Inland* influence negatively, so as *longitude* and *latitude*[33]. *Median income* is the second driven force of the housing price prediction. Some of the coefficients them are like the ones of Ridge regression in Figure 11. However, values differ due to the shrinkage power of the Lasso penalty, for example Near bay location here is a negative number.

4.4. Principal Component Analysis

The following graphs[34] show how the PCA works when it is implemented on this database. The differences between the points on the 1st pc (plotted on x-axes) are more significant than the differences between the 2nd pc (y-axes).

As we can see in Figure 14 there are two main predictors that cause respectively the first largest variance and the second variation. The negative values mean the existence of an inverse correlation between the factor PCA and the variables. The group of points that are close to each other are more similar.

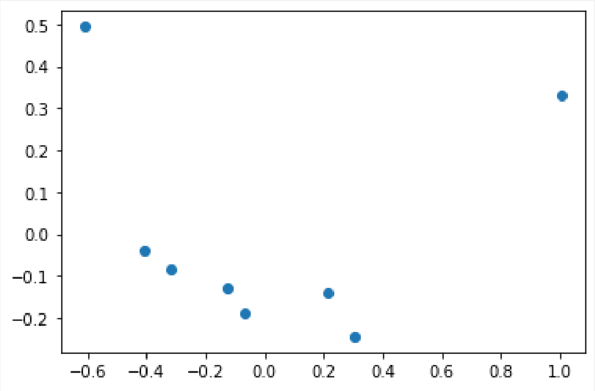

Figure 14 - PCA for 2 principal components on the axes. In the graph we have the predictors and their contribution to the variance.

In the next plot we see how the cumulative variance is explained by each feature (in our case 13).

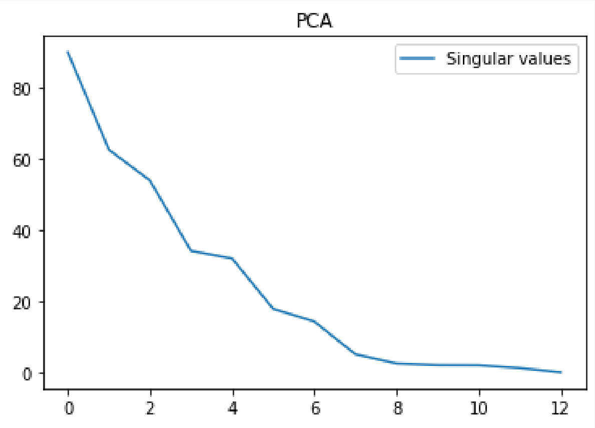

Figure 15 - PCA for singular values vs cumulative variance

After 3 features the variance significatively drops and above 8 features we do not gain more information, indeed the variance is almost 0. We will implement the decomposition on 8 features and plot the learning curve using our learned variables of the Ridge regression and projecting the datapoints, as in **Error! Reference source not found.**.

[33] We can thing about them as the two faces of the same coin, an increase in latitude and longitude means a geographical location more inland and faraway from the ocean.

[34] Cesa-Binchi, 2020, Linear regression and Ridge regression. https://github.com/nicolo63/CDS/blob/master/Part_10.ipynb

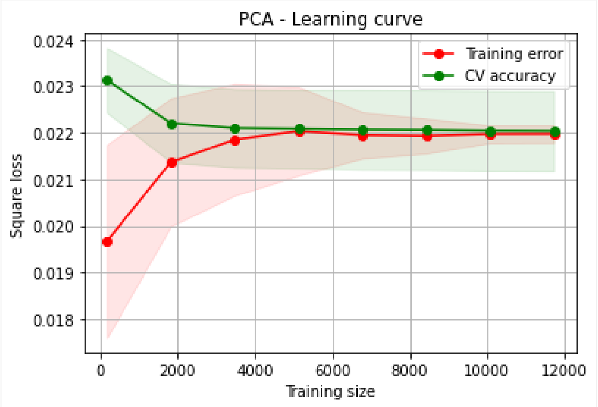

Figure 16 - PCA: test and train loss for the learned model using 8-PCA decomposition

The performance of the curve is quite worst comparing with the Ridge regression curve using an equivalent measurement of the error as in Figure 17: the variance of PCA learning curve is high and the risk estimate is not improving.

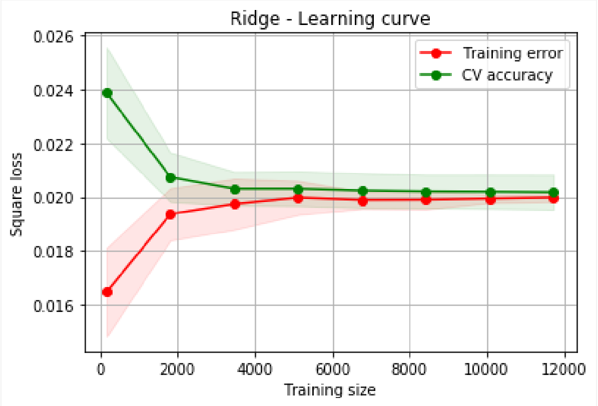

Figure 17 - Ridge learning curve

In the following graph we plot again the principal components using the learned variables. Each point is a predictor that we have learned.

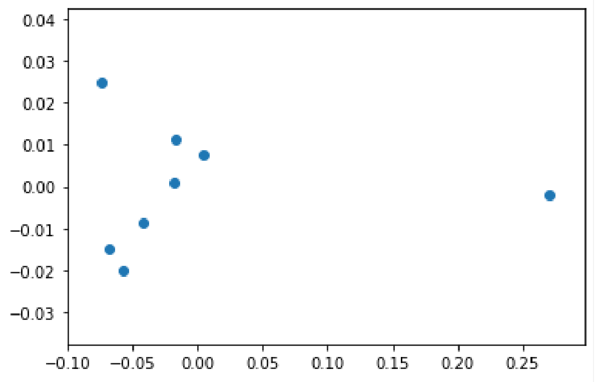

Figure 18 - Plot of the learned model after PCA

As we can see the spread is not improved crucially, we still have just one variable that causes the first large variance, and another for the second large variance. However, there are 2 main groups among the second main variance: the three variables that cluster near 0 are similar and have less variation, and 3 variables in the negative part of the graph.

5. Some critical considerations

We have performed two different regression on the dataset oh houses, and it results in a very similar performance for both Ridge and Lasso regularizations, this can be because in both methods we are using the best nested cross- validated penalized parameter. Indeed, the predicted value by Lasso model in the belove Figure 19 is almost the same as the Ridge prediction in Figure 10. However, comparing the risk estimator, the Ridge regression performs slightly better than Lasso (4,86 vs 4,87).

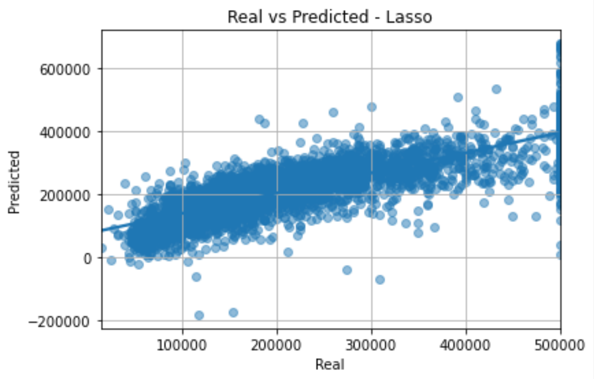

Figure 19 - Lasso: scatter plot prediction vs test

Moreover, we have tried to improve the risk using PCA, but the evidence shows that PCA do not improve the risk estimate. The regularized forms are enough to have an accurate prediction of the housing prices. In general, for this specific database we have noticed if we drop some feature via variable selection or some elements such as outliers, the predictive power is poor. Indeed, the performance is better when all the features are considered in the prediction as we have done in this experiment.

6. Bibliography

- Berna K., Creig W., *“Defining spatial housing submarkets: Exploring the case for expert delineated boundaries”,* SAGE journals, 2016.
- Calhoun C. A., *“Property Valuation Models and House Price Indexes for The Provinces of Thailand: 1992 –2000”*, Housing Finance International, 2003
- Dicker Lee H. *“Ridge regression and asymptotic minimax estimation over spheres of growing dimension”*, 2016.
- Dobriban E., Wager S., *“High-Dimensional Asymptotics of Prediction: Ridge Regression and Classification”,* The Annals of Statistics.46, 2015.
- Dubin R., *“Predicting House Prices Using Multiple Listings Data”*, The Journal of Real Estate Finance and Eco-nomics. 17, 35-59, 1998
- Frew J., Wilson B., *“Estimation The Connection Between Location and Property Value”*, Essay in Honor of James A.Graaskamp, Boston, MA: Kluwer Aca-demic Publishers, 2000.
- Griliches Z., *"Hedonic Price Index-es and the Measurement of Capital and Productivity: Some Historical Reflections"*, Fifty Years of Economic Measurement: The Jubilee of the Conference on Research in Income and Wealth, 185-206, National Bureau of Economic Research, Inc ., 1991.
- Gupta R., Kabundi A., *“Forecasting Real U.S.House Prices: Principal Components Versus Bayesian Regressions”*. International Business & Economics Research Journal, 2010
- Hoerl A. E., Kennard R. W., *“Ridge Regression: Biased Estimation for Nonorthogonal Problems”*, Technometrics, Vol. 12, No. 1, pp. 55-67, American Statistical Association and American Society for Quality Stable, 1970.
- Hotelling H., *“Analysis of a complex of statistical variables into principal components”*, J Educ Psychol. 25: 417-441, 1933.
- Liu S., Dobriban E., *“Ridge Regression: Structure, Cross-Validation, and Sketching”*, 2020.
- Manjula R., Shubham J., Srivastava S., Pranav K., *“Real estate value prediction using multivariate regression models”*, IOP Conference Series: Materials Science and Engineering. 263, 2017.
- Pearson K., “On lines and planes of closest fit to systems of points in space”, Philosophical Magazine 2(11):559-572, 1901.
- Ray, *“Improve Your Model Performance using Cross Validation (in Python and R)”,* 2018
- Rosen S., *“Hedonic Prices and Im-plicit Markets: Product Differentiation in Pure Competition”,* Journal of Political Economics, 82: 34 – 55, 1974,
- Santarelli, *“US Housing Market Forecast 2020 & 2021: Crash or Boom?”* Norada Real Estate, 2020.
- Seng X., Kamil K., *“Model-ling House Price Using Ridge Regression and Lasso Regression”*, International Journal of Engineering & Technology. 7. 498, 2018.
- Sidharth M., Uttam S., Subhash T., Sanjoy D., Devi S., Reshma S., Sasmita P., Menalsh L., *“Principal Component Analysis”*, International Journal of Livestock Research, 2017.
- Timothy O., Sharad S., *“Hedonic Housing Theory – A Machine Learning Investigation”*, 2016.
- Visit L., Christopher G., Minsoo L., *“House Price Prediction: Hedonic Price Model vs. Artificial Neural Network”*, American Journal of Applied Sciences, 2004.

7. Sitography

- afire.org
- builtin.com
- census.gov
- datacamp.com

- jstor.org
- github.com
- machinelearningmastery.com
- mlfromscratch.com
- noradarealestate.com
- projecteuclid.org
- psu.edu
- researchgate.net
- stackabuse.com
- statisticshowto.com
- towardsdatascience.com

8. Table of figure

9. Copyright

We declare that this material, which We now submit for assessment, is entirely our 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 our work. We 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 us or any other person for assessment on this or any other course of study.

9.1. MIT License