Analysis of Kernel Ridge Regression and Error Bound

Adilkhan Bakridenov [*] **Nurmadi Nurgali** [*]

Abstract

The purpose of this paper was to conduct an analysis of Kernel Ridge Regression. We generated the test and training data according to the Gaussian distribution. After splitting the generated data we trained and evaluated it on our model. Then, we tuned the parameters m (number of samples) and d (number of features) on which the error bound of ridge regression depends and compared the experimental results versus theoretical predictions. The results showed that experimental difference between Test Error and Train error most of the time do not exceed the Theoretical Bound. By multiplying or dividing this experimental difference with appropriate variables($\sqrt(m), d^2$), we received approximately constant functions. Finally, we evaluated our learning algorithm on the real data the same way as we did on the generated one. In this case, the results were the same, however with some errors.

1. Introduction

Ridge Regression is a method of regression analysis used to analyze any data that suffer from multicollinearity using a regularization technique called "L2 regularization". It adds a penalty to the regression coefficients for large values, which helps reduce the model's complexity and prevent overfitting. The ridge regression technique is used to fit a model by minimizing the least squares error of the model while penalizing significant coefficients.
Kernel Ridge Regression (KRR) combines Ridge regression with the kernel trick. The **kernel trick** is a mathematical procedure that allows an algorithm to operate in a higher-dimensional space without explicitly computing the coordinates of the data points in the higher-dimensional space. This technique is used to solve nonlinear problems using linear algorithms and is widely used in a variety of machine learning algorithms, especially kernel-based methods such as support vector machines (SVMs).

Algorithm 1 KRR

```
Input: $X \in R^{m*l}$,$y \in R^m$, size $m$, kernel = 'linear', $\lambda = 1$
    $\mathbf{\Phi} = \begin{bmatrix} -\phi(x_1)^T- \\ -\phi(x_2)^T- \\ . \\ . \\ -\phi(x_m)^T- \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ y_m \end{bmatrix}$    (1)
Output: $w = (\mathbf{\Phi^T\Phi} + \lambda\mathbf{I})^{-1}\mathbf{\Phi^T y}$
```

2. Kernel Ridge Regression

2.1. Algorithm

In ridge regression, we have $y \in \mathbb{R}^m$ and a matrix $X \in \mathbb{R}^{\mathbb{R}\times l}$, where $\mathbf{m}$ stands for a number of training points and l is the dimension of the raw data points. Because of the Theorem 3.1 structure, we need to augment features to the data points and replace X with $\Phi \in \mathbb{R}^{n*d}$, where $\phi_i^T = \phi(x_i) \in \mathbb{R}^d$(Nasiriany et al., 2019). However, as we use **KRR** with a 'linear' kernel parameter, dimensions will stay the same: $n = m; d = l$.In the end, we solve a well-defined optimization problem that involves Φ and y, over the parameters $\mathbf{w} \in \mathbb{R}^d$.

2.2. Error Analysis

L is the squared loss, this represents the *mean squared error* of h:

$$L(h(x), y) = |y - h(x)|^2 \tag{2}$$

Expected loss or generalization error R:

$$R(h) = \mathbb{E}_{(x,y)\sim D}[L(h(x), y)] \tag{3}$$

The empirical loss or error of $h \in \mathscr{H}$ is denoted by $\hat{R}_S(h)$ and defined by:

$$\hat{R}_S(h) = \frac{1}{m}\sum_{i=1}^{m}[L(h(x_i), y_i)] \tag{4}$$

2.3. Data

The training and test data were randomly generated according to the Gaussian distribution which corresponds to the

Ridge Regression model. To check if the experimental test error decays as $\frac{1}{\sqrt{m}}$, the number of samples (m) in the data set varied from 100 to 10000, while the number of features (d) was fixed at 15. And to check if the experimental test error grows as d^2, the number of features (d) in the data set varied from 10 to 100, while the number of samples (m) was fixed at 2000.
To evaluate our learning algorithm on some real data. We have performed a similar procedure for the California housing dataset as for the generated data. The California housing dataset was chosen since it contains more than 20,000 samples and 8 features that consist of real numbers, making it a suitable dataset to evaluate our model.

3. Relevant Work

According to Theorem 11.11 (Mohri et al., 2018), the error bounds for kernel ridge regression are as follows

Theorem 3.1 *Let $K : X \times X \to \mathbb{R}$ be a PDS kernel, $\Phi : X \to \mathbb{H}$ a feature mapping associated to K, and $\mathscr{H} = \{x \mapsto \mathbf{w} \cdot \mathbf{\Phi}(\mathbf{x}) : \|\mathbf{w}\|_{\mathbb{H}} \leq \mathbf{\Lambda}\}$. Assume that there exists $r > 0$ such that $K(x,x) \leq r^2$ and $M > 0$ such that $|h(x) - y| < M$ for all $(x,y) \in X \times Y$. Then, for any $\delta > 0$, with probability at least $1-\delta$, each of the following inequalities holds for all $h \in \mathscr{H}$:*

$$R(h) \leq \hat{R}_s(h) + 4M\sqrt{\frac{r^2\Lambda^2}{m}} + M^2\sqrt{\frac{log\frac{1}{\delta}}{2m}} \qquad (5)$$

$$R(h) \leq \hat{R}_s(h) + \frac{4M\mathbf{\Lambda}\sqrt{\mathbf{Tr}[\mathbf{K}]}}{m} + 3M^2\sqrt{\frac{log\frac{2}{\delta}}{2m}} \qquad (6)$$

Let's compare the computational complexities of kernelized and non-kernelized ridge regression. Suppose that $z \in R^{\ell}$, and we would like to compute its predicted value $\hat{y}$. Let's observe how these values are calculated in each case:
1. Kernelized

$$\hat{\mathbf{y}} = \langle \phi(\mathbf{z}), \mathbf{w}^* \rangle = \phi(\mathbf{z})\mathbf{\Phi^T}(\mathbf{\Phi}\mathbf{\Phi^T} + \lambda\mathbf{I})^{-\mathbf{1}}\mathbf{y} \qquad (7)$$

Computing the $\mathbf{K}$ term takes $O(n^2)(\ell + logp))$, and inverting the matrix takes $O(n^3)$. These two computations dominate, for a total computation time of $O(n^3 + n^2(\ell + logp))$.
2. Non-kernelized

$$\hat{\mathbf{y}} = \langle \phi(\mathbf{z}), \mathbf{w}^* \rangle = \phi(\mathbf{z})(\mathbf{\Phi^T}\mathbf{\Phi} + \lambda\mathbf{I})^{-\mathbf{1}}\mathbf{\Phi^T}\mathbf{y} \qquad (8)$$

Computing the $\mathbf{\Phi^T\Phi}$ term takes $O(d^2n)$, and inverting the matrix takes $O(d^3)$. These two computations dominate, for a total computation time of $O(d^3 + d^2n)$.
The data we are going to use contains much more samples n than the number of features d. So we are going to use the non-kernelized ridge regression, because if $d \ll n$, the non-kernelized method is preferable.

4. Numerical Experiments

4.1. Generated Data

First of all, we started our experiments with varying the **m** parameter from 100 to 10000 with $step = 100$.At each step we trained **KRR** with parameters mentioned in algorithm section and evaluated the Test Error and Theoretical Bound, where:

- Test Error = $R(h) - \hat{R}_s(h)$
- Theoretical Bound = $4M\sqrt{\frac{r^2\Lambda^2}{m}} + M^2\sqrt{\frac{log\frac{1}{\delta}}{2m}}$
- $r = \Lambda = M = 1$ and $\delta = 0.05$

In Theoretical Bound, we fixed r, Λ, M as 1, due to Gaussian Distribution of Generated Data, and $\delta = 0.05$. In Figure 1, we can see that test error does not exceed the error bound for the whole range. To check that error decays as $1/\sqrt{(m)}$, we will multiply Test error and Theoretical Bound on $\sqrt{(m)}$ to get approximately constant function. In the 2nd graph of the Figure 1, we can see how the Test Error and and Theoretical Bound have the same approximately constant pattern.

Then, we tuned the number of features(d) as the **d** parameter play important role in Theoretical Bound. In the (5)eq. we need to calculate $Tr[\mathbf{K}]$ which requires the computation of $\mathbf{\Phi^T\Phi}$ term. This term takes $O(d^2n)$, where $n = m$ because this paper use Non-kernelized way and m stays fixed(2000). As a result, we get $O(d^2)$ and for this reason we will divide the Test Error and Theoretical bound by d^2 to get approximate constant function. Other Theoretical Bound parameters stay the same as in the first experiment, except m.
In the Figure 2, we plotted the graph, where **d** was tuned from 10 to 100 with step 1 and visualised the $\frac{TestError}{d^2}$. The test error grows quadratically with d^2 and the 1-degree polynomial which was fitted to $\frac{TestError}{d^2}$, shows a graph of approximate constant function.

4.2. Real Data

To evaluate our learning algorithm on some real data. We have performed a similar procedure for the California housing dataset as for the generated data. The California housing dataset was chosen since it contains more than 20,000 samples and 8 features that consist of real numbers, making it a suitable dataset to evaluate our model.

By conducting the same experiments on California housing dataset, we received almost the same patterns of Test Error. In Figure 3, we varied the **m** parameter and visualised the Test Error, Theoretical Bound and 2-degree polynomial to

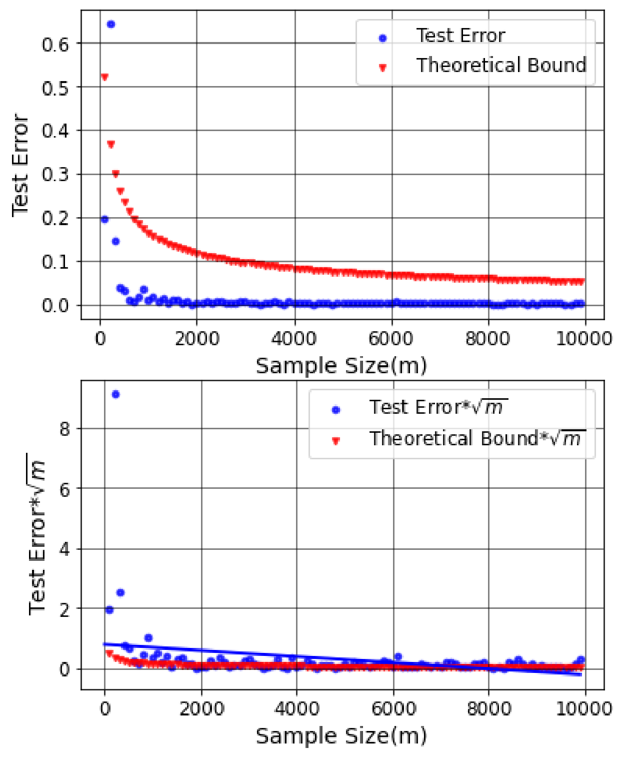

Figure 1. 1st graph visualise the KRR Test Error and Theoretical bound trends for Sample Size range. While 2nd graph visualise KRR Test Error and Theoretical Error bound multiplied by $\sqrt{m}$ trend for Sample Size range.

show the Test Error trend. The 2d polynomial do not exceeds the Theoretical bound and this behavior is described by Theorem 3.1.Although, there are some points which exceeds the Theoretical Bound and we tend to think that the reason for this is lack of data normalization.

Unfortunately, the California housing dataset's number of features is limited to 8 and by varying the **d** parameter we get a poor scatter plot. In Figure 4, we visualised the $\frac{TestError}{d^2}$ and 1-degree polynomial to show the $\frac{TestError}{d^2}$ trend. The reason for dividing the Test Error by d^2 is the same as in the experiment with randomized data. We can claim that test error grows quadratically with d^2 and the 1-degree polynomial which was fitted to $\frac{TestError}{d^2}$, shows a graph of an approximate constant function.

5. Conclusion

The goal of this paper was to verify the error bound of Kernel Ridge regression. We trained and evaluated our learning algorithm by varying the parameters m and d on which the error bound of our model depends for randomly generated data and for California housing data. The results

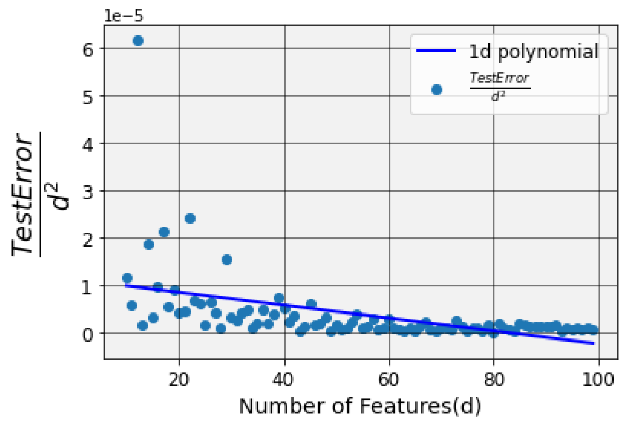

Figure 2. This graph visualise KRR Test Error divided by d^2 and 1-degree fitted polynomial for Number of Features range(d).

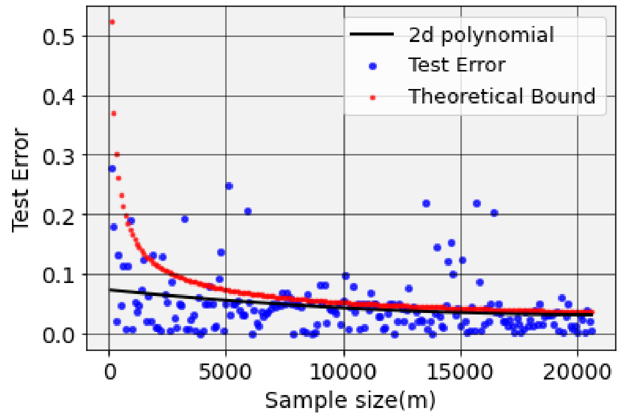

Figure 3. This graph visualise the KRR Test Error, Theoretical bound and 2-degree polynomial for the California housing dataset size range.

of the experiment showed that the difference between Test Error and Train Error does not exceed the Theoretical Bound with some exceptions. When we multiplied or divided the experimental difference by appropriate variables $(\sqrt(m), d^2)$, the results were approximately constant functions. We tested our learning algorithm on real data, and the results were similar, albeit with some deviations.

To see the the gihub repository for this Research paper, here is the link

6. Contributions

The overall workload was divided equally. In particular:

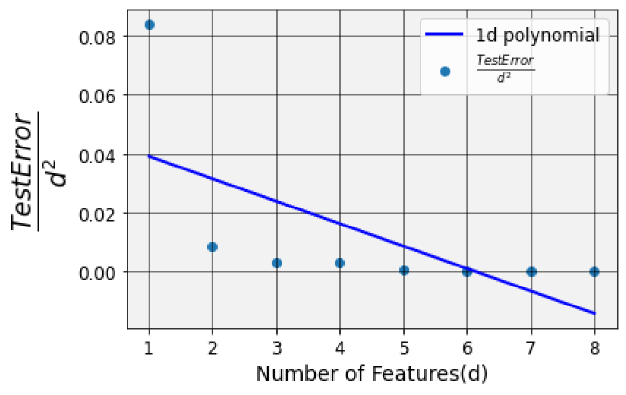

Figure 4. This graph visualises KRR Test Error divided by d^2 and 1-degree fitted polynomial for California housing dataset's Number of Features range(d).

- Adilkhan Bakridenov: Random Data Generation, Conduction of Experiments on the generated Data, Theorem Elaboration, writing Research paper and Readme file.
- Nurmadi Nurgali: Conduction of Experiments on the California housing dataset, Theorem Elaboration, writing Research paper and Readme file.

7. Acknowledgements

We would like to thank professor Zhenisbek Assylbekov for guiding us throughout whole MATH 490: Statistical Learning course. Your patience and encouragement have been invaluable in helping us stay on track and ensuring that we produce a quality project. We are sincerely grateful for your support and guidance.

References

[1] Soroush Nasiriany, Garrett Thomas, William Wang, Alex Yang, Jennifer Listgarten, Anant Sahai. *A Comprehensive Guide to Machine Learning*. The University of California, Berkeley, 2019

[2] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning*. MIT press, 2018.