Восстановление регрессии с использованием алгоритма Kernel RLS (KRLS)

Казакевич Анна Юрьевна

15 апреля 2025 г.

Содержание

Постановка задачи

Дана обучающая выборка:

$$\{(x_i, y_i)\}_{i=1}^{l}, \quad x_i \in \mathbb{R}, \; y_i \in \mathbb{R}.$$

Необходимо найти функцию $\hat{f}(x)$, минимизирующую квадратичную функцию потерь:

$$L(f) = \sum_{i=1}^{l} \Big(y_i - f(x_i) \Big)^2.$$

При использовании ядерных методов аппроксимация задаётся как:

$$\hat{f}(x) = \sum_{j=1}^{m} \alpha_j \, k(x_j, x),$$

где $k(x, x')$ — ядровая функция, а $\{x_j\}_{j=1}^{m}$ образуют словарь (подмножество обучающих примеров).

Базовая идея метода

- Стандартные ядерные методы используют все обучающие точки, что приводит к высоким вычислительным затратам.
- Для уменьшения сложности применяется метод разрежённой аппроксимации.
- Каждая новая точка x_t отображается в пространство признаков $\phi(x_t)$.
- Проверяется условие *approximate linear dependency (ALD)*:

$$\delta_t = k(x_t, x_t) - \mathbf{k}_{t-1}(x_t)^\top \tilde{K}_{t-1}^{-1} \mathbf{k}_{t-1}(x_t) \leq \nu.$$

- Если $\delta_t > \nu$, точка x_t добавляется в словарь, что обеспечивает разрежённое представление.

Общее описание алгоритма KRLS

- Классический алгоритм RLS минимизирует сумму квадратов ошибок в виде:
$$f(x) = \langle \mathbf{w}, \phi(x) \rangle.$$
- В ядерном варианте решение записывается через словарь:
$$\mathbf{w}_t = \sum_{j \in D_t} \alpha_j \, \phi(x_j) \quad \Rightarrow \quad \hat{f}_t(x) = \sum_{j \in D_t} \alpha_j \, k(x_j, x).$$
- Динамическое расширение словаря осуществляется с помощью ALD-проверки.
- Обновление коэффициентов происходит по двум схемам:
  1. При $\delta_t > \nu$ — добавление новой точки.
  2. При $\delta_t \le \nu$ — обновление без расширения словаря.

Обновление: Добавление новой точки (ALD не выполнено)

Если $\delta_t > \nu$:

- Добавляем x_t в словарь.
- Обновляем инвертированную матрицу ядровых значений:

$$K_{\tilde{t}}^{-1} = \frac{1}{\delta_t} \begin{pmatrix} \delta_t \, K_{\tilde{t-1}}^{-1} + a_t \, a_t^\top & -a_t \\ -a_t^\top & 1 \end{pmatrix},$$

  где $a_t = K_{\tilde{t-1}}^{-1} \, \mathbf{k}_{t-1}(x_t)$.
- Обновление коэффициентов:

$$\tilde{\alpha}_t = \begin{pmatrix} \tilde{\alpha}_{t-1} - a_t \, \dfrac{e_t}{\delta_t} \\ \dfrac{e_t}{\delta_t} \end{pmatrix}, \quad e_t = y_t - \mathbf{k}_{t-1}(x_t)^\top \, \tilde{\alpha}_{t-1}.$$

Обновление: Без расширения словаря (ALD выполнено)

Если $\delta_t \le \nu$:

- Обновляем вспомогательную матрицу P:

$$P_t = P_{t-1} - \frac{P_{t-1}\, a_t\, a_t^\top\, P_{t-1}}{1 + a_t^\top\, P_{t-1}\, a_t}.$$

- Обновляем коэффициенты:

$$\tilde{\alpha}_t = \tilde{\alpha}_{t-1} + K_{\tilde{t-1}}^{-1}\, q_t\, e_t, \quad q_t = P_{t-1}\, a_t.$$

Псевдокод алгоритма KRLS (ALD)

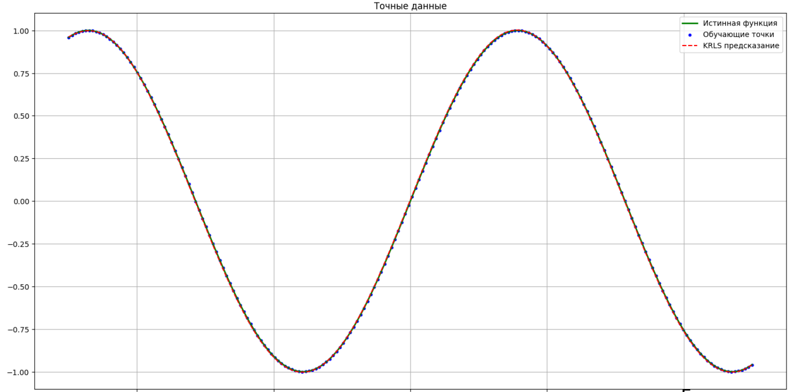
	Cost
Parameter: ν	
Initialize: $\tilde{\mathbf{K}}_1 = [k_{11}]$, $\tilde{\mathbf{K}}_1^{-1} = [1/k_{11}]$,	
$\boldsymbol{\alpha}_1 = (y_1/k_{11})$, $\mathbf{P}_1 = [1]$, $m = 1$	
for $t = 2, 3 \ldots$	
1. Get new sample: $(\mathbf{x}_t, y_t)$	
2. Compute $\tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)$	$O(m)$
3. ALD test:	
$\boldsymbol{a}_t = \tilde{\mathbf{K}}_{t-1}^{-1} \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)$	$O(m^2)$
$\delta_t = k_{tt} - \tilde{\mathbf{k}}_{t-1}(\mathbf{x}_t)^\top \boldsymbol{a}_t$	$O(m)$
if $\delta_t > \nu$ % add $\mathbf{x}_t$ to dictionary	$O(1)$
$\mathcal{D}_t = \{\mathcal{D}_{t-1} \cup \{\tilde{\mathbf{x}}_t\}\}$	$O(1)$
Compute $\tilde{\mathbf{K}}_t^{-1}$ (3.14) with $\tilde{\boldsymbol{a}}_t = \boldsymbol{a}_t$	$O(m^2)$
$\boldsymbol{a}_t = (\mathbf{0}, \ldots, 1)^\top$	$O(m)$
Compute $\mathbf{P}_t$ (3.15)	$O(m)$
Compute $\boldsymbol{\alpha}_t$ (3.16)	$O(m)$
$m := m + 1$	$O(1)$
else % dictionary unchanged	
$\mathcal{D}_t = \mathcal{D}_{t-1}$	
$\mathbf{q}_t = \frac{\mathbf{P}_{t-1}\boldsymbol{a}_t}{1+\boldsymbol{a}_t^\top \mathbf{P}_{t-1}\boldsymbol{a}_t}$	$O(m^2)$
Compute $\mathbf{P}_t$ (3.12)	$O(m^2)$
Compute $\boldsymbol{\alpha}_t$ (3.13)	$O(m^2)$
Output: $\mathcal{D}_t$, $\boldsymbol{\alpha}_t$	

Восстановление регрессии по точным данным

- Генерируется равномерная выборка $x \in [-5, 5]$ с зависимостью:

$$y = \sin(x).$$

- Отсутствует шум в данных.
- Алгоритм KRLS использует число опорных векторов $m \ll 200$.

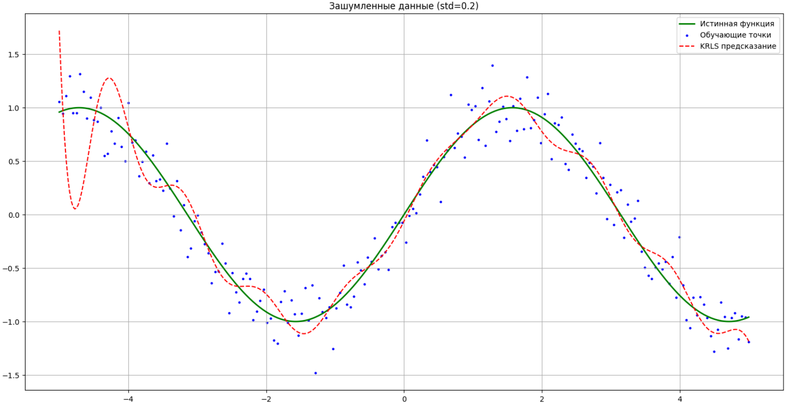


Вывод: Число опорных векторов $= 31$, MSE $\approx 2.78 \times 10^{-5}$.

Восстановление регрессии по зашумлённым данным

- Модельные данные с зависимостью:

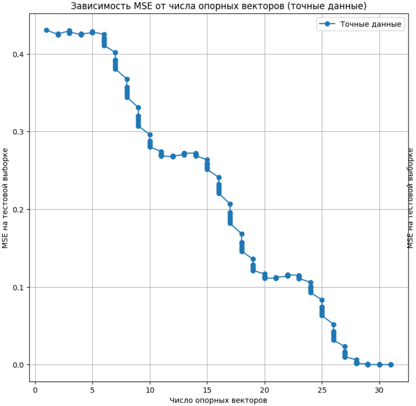
$$y = \sin(x) + \eta, \quad \eta \sim \mathcal{N}(0, 0.2).$$

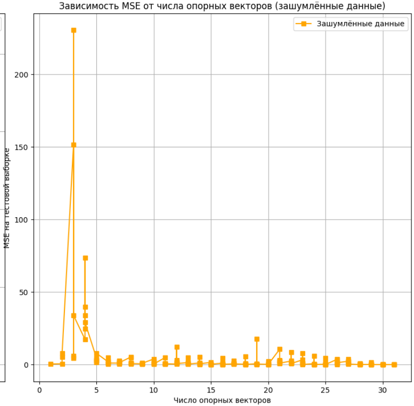


Вывод: Число опорных векторов = 31, MSE ≈ 0.03445.

Зависимость MSE от числа опорных векторов

- При уменьшении порога ν число опорных точек растёт.
- Снижение MSE происходит до определённого уровня, демонстрируя компромисс между размером модели и точностью.





Полный код алгоритма

Полный код алгоритма, а также более подробный отчёт о методе и его работе можно найти по ссылке
https://github.com/AnsKaz-yu/KRLS-SVM-algorithm/tree/main
Там же в отчёте находится составленный мной подробный псевдокод, который из-за объёма не получилось вставить в презентацию.

Выводы

- Алгоритм KRLS расширяет классический метод RLS для задач регрессии в высокоразмерном пространстве признаков.
- Динамическое формирование разрежённого словаря с помощью ALD существенно уменьшает вычислительные затраты.
- Оптимальный выбор порога ν обеспечивает баланс между точностью аппроксимации и эффективностью модели.
- Метод подходит для онлайн-обучения, обновляя модель за фиксированное количество операций.

Литература

1. Engel Y., Mannor S., Meir R. The Kernel Recursive Least Squares Algorithm // Advances in Neural Information Processing Systems (NeurIPS). – 2003.