Homework 4 report

Luka Utješinović, 63210495

Kernelized ridge regression

Optimal weights are $\hat{\beta} = \Phi(X)^T \alpha$, where $\alpha = (K + \lambda I_N)^{-1} y$, K is the kernel matrix for the kernel of choice $\mathscr{K}$, Φ is a transformation to the vector space corresponding to $\mathscr{K}$ and N is the number of observations. $\hat{\beta}$ is not stored explicitly, while predictions for new observations are made with $f(x') = \sum_{i=1}^{N} \alpha_i \mathscr{K}(x', x_i)$.

Support vector regression

First we solve the dual problem:

$$\begin{aligned} \text{maximize } & -\frac{1}{2} \sum_{i,j=1}^{N} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \mathscr{K}(x_i, x_j) \\ & - \varepsilon \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{N} y_i (\alpha_i - \alpha_i^*) \\ \text{subject to } & \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) = 0, \alpha_i, \alpha_i^* \in [0, \frac{1}{\lambda}] \end{aligned}$$

New predictions are made with $f(x') = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \mathscr{K}(x', x_i) + \hat{b}$, where the optimal intercept $\hat{b}$ can be computed by averaging over the following quantities for numerical stability: $b_l = \max\{-\varepsilon + y_i - \sum_{j=1}^{N} (\alpha_j - \alpha_j^*) \mathscr{K}(x_j, x_i) \mid \alpha_i < C, \alpha_i^* > 0\}$, $b_u = \min\{\varepsilon + y_i - \sum_{j=1}^{N} (\alpha_j - \alpha_j^*) \mathscr{K}(x_j, x_i) \mid \alpha_i^* < C, \alpha_i > 0\}$

Sine dataset

Throughout this homework, we considered the **polynomial kernel**, $\mathscr{K}(x,y) = (1 + \langle x, y \rangle)^n$, and the **RBF kernel**, $\mathscr{K}(x,y) = \exp(-\frac{1}{2\sigma^2} ||x - y||_2^2)$. Before training, X was standardized. We combined these kernels with previous algorithms on the sine dataset to obtain following results:

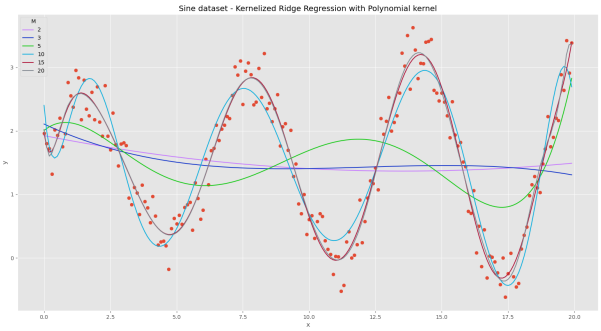

Fig. 1. $\lambda = 0.01$. We can see that kernel degree greatly affects our ability to fit. Since a sine wave is more complex than a simple linear function, we cannot fit the model properly with small degrees, while with higher degrees we can see that there is some overfitting.

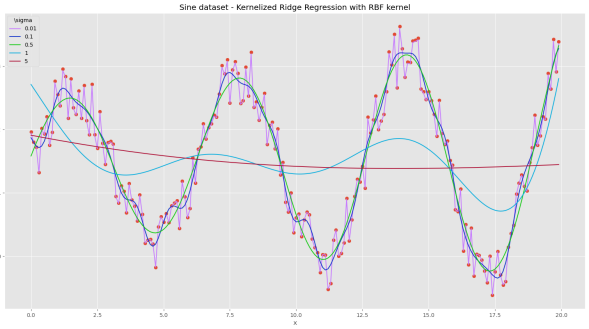

Fig. 2. $\lambda = 0.01$. Small σ values lead to heavy overfitting. By gradually increasing σ we gain in smoothness, until eventually we flatten the entire curve.

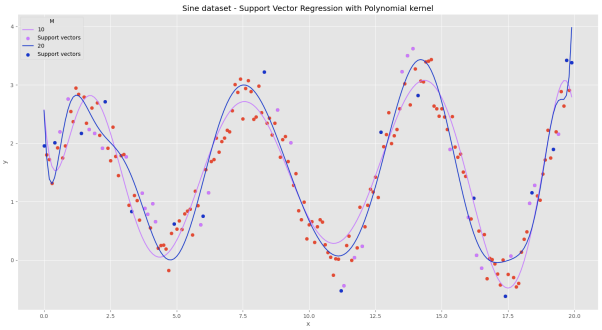

Fig. 3. $\lambda = 0.01$ and $\varepsilon = 0.6$

Fig. 4. $\lambda = 0.01$ and $\varepsilon = 0.6$

Housing dataset

We standardized X for this dataset as well. For hyperparameter tuning, we performed 10-fold cross validation for every model, and we also made sure that same folds were used for every model and hyperparameter combination. For SVR models, we chose $\varepsilon = 10$, which produced a relatively good fit while still getting a sparse solution. Values for σ were

chosen from range [0.1, 10] with increments of 0.1. Following results were obtained:

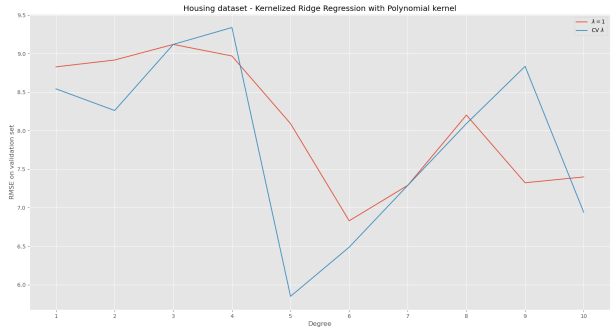

Fig. 5. For lower degrees we can see that RMSE is relatively high. This could be the consequence of having a non-linear relationship between the predictors and the response. For higher degrees we can see that RMSE is still high with some oscillations, and this is probably due to overfitting, as we observed on 1

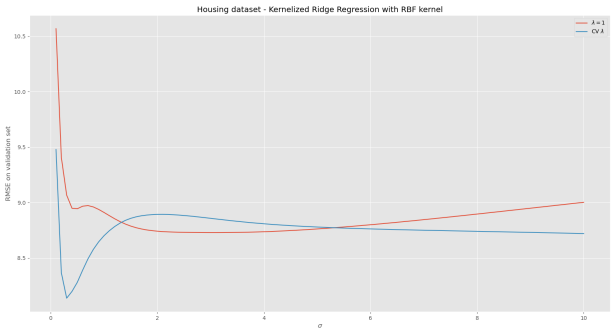

Fig. 6. Starting RMSE is high again, but this time it is due to overfitting, as we have seen from 2

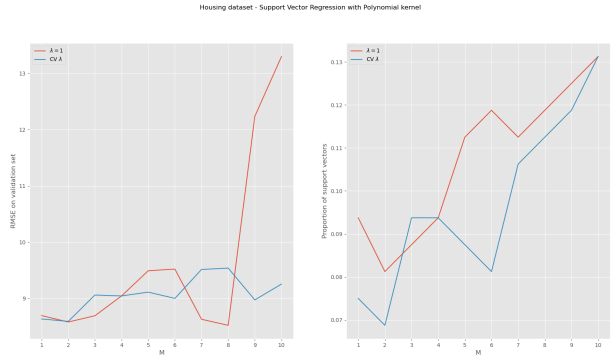

Fig. 7. y-axis of the figure on the right represents proportion of support vectors.

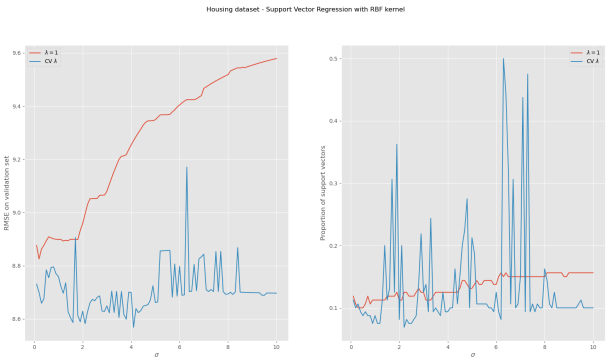

Fig. 8. Caption

Model	λ	M	σ	RMSE	SV Proportion
KRR + Poly	0.01	5	/	**5.85**	/
KRR + RBF	0.01	/	0.3	8.14	/
SVR + Poly	1	8	/	8.63	0.12
SVR + RBF	1	/	0.2	8.82	0.11

Table I

For every model from previous figures we reported the one which achieved best RMSE, along with hyperparameter values that were chosen and proportion of support vectors for SVR models.

Based on these results, we would choose KRR over SVR, following this reasoning:

- Both KRR models obtained better RMSE on the validation set than SVR models.
- Training SVR models requires much more time (we need to solve the dual quadratic program), while training KRR models requires matrix operations (inversion, multiplication) for which there are known efficient algorithms.
- One option would be to tune the ε hyperparameter of SVR models. However, that would increase the dimension of our grid search space and tuning would become significantly harder because of training time.