SVM и RVM

Сергей Николенко

Академический Университет, 2012

Outline

В прошлый раз

- Метод опорных векторов решает задачу классификации.
- Формально: есть точки $x_i \in \mathbb{R}^n$, $i = 1..m$, у точек есть метки $t_i = \pm 1$.
- Мы хотим разделить данные поверхностью так, чтобы максимизировать зазор.

В прошлый раз

- Формальная постановка задачи:

$$\max_{\mathbf{a}} \left\{ \sum_n a_n - \frac{1}{2} \sum_n \sum_m a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m), \right.$$
$$\left. \text{где } \sum_n a_n t_n = 0, \quad a_n \geq 0. \right\}$$

- Это для случая разделимых данных.

В прошлый раз

- Если данные неразделимы, то можно добавить штраф

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2}\|\mathbf{w}\|^2,$$

и задача оптимизации будет выглядеть как

$$\max_{\mathbf{a}} \left\{ \sum_n a_n - \frac{1}{2} \sum_n \sum_m a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m), \right.$$
$$\left. \text{где } \sum_n a_n t_n = 0, \quad 0 \le a_n \le C. \right\}$$

- Целевая функция такая же, параметр C наказывает за точки, лежащие с неправильной стороны.

ν-SVM

- Другой вариант для неразделимых данных – ν-SVM [Schölkopf et al., 2000].
- Максимизируем

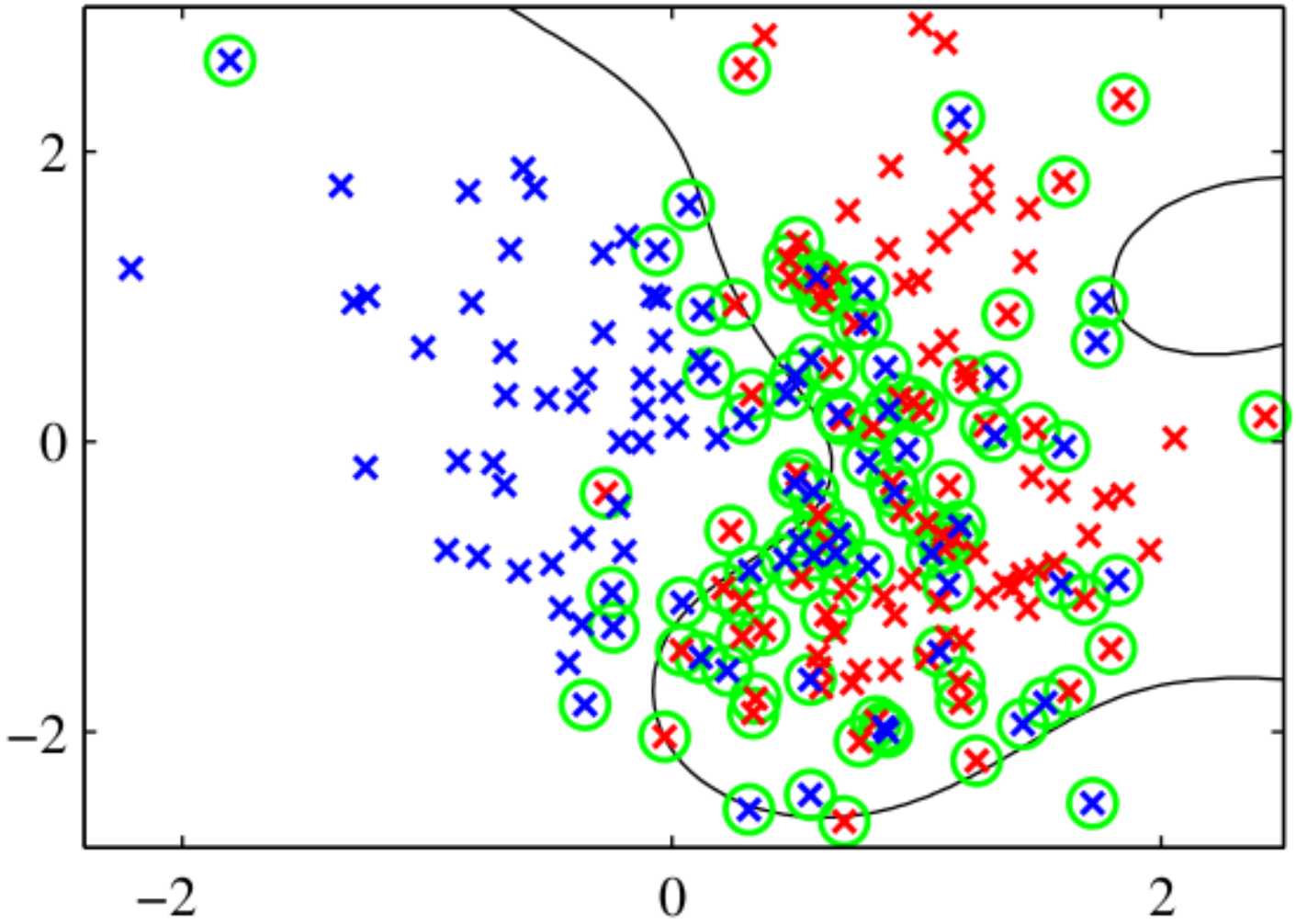
$$L(\mathbf{a}) = -\frac{1}{2}\sum_n\sum_m a_n a_m t_n t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

с ограничениями

$$0 \leq a_n \leq \frac{1}{N},\ \sum_n a_n t_n = 0,\ \sum_n a_n \geq \nu.$$

- Параметр ν можно интерпретировать как верхнюю границу на долю ошибок.

SVM для классификации



Связь с логистической регрессией

- В случае SVM с возможными ошибками мы минимизируем

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2}\|\mathbf{w}\|^2.$$

- Для точек с правильной стороны $\xi_n = 0$, с неправильной – $\xi_n = 1 - y_n t_n$.
- Так что можно записать *hinge error function* $E_{SV}(y_n t_n) = [1 - y_n t_n]_+$ и переписать как задачу с регуляризацией

$$\sum_{n=1}^{N} E_{SV}(y_n t_n) + \lambda\|\mathbf{w}\|^2.$$

Связь с логистической регрессией

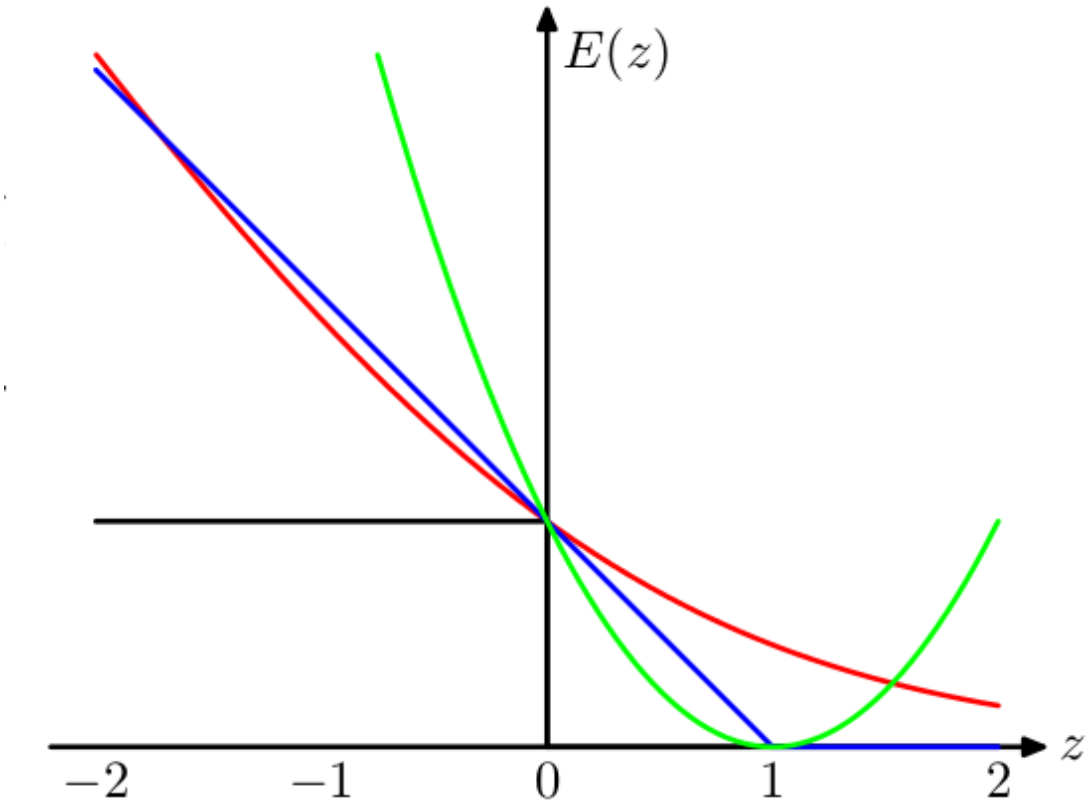
- Вспомним логистическую регрессию и переформулируем её для целевой переменной $t \in \{-1, 1\}$: $p(t = 1 \mid y) = \sigma(y)$, значит, $p(t = -1 \mid y) = 1 - \sigma(y) = \sigma(-y)$, и $p(t \mid y) = \sigma(yt)$.
- И логистическая регрессия – это минимизация

$$\sum_{n=1}^{N} E_{LR}(y_n t_n) + \lambda \|\mathbf{w}\|^2,$$

где $E_{LR}(y_n t_n) = \ln\left(1 + e^{-yt}\right)$.

Связь с логистической регрессией

- График hinge error function, вместе с функцией ошибки для логистической регрессии:



SVM с несколькими классами

- Как обобщить SVM на несколько классов? Варианты (без подробностей):
  1. обучить одну против всех и классифицировать $y(\mathbf{x}) = \max_k y_k(\mathbf{x})$ (нехорошо, потому что задача становится несбалансированной, и $y_k(\mathbf{x})$ на самом деле несравнимы);
  2. можно сформулировать единую функцию для всех K SVM одновременно, но обучение становится гораздо медленнее;
  3. можно обучить попарно $K(K-1)/2$ классификаторов, а потом считать голоса – кто победит;
  4. DAGSVM: организуем попарные классификаторы в граф и будем идти по графу, для классификации выбирая очередной вопрос;
  5. есть даже методы, основанные на кодах, исправляющих ошибки.

SVM с одним классом

- SVM также можно использовать с *одним* классом.
- Как и зачем?

SVM с одним классом

- SVM также можно использовать с *одним* классом.
- Как и зачем?
- Можно при помощи SVM очертить границу области высокой плотности.
- Тем самым найдём выбросы данных (outliers).
- Задача будет такая: найти наименьшую поверхность (сферу, например), которая содержит все точки, кроме доли ν.

SVM для регрессии

- SVM можно использовать для регрессии, сохраняя свойство разреженности (т.е. то, что SVM зависит только от опорных векторов).
- В обычной линейной регрессии мы минимизировали

$$\frac{1}{2}\sum_{n=1}^{N}(y_n - t_n)^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2.$$

- В SVM мы сделаем так: если мы попадаем в ϵ-окрестность предсказания, то ошибки, будем считать, совсем нет.

SVM для регрессии

- ϵ-insensitive error function:

$$E_\epsilon(y(\mathbf{x}) - t) = \begin{cases} 0, & |y(\mathbf{x}) - t| < \epsilon, \\ |y(\mathbf{x}) - t| - \epsilon & \text{иначе.} \end{cases}$$

- И задача теперь выглядит как минимизация

$$C \sum_{n=1}^{N} E_\epsilon \left(y(\mathbf{x}_n) - t_n \right) + \frac{\lambda}{2} \|\mathbf{w}\|^2.$$

SVM для регрессии

- Чтобы переформулировать, нужны по две slack переменные, для обеих сторон «трубки»:

$$y(\mathbf{x}_n) - \epsilon \leq t_n \leq y(\mathbf{x}_n) + \epsilon$$

превращается в

$$t_n \leq y(\mathbf{x}_n) + \epsilon + \xi_n,$$
$$t_n \geq y(\mathbf{x}_n) - \epsilon - \hat{\xi}_n,$$

и мы оптимизируем

$$C \sum_{n=1}^{N} E_\epsilon \left(\xi_n + \hat{\xi}_n\right) + \frac{\lambda}{2}\|\mathbf{w}\|^2.$$

SVM для регрессии

- Если же теперь пересчитать дуальную задачу, то получится

$$L(\mathbf{a}, \hat{\mathbf{a}}) = -\frac{1}{2} \sum_n \sum_m (a_n - \hat{a}_n)(a_m - \hat{a}_m) k(\mathbf{x}_n, \mathbf{x}_m) -$$
$$- \epsilon \sum_{n=1}^{n} (a_n + \hat{a}_n) + \sum_{n=1}^{N} (a_n - \hat{a}_n) t_n,$$

и мы её минимизируем по $a_n, \hat{a}_n$ с условиями

$$0 \leq a_n \leq C,$$
$$0 \leq \hat{a}_n \leq C,$$
$$\sum_{n=1}^{N} (a_n - \hat{a}_n) = 0.$$

SVM для регрессии

- Когда решим эту задачу, сможем предсказывать новые значения как

$$y(\mathbf{x}) = \sum_{n=1}^{N} (a_n - \hat{a}_n)\, k(\mathbf{x}, \mathbf{x}_n) + b,$$

где b можно найти как

$$b = t_n - \epsilon - \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n) =$$

$$= t_n - \epsilon - \sum_{m=1}^{N} (a_m - \hat{a}_m)\, k(\mathbf{x}_n, \mathbf{x}_m).$$

SVM для регрессии

- А условия KKT превращаются в

$$\begin{aligned} a_n \left(\epsilon + \xi_n + y(\mathbf{x}_n) - t_n\right) &= 0, \\ \hat{a}_n \left(\epsilon + \hat{\xi}_n - y(\mathbf{x}_n) + t_n\right) &= 0, \\ (C - a_n)\xi_n &= 0, \\ (C - \hat{a}_n)\hat{\xi}_n &= 0. \end{aligned}$$

- Отсюда очевидно, что либо a_n, либо $\hat{a}_n$ всегда равны 0, и хотя бы один из них не равен, только если точка лежит на или за границей «трубки».
- Опять получили решение, зависящее только от «опорных векторов».

SVM для регрессии

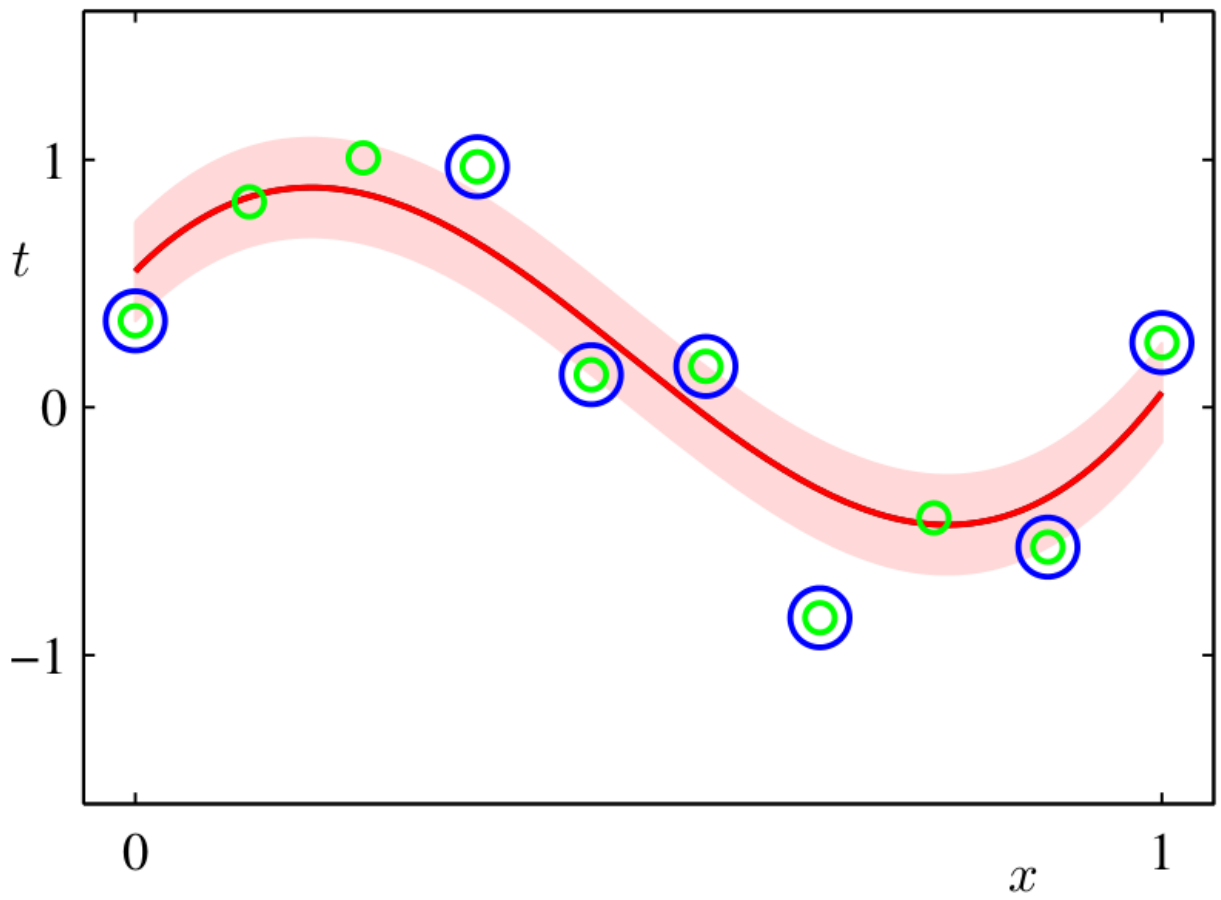
- Но снова можно переформулировать в виде ν-SVM, в котором параметр более интуитивно ясен: вместо ширины трубки ϵ рассмотрим ν – долю точек, лежащих вне трубки; тогда минимизировать надо

$$L(\mathbf{a}) = -\frac{1}{2}\sum_n\sum_m (a_n - \hat{a}_n)(a_m - \hat{a}_m)\, k(\mathbf{x}_n, \mathbf{x}_m) + \sum_{n=1}^{N}(a_n - \hat{a}_n)\, t_n$$

при условиях

$$0 \le a_n \le \frac{C}{N}, \qquad \sum_{n=1}^{N}(a_n - \hat{a}_n) = 0,$$
$$0 \le \hat{a}_n \le \frac{C}{N}, \qquad \sum_{n=1}^{N}(a_n + \hat{a}_n) \le \nu C.$$

SVM для регрессии



Outline

Постановка задачи

- SVM – отличный метод. Но и у него есть недостатки.
  1. Выходы SVM – решения, а апостериорные вероятности непонятно как получить.
  2. SVM – для двух классов, обобщить на несколько проблематично.
  3. Есть параметр C (или ν, или ещё вдобавок ϵ), который надо подбирать.
  4. Предсказания – линейные комбинации ядер, которым необходимо быть положительно определёнными и которые центрированы на точках из датасета.
- Сейчас мы рассмотрим байесовский аналог SVM – *relevance vector machines* (RVM).

RVM для регрессии

- RVM удобнее сразу формулировать для регрессии.
- Вспомним обычную нашу линейную модель:

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t \mid y(\mathbf{x}), \beta^{-1}), \text{ где}$$

$$y(\mathbf{x}) = \sum_{i=1}^{M} w_i \phi_i(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}).$$

RVM для регрессии

- RVM – это вариант такой модели, который старается работать как SVM.
- Рассмотрим

$$y(\mathbf{x}) = \sum_{n=1}^{N} w_n k(\mathbf{x}, \mathbf{x}_n) + b.$$

- Т.е. мы сразу ищем решение в форме линейной комбинации значений ядра (вспомним «эквивалентное ядро» для линейной регрессии), но, в отличие от SVM, теперь ядро никак не ограничивается.

RVM для регрессии

- Для N наблюдений вектора $\mathbf{x}$ (обозначим через $\mathbf{X}$) со значениями $\mathbf{t}$ получим правдоподобие

$$p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n \mid \mathbf{x}_n, \mathbf{w}, \beta^{-1}).$$

- Априорное распределение тоже будет нормальное, но вместо единого гиперпараметра для всех весов мы введём отдельный гиперпараметр для каждого:

$$p(\mathbf{w} \mid \boldsymbol{\alpha}) = \prod_{i=1}^{M} \mathcal{N}(w_i \mid 0, \alpha_i^{-1}).$$

RVM для регрессии

- Отдельные гиперпараметры:

$$p(\mathbf{w} \mid \boldsymbol{\alpha}) = \prod_{i=1}^{M} \mathcal{N}(w_i \mid 0, \alpha_i^{-1}).$$

- Идея здесь в том, что при максимизации апостериорной вероятности большая часть α_i просто уйдёт на бесконечность, и соответствующие веса будут нулевыми.
- Сейчас увидим, как это получается.

RVM для регрессии

- Апостериорное распределение нам знакомо:

$$p(\mathbf{w} \mid \mathbf{t}, \mathbf{X}, \boldsymbol{\alpha}, \beta) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}, \boldsymbol{\Sigma}), \text{ где}$$

$$\mathbf{m} = \beta \boldsymbol{\Sigma} \boldsymbol{\Phi}^\top \mathbf{t},$$

$$\boldsymbol{\Sigma} = \left(\mathbf{A} + \beta \boldsymbol{\Phi}^\top \boldsymbol{\Phi} \right)^{-1},$$

где $\mathbf{A} = \text{diag}(\alpha_1, \ldots, \alpha_M)$, а $\boldsymbol{\Phi}$ в нашем случае – это $\mathbf{K}$, симметрическая матрица с элементами $k(\mathbf{x}_n, \mathbf{x}_m)$.

RVM для регрессии

- Как найти $\boldsymbol{\alpha}$ и β? Нужно максимизировать маргинальное правдоподобие датасета

$$p(\mathbf{t} \mid \mathbf{X}, \boldsymbol{\alpha}, \beta) = \int p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}, \beta) p(\mathbf{w} \mid \boldsymbol{\alpha}) d\mathbf{w}.$$

- Это свёртка двух гауссианов, тоже гауссиан:

$$\ln p(\mathbf{t} \mid \mathbf{X}, \boldsymbol{\alpha}, \beta) = \ln \mathcal{N}(\mathbf{t} \mid \mathbf{0}, \mathbf{C}) =$$

$$= -\frac{1}{2}\left[N \ln(2\pi) + \ln|\mathbf{C}| + \mathbf{t}^\top \mathbf{C}^{-1} \mathbf{t}\right], \text{ где } \mathbf{C} = \beta^{-1}\mathbf{I} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^\top.$$

- Как это оптимизировать?

RVM для регрессии

- Можно подсчитать производные и получить

$$\alpha_i = \frac{\gamma_i}{m_i^2},$$

$$\beta^{-1} = \frac{\|\mathbf{t} - \mathbf{\Phi m}\|^2}{N - \sum_i \gamma_i},$$

где $\gamma_i = 1 - \alpha_i \Sigma_{ii}$.

- Теперь можно просто итеративно пересчитывать $\boldsymbol{\alpha}, \beta$ из $\mathbf{m}, \mathbf{\Sigma}$, потом наоборот, потом опять наоборот, и до сходимости.

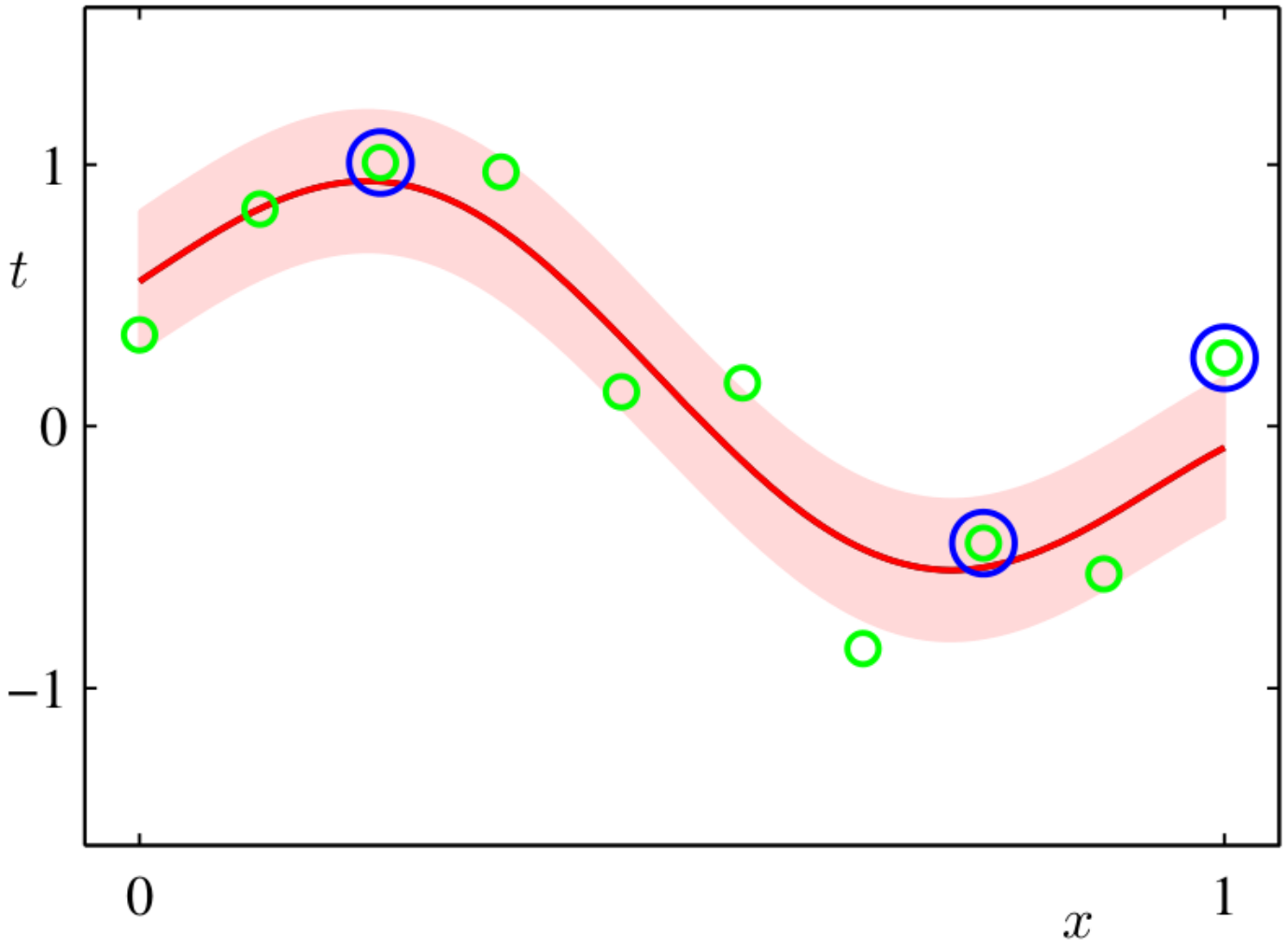
RVM для регрессии

- В результате получается обычно, что большинство α_i неограниченно растут, и соответствующие веса можно считать нулевыми.
- Оставшиеся называются *relevance vectors*, их обычно мало.
- Если теперь мы найдём $\boldsymbol{\alpha}^*, \beta^*$, то предсказывать в новых точках можно как

$$p(t \mid \mathbf{x}, \mathbf{X}, \mathbf{t}, \boldsymbol{\alpha}^*, \beta^*) = \int p(t \mid \mathbf{x}, \mathbf{w}, \beta^*) p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}, \boldsymbol{\alpha}^*, \beta^*) d\mathbf{w} = \\ = \mathcal{N}(t \mid \mathbf{m}^\top \boldsymbol{\phi}(\mathbf{x}), \sigma^2(\mathbf{x})),$$

где $\sigma^2(\mathbf{x}) = (\beta^*)^{-1} + \boldsymbol{\phi}(\mathbf{x})^\top \boldsymbol{\Sigma} \boldsymbol{\phi}(\mathbf{x})$.

RVM для регрессии



RVM для классификации

- Можно сделать то же самое и для классификации. Рассмотрим классификацию с двумя классами, $t \in \{0, 1\}$:

$$y(\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x})).$$

- И добавим сюда, опять же, априорное распределение с разными α_i для каждого веса:

$$p(\mathbf{w} \mid \boldsymbol{\alpha}) = \prod_{i=1}^{M} \mathcal{N}(w_i \mid 0, \alpha_i^{-1}).$$

- Идея: инициализируем $\boldsymbol{\alpha}$, считаем лапласовское приближение к апостериорному распределению, максимизируем, получаем новое $\boldsymbol{\alpha}$, и т.д.

RVM для классификации

- Апостериорное распределение:

$$\ln p(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}) = \ln \left(p(\mathbf{t} \mid \mathbf{w}) p(\mathbf{w} \mid \boldsymbol{\alpha}) \right) - \ln p(\mathbf{t} \mid \boldsymbol{\alpha}) =$$

$$= \sum_{n=1}^{N} \left[t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \right] - \frac{1}{2} \mathbf{w}^\top \mathbf{A} \mathbf{w} + \text{const}.$$

- Мы уже обсуждали, как его максимизировать – IRLS; для этого подсчитаем

$$\nabla \ln p(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}) = \boldsymbol{\Phi}^\top (\mathbf{t} - \mathbf{y}) - \mathbf{A} \mathbf{w},$$

$$\nabla \nabla \ln p(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha}) = - \left(\boldsymbol{\Phi}^\top \mathbf{B} \boldsymbol{\Phi} + \mathbf{A} \right),$$

где $\mathbf{B}$ – диагональная матрица с элементами $b_n = y_n(1 - y_n)$.

RVM для классификации

- Лапласовское приближение получится из $\nabla \ln p(\mathbf{w} \mid \mathbf{t}, \boldsymbol{\alpha})$, и получится

$$\mathbf{w}^* = \mathbf{A}^{-1}\boldsymbol{\Phi}^\top (\mathbf{t} - \mathbf{y}),$$
$$\boldsymbol{\Sigma} = \left(\boldsymbol{\Phi}^\top \mathbf{B}\boldsymbol{\Phi} + \mathbf{A}\right)^{-1},$$

и распределение для предсказания получится

$$p(\mathbf{t} \mid \boldsymbol{\alpha}) = \int p(\mathbf{t} \mid \mathbf{w}) p(\mathbf{w} \mid \boldsymbol{\alpha}) d\mathbf{w} \approx$$
$$\approx p(\mathbf{t} \mid \mathbf{w}^*) p(\mathbf{w}^* \mid \boldsymbol{\alpha}) (2\pi)^{M/2} |\boldsymbol{\Sigma}|^{1/2}.$$

RVM для классификации

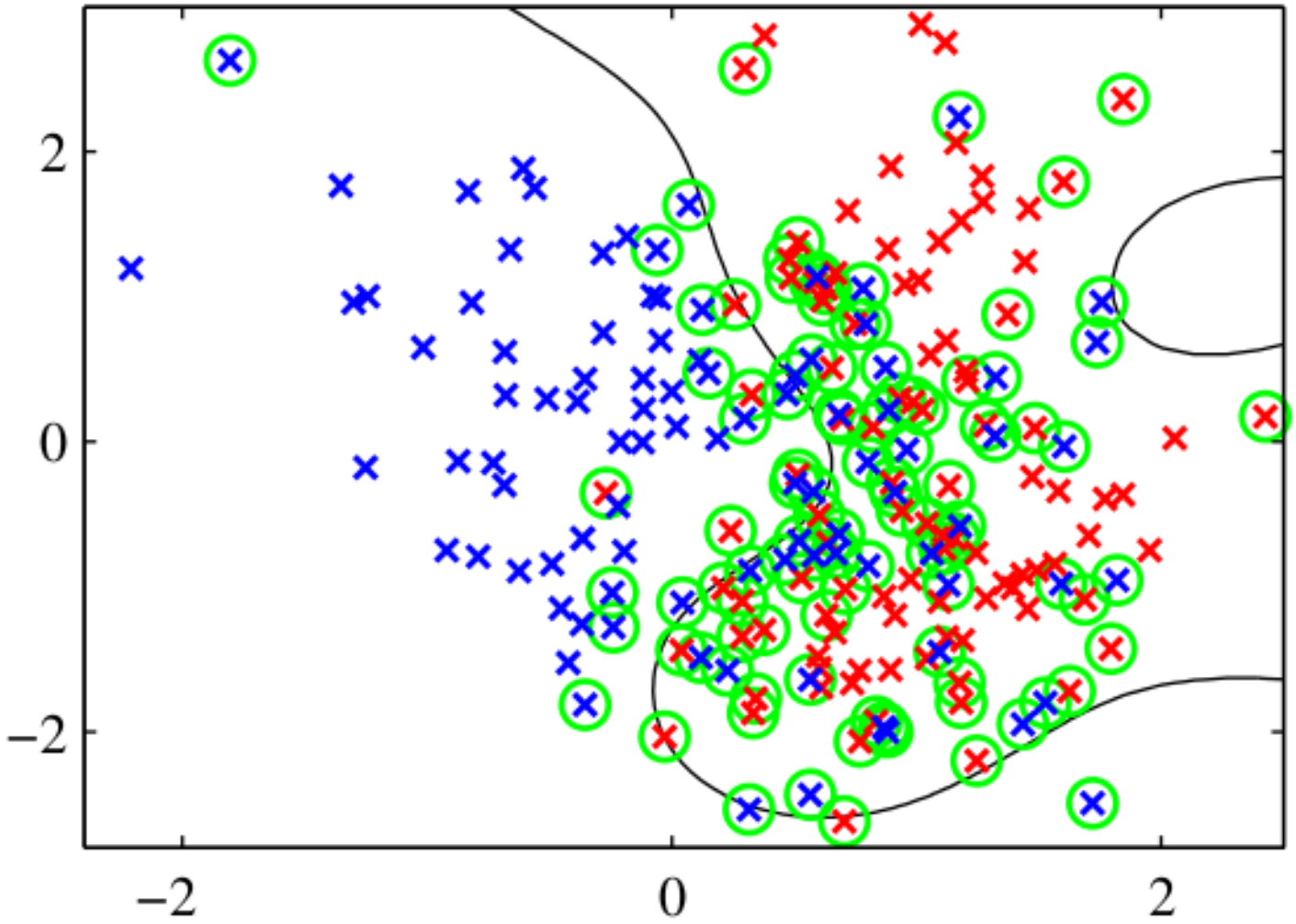
- $p(\mathbf{t} \mid \boldsymbol{\alpha}) = \int p(\mathbf{t} \mid \mathbf{w}) p(\mathbf{w} \mid \boldsymbol{\alpha}) d\mathbf{w} \approx p(\mathbf{t} \mid \mathbf{w}^*) p(\mathbf{w}^* \mid \boldsymbol{\alpha}) (2\pi)^{M/2} |\boldsymbol{\Sigma}|^{1/2}$.
- Теперь мы оптимизируем это по $\boldsymbol{\alpha}$: берём производную, получаем

$$-\frac{1}{2}(w_i^*)^2 + \frac{1}{2\alpha_i} - \frac{1}{2}\Sigma_{ii} = 0, \text{ т.е.}$$

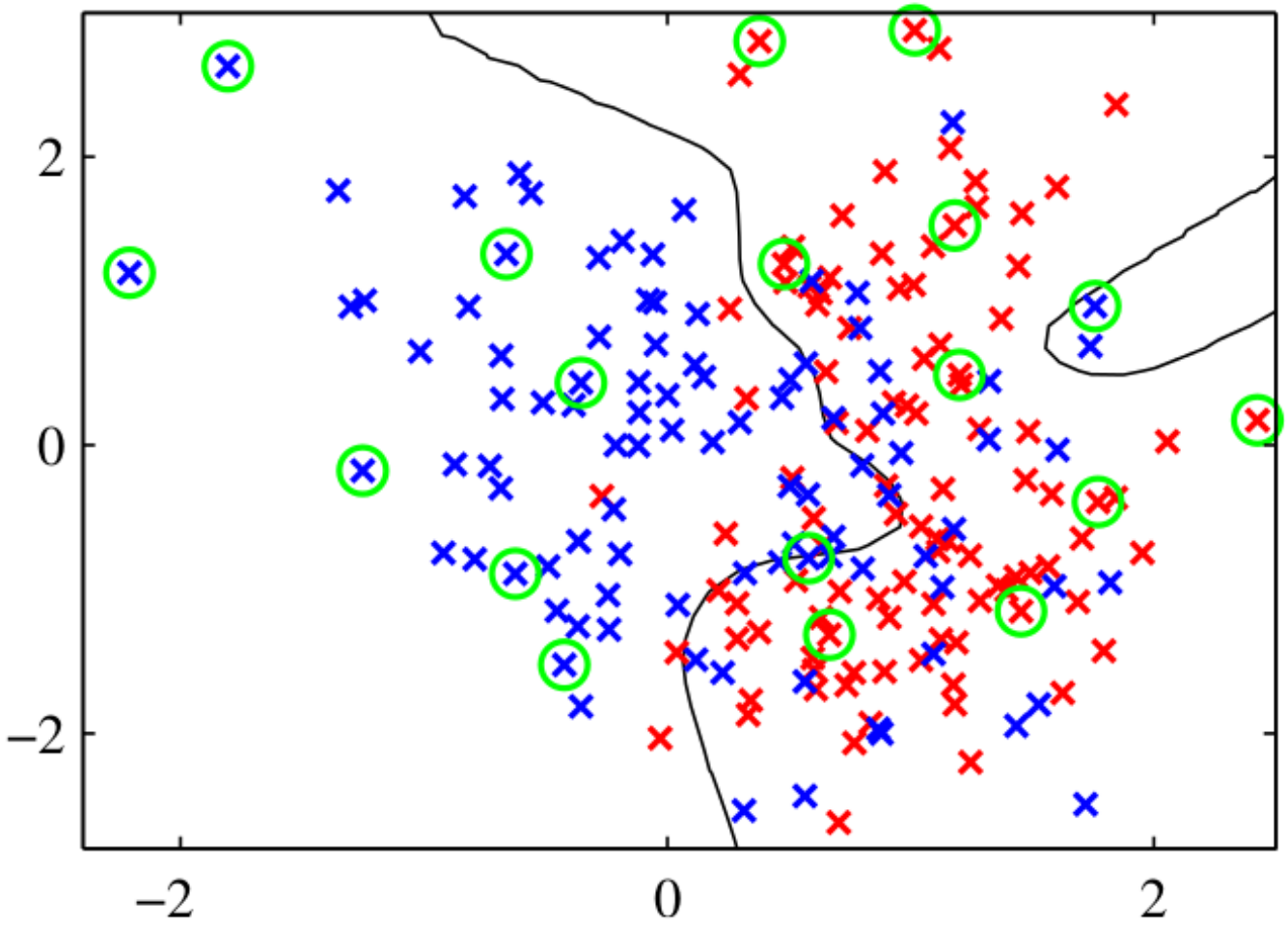
$$\alpha_i = \frac{\gamma_i}{(w_i^*)^2}, \ \gamma_i = 1 - \alpha_i \Sigma_{ii}.$$

- Т.е. формула получилась точно такая же, как в случае регрессии.

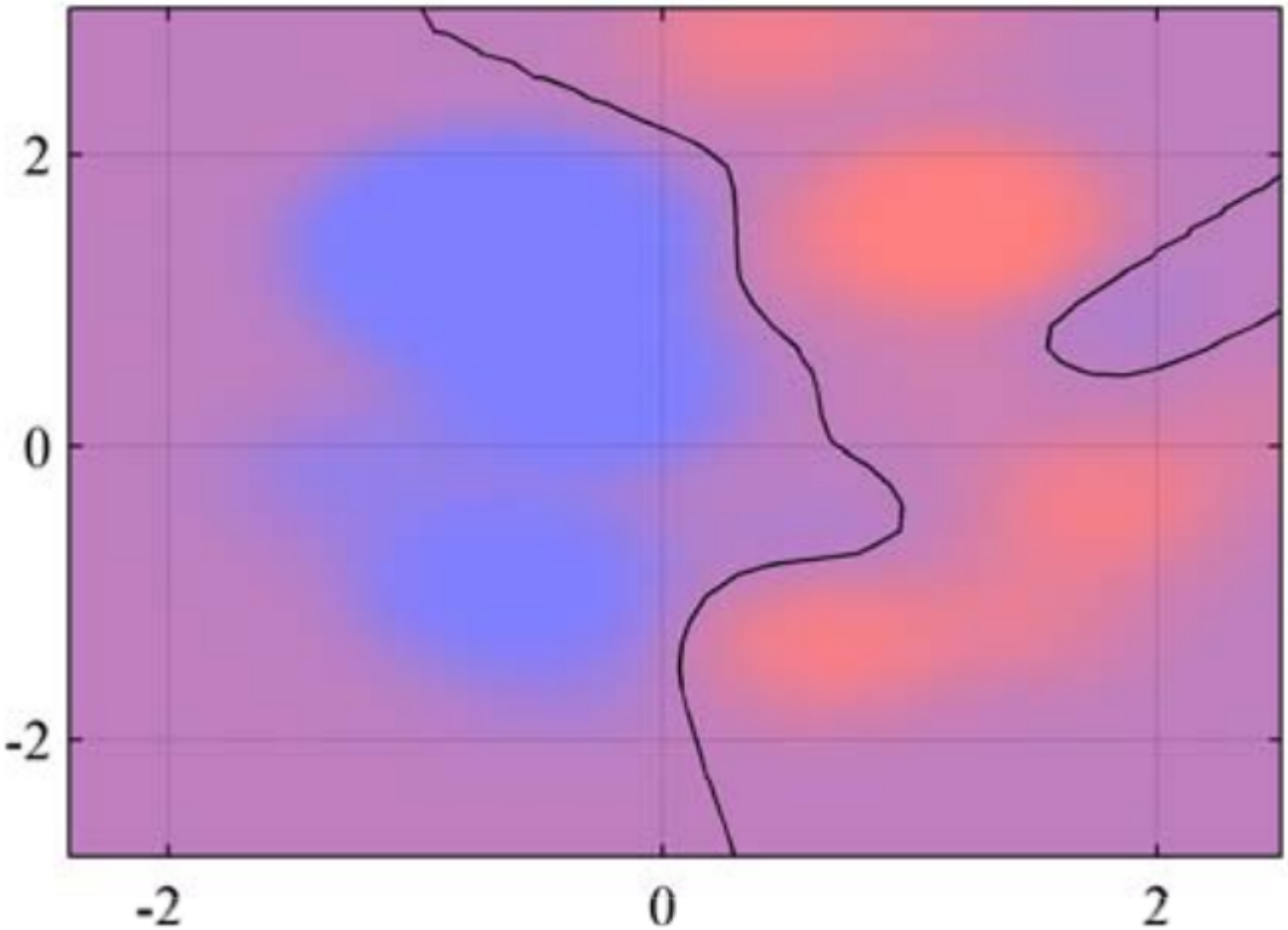
Было: SVM



Стало: RVM



Стало: RVM



RVM для нескольких классов

- На несколько классов теперь обобщается естественным образом:

$$a_k = \mathbf{w}_k^\top \mathbf{x}, \quad y_k(\mathbf{x}) = \frac{e^{a_k}}{\sum_j e^{a_j}}.$$

- И дальше всё то же самое.

Сравнение SVM и RVM

- RVM как-то получилось лучше со всех сторон.
- Главный минус – в RVM обучение гораздо дольше (хотя есть алгоритмы и побыстрее, чем мы рассматривали, но всё равно дольше).
- Но даже это не то чтобы минус, потому что в SVM нужна кросс-валидация для подбора параметров, т.е. на самом деле обучение SVM дольше, чем кажется.
- Есть и (даже более важный) плюс с точки зрения скорости – в RVM гораздо быстрее применение модели к новым точкам, потому что опорных векторов гораздо меньше.

Thank you!

Спасибо за внимание!

Быстрый алгоритм обучения RVM

- В RVM для регрессии получается правдоподобие

$$\ln p(\mathbf{t} \mid \mathbf{X}, \boldsymbol{\alpha}, \beta) = \ln \mathcal{N}(\mathbf{t} \mid \mathbf{0}, \mathbf{C}) =$$

$$= -\frac{1}{2}\left[N \ln(2\pi) + \ln|\mathbf{C}| + \mathbf{t}^\top \mathbf{C}^{-1}\mathbf{t}\right], \text{ где } \mathbf{C} = \beta^{-1}\mathbf{I} + \boldsymbol{\Phi}\mathbf{A}^{-1}\boldsymbol{\Phi}^\top.$$

Быстрый алгоритм обучения RVM

- Выделим вклад в $\mathbf{C}$ одного компонента α_i:

$$\mathbf{C} = \beta^{-1}\mathbf{I} + \sum_{j \neq i} \alpha_j^{-1} \boldsymbol{\varphi}_j \boldsymbol{\varphi}_j^\top + \alpha_i^{-1} \boldsymbol{\varphi}_i \boldsymbol{\varphi}_i^\top =$$

$$= \mathbf{C}_{-i} + \alpha_i^{-1} \boldsymbol{\varphi}_i \boldsymbol{\varphi}_i^\top,$$

где $\boldsymbol{\varphi}_i$ – i-я строка $\boldsymbol{\Phi}$ ($\boldsymbol{\Phi}_n$ был n-м столбцом).

Быстрый алгоритм обучения RVM

- $\mathbf{C} = \mathbf{C}_{-i} + \alpha_i^{-1}\boldsymbol{\varphi}_i\boldsymbol{\varphi}_i^\top$.
- Верны следующие тождества для определителя и обратной матрицы:

$$|\mathbf{C}| = |\mathbf{C}_{-i}||1 + \alpha_i^{-1}\boldsymbol{\varphi}_i^\top\mathbf{C}_{-i}^{-1}\boldsymbol{\varphi}_i|,$$

$$\mathbf{C}^{-1} = \mathbf{C}_{-i}^{-1} - \frac{\mathbf{C}_{-i}^{-1}\boldsymbol{\varphi}_i\boldsymbol{\varphi}_i^\top\mathbf{C}_{-i}^{-1}}{\alpha_i + \boldsymbol{\varphi}_i^\top\mathbf{C}_{-i}^{-1}\boldsymbol{\varphi}_i}.$$

Упражнение. Докажите это.

Быстрый алгоритм обучения RVM

- Значит, $L(\boldsymbol{\alpha})$ можно переписать в виде

$$L(\boldsymbol{\alpha}) = L(\boldsymbol{\alpha}_{-i}) + \lambda(\alpha_i), \text{ где}$$

$$\lambda(\alpha_i) = \frac{1}{2}\left[\ln \alpha_i - \ln(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right].$$

- Здесь

$$\begin{aligned} s_i &= \boldsymbol{\varphi}_i^\top \mathbf{C}_{-i}^{-1} \boldsymbol{\varphi}_i && \text{sparsity } \boldsymbol{\varphi}_i \\ q_i &= \boldsymbol{\varphi}_i^\top \mathbf{C}_{-i}^{-1} \mathbf{t} && \text{quality } \boldsymbol{\varphi}_i. \end{aligned}$$

Быстрый алгоритм обучения RVM

- $s_i = \boldsymbol{\varphi}_i^\top \mathbf{C}_{-i}^{-1} \boldsymbol{\varphi}_i$, $q_i = \boldsymbol{\varphi}_i^\top \mathbf{C}_{-i}^{-1} \mathbf{t}$.
- Sparsity – то, насколько $\boldsymbol{\varphi}_i$ перекрывается с остальными векторами модели.
- Quality – то, насколько $\boldsymbol{\varphi}_i$ сонаправлен с ошибкой между $\mathbf{t}$ и $\mathbf{y}_{-i}$ (ошибкой модели без $\boldsymbol{\varphi}_i$).
- Чем больше sparsity и чем меньше quality, тем более вероятно, что этот базисный вектор из модели исключат (т.е. $\alpha_i \to \infty$).

Быстрый алгоритм обучения RVM

- $\lambda(\alpha_i) = \frac{1}{2}\left[\ln \alpha_i - \ln(\alpha_i + s_i) + \frac{q_i^2}{\alpha_i + s_i}\right]$.
- Возьмём производную, приравняем нулю, получим (т.к. $\alpha_i \geq 0$)

$$\alpha_i = \begin{cases} \infty, & q_i^2 \leq s_i, \\ \frac{s_i^2}{q_i^2 - s_i}, & q_i^2 > s_i. \end{cases}$$

- Как мы и ожидали.

Быстрый алгоритм обучения RVM

- И алгоритм теперь получается такой:
  1. инициализировать β, $\boldsymbol{\varphi}_1$, $\alpha_1 = s_1^2/(q_1^2 - s_1)$, остальные $\alpha_j = \infty$;
  2. вычислить $\boldsymbol{\Sigma}$, $\mathbf{m}$, q_i и s_i для всех i;
  3. выбрать i, проапдейтить α_i, проапдейтить β;
  4. goto 2 и так пока не сойдётся.