Kernels

Biljana Vitanova
MLDS1 2024/25 , FRI, UL
bv7063@student.uni-lj.si

I. Introduction

In this project, we implement two models: Ridge Regression and ε-Support Vector Regression, both with support for kernel functions. We study their parameters, fit the models on a housing dataset, and analyze their performance. Additionally, we implement a temporal RBF kernel, apply it to time series data, and compare its performance with the standard RBF kernel.

II. Model Implementation

A. Methodology

In the first part, we implement two models: Kernelized Ridge Regression (KRR) and ε-Support Vector Regression (ε-SVR), using two kernel functions: polynomial kernel, and RBF kernel.

For Ridge Regression, the prediction function is $\hat{y} = Xw$, where w is the weight vector. In the dual representation, the weight vector is expressed as a linear combination of the training points: $w = X^\top \alpha$. To solve for α, we define the loss function, take the derivative with respect to α, and get the closed-form solution $\alpha = (K + \lambda I)^{-1} y$ where K is the kernel or Gram matrix computed as $K = XX^\top$ in the linear case, or as $K_{ij} = k(x_i, x_j)$ using a positive definite kernel. Once the model is fitted, predictions for a new input x' are made using $\hat{y}(x') = k(x', X)^\top \alpha$ where $k(x', X)$ is the vector of kernel evaluations between x' and all training samples.

For ε-Support Vector Regression (ε-SVR), the dual problem introduces coefficients α_i and α_i^*, which correspond to upper and lower margin violations, respectively. The solution involves computing a kernel matrix K and solving a constrained quadratic optimization problem with $0 \leq \alpha_i, \alpha_i^* \leq C$ and the equality constraint $\sum_i (\alpha_i - \alpha_i^*) = 0$. Once solved, the prediction function is given by

$$\hat{y}(x') = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\, k(x_i, x') + b.$$

B. Results and Discussion

Next, we apply the implemented models to a sine wave data. For the polynomial kernel, we set the degree to $M = 9$. Increasing the degree above 9 did not improve the fit, while using $M < 9$ resulted in underfitting. For the RBF kernel, we set the parameter $\sigma = 5$, which controls the similarity between points. Increasing σ further caused the model to underfit the data, while decreasing it led to overfitting, capturing noise and losing the clear sine pattern.

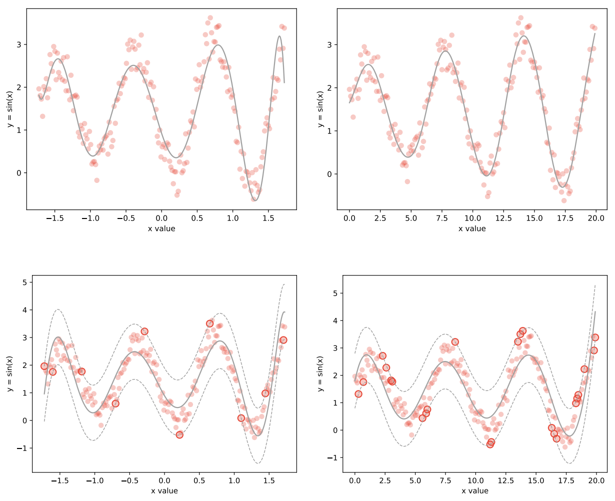

Fig. 1: Comparison of kernelized models. Top row: KRR with polynomial (left) and RBF (right) kernels. Bottom row: SVR with polynomial (left) and RBF (right) kernels.

For the regularization parameter λ, we set $\lambda = 0.001$ for both SVR and KRR. To produce a sparse solution that still fits the data well, we set $\varepsilon = 1$ in SVR. This resulted in approximately 5% of the data points being support vectors when using the polynomial kernel, and 12% with the RBF kernel. Decreasing ε led to an increased number of support vectors, while increasing it further caused the model to underfit.

III. Application of kernels functions

A. Results and Discussion

For tuning the regularization parameter, we experimented with values between 0.0001 and 1000. For SVR, we set $\varepsilon = 10.0$, which provided a good balance between MSE and the number of support vectors — approximately 10% of the data. Decreasing ε to 5 led to around 50% of the data becoming support vectors with similar performance, while reducing it to 1 resulted in nearly all data points being support vectors.

With the RBF kernel, we experimented with different values for the σ parameter in the range [0.1, 10.0]. For the KRR model, we observed that increasing σ improved performance, and a similar trend was seen for the SVR model.

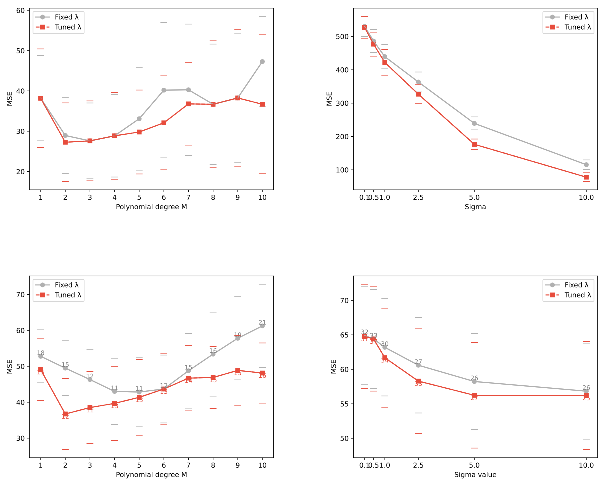

Fig. 2: Comparison of the models on housing dataset. Top row: KRR with polynomial (left) and RBF (right) kernels. Bottom row: SVR with polynomial (left) and RBF (right) kernels.

For both models, polynomial kernels provided better results and lower MSE. The optimal regularization parameter varied with the polynomial degree, but in most cases, the best results were obtained with lower regularization values between 10 and 0.01.

Comparing KRR and SVR, KRR converged faster than SVR and achieved lower MSE. In our opinion, using KRR with a polynomial kernel gave the best overall results.

IV. Temporal data custom kernel

A. Methodology

To account for the structure in time series, we try to implement a temporal kernel that extends over the standard RBF kernel. The idea behind is that rather than comparing vectors directly, we compare sequences of feature-time pairs. With this we dont discard the time information, which in some application is crucial. The kernel is defined as follows:

$$K((x_a, t_a), (x_b, t_b)) = \exp\left(-\frac{\|x_a - x_b\|^2}{\sigma_f^2} - \frac{(t_a - t_b)^2}{\sigma_t^2}\right)$$

In this definition of the kernel, two parameters play a crucial role: σ_f, and σ_t. σ_f determines how much feature differences affect similarity. With a large σ_f , even sequences that are not very similar can have high similarity scores, and vice versa. The parameter σ_t controls how strongly the kernel emphasizes time alignment. A small σ_t enforces that patterns must occur at nearly the same time points, and the opposite for large σ_t.

B. Results and Discussion

Next, we evaluate the temporal kernel on two datasets: one synthetic, to test whether our assumption holds, and the other a real-life dataset for monthly revenue prediction. The goal is to show that adding temporal information improves the performance of the models.

1) Synthetic Dataset: We generate N synthetic sequences of length w. Each sequence is a noisy sine wave with a randomly placed bump. Each data point is represented as a pair (f, t), where f is the sequence of values and t is a vector indicating the bump position. The goal is to predict the bump location. We use the same hyperparameter values for ϵ, λ, and set both σ_f and σ_t to 1 for the kernels.

Model	**Kernel**	**MSE**
SVR	RBF	0.0841 ± 0.0044
	Temporal RBF	$\mathbf{0.0003 \pm 0.0000}$

TABLE I: Performance comparison of RBF and Temporal RBF kernels on synthetic data.

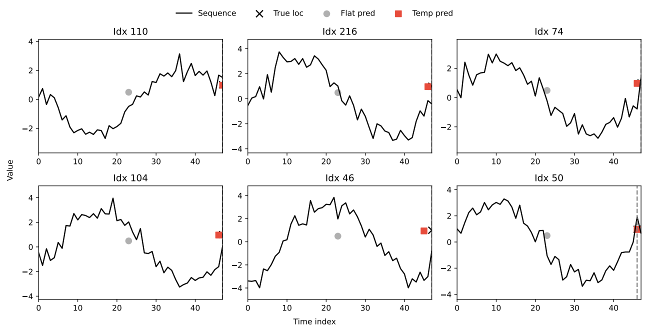

Fig. 3: Bump location prediction using RBF and Temporal RBF kernels on synthetic data.

Table I shows that the Temporal RBF kernel outperforms the standard RBF kernel. Also, Figure 3 illustrates cases where the RBF prediction is significantly off, while the Temporal RBF kernel gives much more accurate results.

2) Revenue Dataset [1]: The dataset contains monthly observations over five years. Each row represents a single month and includes multiple economic indicators such as sales quantity, average cost, and the average annual payroll of the region. The target variable we aim to predict is Revenue.

We used the last 20% of the data as the test set and applied a time window of 10 months. Using the same hyperparameters for both methods, the Temporal RBF kernel outperforms the standard RBF kernel in both KRR and SVR models, as shown in Table II.

Model	kernel	MSE
KRR	RBF	0.6077 ± 0.0872
	Temporal RBF	$\mathbf{0.1045 \pm 0.0318}$
SVR	RBF	0.1675 ± 0.0421
	Temporal RBF	$\mathbf{0.0861 \pm 0.0273}$

TABLE II: Performance comparison of RBF and Temporal RBF kernels on Revenue data.

V. Conclusion

In this project, we successfully implemented two models: KRR and SVR, both with kernel support. We analyzed the model parameters and performed fine-tuning to improve performance on the housing dataset. Additionally, we implemented a temporal RBF kernel and applied it to time-series data, where it performed overall better compared to the standard RBF kernel.

References

[1] Podsyp. Time series starter dataset. https://www.kaggle.com/datasets/podsyp/time-series-starter-dataset, 2020.