# Kernels

Marko Medved

## I. Part 1

In this section we implemented the kernelized ridge regression and support vector regression. In addition to the Linear kernel, we implemented the Polynomial kernel and the RBF kernel.

### A. Kernelized Ridge Regression Implementation

This implementation is simple because the fitting algorithm has a closed-form solution:

$$\boldsymbol{\alpha} = (K_{\text{train}} + \lambda\mathbf{I})^{-1}\mathbf{y}$$

where $K_{\text{train}}$ is the kernel matrix computed between training points, and $\lambda$ is the regularization constant.

Then we can predict with:

$$\hat{\mathbf{y}} = K_{\text{test, train}}\boldsymbol{\alpha}$$

where $K_{\text{test, train}}$ is the kernel matrix computed between the test inputs and the training inputs.

### B. Support vector regression implementation

Here we needed to adapt the Equation 10 from (Smola and Scholkopf, 2004) to a form which fits the requirements of the *cvxopt.solvers.qp*, that is:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\mathbf{x}^T P\mathbf{x} + \mathbf{q}^T\mathbf{x} \\ \text{subject to} \quad & G\mathbf{x} \leq \mathbf{h} \\ & A\mathbf{x} = \mathbf{b} \end{aligned}$$

where:

$$x = [\alpha_1, \alpha_1^*, \alpha_2, \alpha_2^*, \ldots]$$

The matrices were calculated as:

$$\begin{aligned} P_{2i,\,2j} &= \phantom{-}k_{ij}, \\ P_{2i,\,2j+1} &= -k_{ij}, \\ P_{2i+1,\,2j} &= -k_{ij}, \\ P_{2i+1,\,2j+1} &= \phantom{-}k_{ij}. \end{aligned} \qquad q = \begin{bmatrix} \varepsilon - y_1 \\ \varepsilon + y_1 \\ \vdots \\ \varepsilon - y_n \\ \varepsilon + y_n \end{bmatrix} \in \mathbb{R}^{2n}$$

$$G = \begin{bmatrix} -I_{2n} \\ I_{2n} \end{bmatrix} \in \mathbb{R}^{(4n)\times(2n)} \quad h = \begin{bmatrix} \mathbf{0}_{2n} \\ C \cdot \mathbf{1}_{2n} \end{bmatrix} \in \mathbb{R}^{4n}$$

$$A = \begin{bmatrix} 1 & -1 & 1 & -1 & \ldots & 1 & -1 \end{bmatrix} \in \mathbb{R}^{1\times 2n} \quad b = 0$$

where $K$ is the kernel on the training data, $n$ is the number of data points, $\epsilon$ is the margin of tolerance and $C$ is the regularization constant defined as $1/\lambda$.

We can then predict as:

$$\hat{y} = K(X, X_{\text{train}}) \cdot (\boldsymbol{\alpha} - \boldsymbol{\alpha}^*) + b$$

where b is the bias.

### C. Fitting both methods to the 1-dimensional sine data

We fit both methods, each with both kernels to the 1-D sine data, which we can see on Figure 1. Note that the support vectors are the data points where: $\alpha_i - \alpha_i^* > 1\times10^{-5}$. In Table I we can see the parameters we used for each model. Note that for the polynomial kernel to work, we needed to scale the data.

| Model | Kernel | Parameters |
|---|---|---|
| SVR | RBF | $\sigma = 1,\ \varepsilon = 0.5,\ \lambda = 0.001$ |
| Ridge Regression | RBF | $\sigma = 1,\ \lambda = 0.01$ |
| SVR | Polynomial | $M = 10,\ \varepsilon = 0.7,\ \lambda = 0.0001$ |
| Ridge Regression | Polynomial | $M = 10,\ \lambda = 0.001$ |

Table I

Parameters for SVR and Ridge Regression.

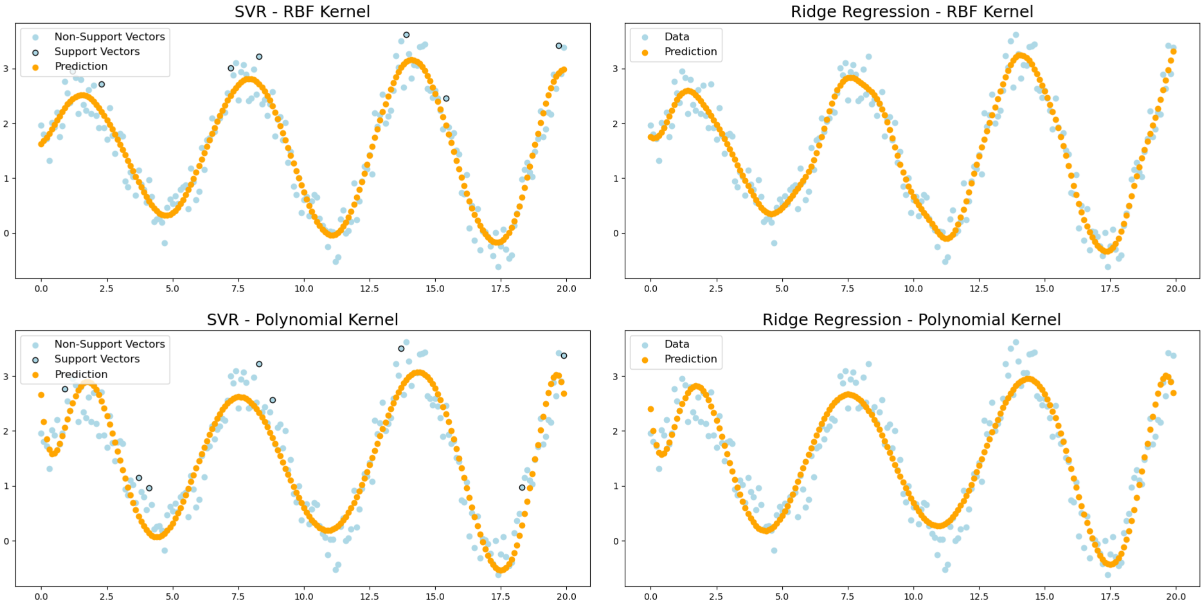

Figure 1. Comparison of SVR and Ridge Regression

## II. Part 2

### A. Results

For this part, we applied both methods and kernels to the *housing2r* dataset. For the polynomial kernel, we evaluated different degrees: $M \in [1, 10]$. For the RBF kernel, we tested a wide range of standard deviations: $\sigma \in \{0.001, 0.01, 0.1, 1, 2, 3, 4, 5, 8, 10, 100\}$. This range includes values that produce smoother decision boundaries (larger $\sigma$) as well as more complex ones (smaller $\sigma$).

To assess predictive performance, we used 10-fold cross-validation. We considered two strategies for selecting the regularization parameter $\lambda$: a fixed value $\lambda = 1$, and a data-driven approach using nested cross-validation (with 6 folds in the inner loop) to tune $\lambda$ for each split.

Uncertainty estimates were computed assuming asymptotic normality of the cross-validation results. Specifically, we calculated the standard deviation across folds and divided by $\sqrt{n}$, where $n$ is the number of folds.

To get a somewhat sparse SVR solution, we set $\epsilon = 1.5$. Figure 2 shows the results. Numbers above and below the curves indicate the number of support vectors for each configuration.

### B. Disscussion

The first comparison we can draw from Figure 2 (Note that the numbers below and above the curves represent the numbers of support vectors) concerns the choice of kernel. The polynomial kernel tends to overfit quickly at higher degrees. For both Ridge Regression and SVR, a degree of 2 gives the best performance. Interestingly, in the case of SVR, degree 10 performs better than degrees 8 or 9.

The RBF kernel is also quite sensitive to the $\sigma$ parameter, especially for Ridge Regression. Very small $\sigma$ values lead to overfitting and high loss, while values in the range of 1 to 10 yield the best results. Larger $\sigma$ values result in underfitting and poor performance.

Overall, all the models with best chosen parameters achieve MSE between 26 and 28, so other factors should influence model choice. Ridge Regression benefits from a closed-form solution and faster training, making it suitable when training time is critical. SVR, on the other hand, requires more training time due to the optimization solver, but inference can be faster and more memory-efficient since it relies only on support vectors.

Both kernels achieve similar performance and yield a comparable number of support vectors at their optimal configurations, so either could be a viable choice depending on practical constraints.

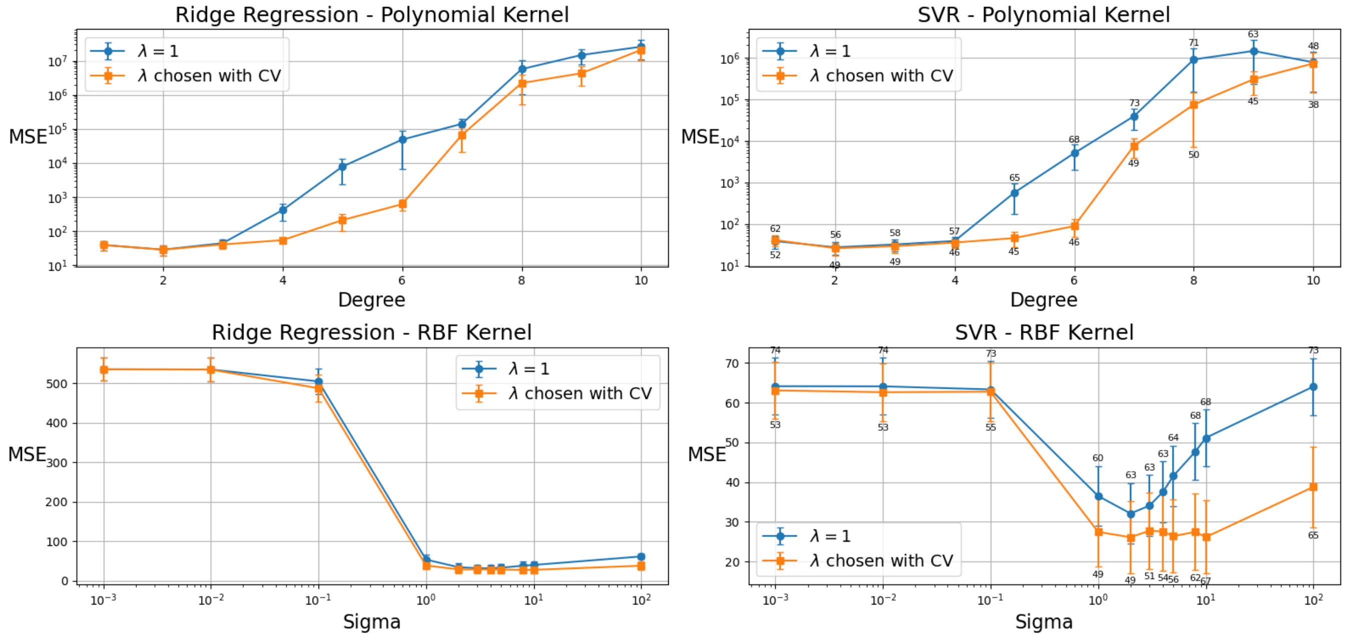

Figure 2. Comparison of models at different parameter values

## III. Part 3