

Теоретические задачи

3.1 Знакомство с линейного классификатором

1) Бинарный линейный классификатор:

$$a(x) = \begin{cases} 1 & , f(x) > 0 \\ -1 & , f(x) < 0 \end{cases} \quad (1)$$

где $f(x) = w_0 + w_1 \cdot x_1 + \dots + w_n \cdot x_n$

2) Отступ алгоритма $a(x) = \text{sign}\{f(x)\}$ на объекте x_i - величина $M_i = y_i \cdot f(x_i)$, где y_i - класс объекта x_i . Заметим, что

$$M_i \leq 0 \Leftrightarrow a(x) \neq y_i$$

$$M_i > 0 \Leftrightarrow a(x) = y_i$$

3) Классификаторы вида $a(x) = \text{sign}(\langle w, x \rangle - w_0)$ можно свести к классификаторам вида $a(x) = \text{sign}(\langle w, x \rangle)$, добавив к x ещё одну координату равную -1, тогда $x = (-1, x_1, x_2, \dots, x_n)^T$, $w = (w_0, w_1, \dots, w_n)^T$.

4) Функция эмпирического риска через отступы: $Q(w) = \sum_{i=1}^l [M_i(w) < 0]$. Этот функционал минимален для наилучшего алгоритма классификации (в самом лучшем случае равен 0).

5) Если функционал эмпирического риска равен $Q(w) = \sum_{i=1}^l [M_i(w) < 0]$, то при $w = 0$ $f(x_i) = \text{sign}(\langle w, x \rangle) = 0$ при любом x_i , значит $M_i(w) = 0$ для любого i и $Q(0) = 0$. Получим, что при $w = 0$ достигается минимум функционала для алгоритма $a(x) = \text{sign}(f(x))$.

6) Функционал аппроксимированного эмпирического риска - $\tilde{Q}(w) = \sum_{i=1}^l L(M_i(w))$, где L - функция потерь.

7) Функция потерь характеризует величину отклонения ответа алгоритма $y = a(x)$ от правильного ответа y^* . Через функцию потерь выражается функционал эмпирического риска, который равен ошибке алгоритма на заданной выборке. Среди заданных алгоритмов нужно найти тот, на котором функционал эмпирического риска минимален. Для того чтобы можно было применять градиентные методы оптимизации, берут непрерывные аппроксимации пороговой функции потерь: $Q(M) = (1 - M)^2$, $V(M) = (1 - M)_+$, $L(M) = \log_2(1 + e^{-M})$... Обычно функции потерь монотонные, гладкие, равны 1 в 0, стремятся к 0 на $+\infty$ и к бесконечности при $-\infty$.

8) Негладкая функция потерь - пороговая функция потерь $Q(w) = [M(w) < 0]$.

9) Регуляризация - это метод для уменьшения переобучения модели, к функционалу аппроксимированного риска добавляется штраф на веса (уменьшение переобучения происходит из-за того, что веса не могут быть сильно большими)

l1-регуляризация: $\gamma \cdot \sum_{k=1}^l |w_k|$

l2-регуляризация: $\gamma \cdot \sum_{k=1}^l w_k^2$

10) Алгоритм обладает обобщающей способностью, если вероятность ошибки на тестовой выборке несильно отличается от вероятности ошибки на обучающей выборке. При переобучении же вероятность ошибки на тестовой выборке значительно больше вероятности ошибки на обучающей выборке. То есть если алгоритм переобучился, то у него низкая обобщающая способность. Так как регуляризация уменьшает переобучение, она увеличивает обобщающую способность алгоритма.

11)

12) Так как регуляризация ограничивает веса алгоритма, из-за неё часть весов обнуляется, причем чем больше параметр γ , тем больше весов обнуляются. Регуляризация ответственна за отбор признаков, значит по сути она уменьшает число переменных функции риска, которые имеют значение.

13) Регуляризация предотвращает переобучение, но увеличивает ошибку алгоритма на обучающей выборке (при переобучении вероятность ошибки на обучающей выборке меньше, так как алгоритм подстраивается под данные)

14)

15) Рассмотрим задачу бинарной классификации, каждый объект относится к одному из двух классов - 0 и 1. Вводятся следующие метрики качества алгоритма:

Accuracy - доля правильных ответов (число правильных ответов / число ответов)

Precision - точность ответа (число правильно угаданных 1 / число раз, когда алгоритм ответил "1")

Recall - полнота ответа (число правильно угаданных 1 / число 1)

TP(True Positive) - истинное значение - 1, ответ алгоритма - 1

TN(True Negative) - истинное значение - 0, ответ алгоритма - 0

FP(False Positive) - истинное значение - 0, ответ алгоритма - 1

FN(False Negative) - истинное значение - 1, ответ алгоритма - 0

Тогда $Precision = \frac{TP}{TP+FP}$, $Recall = \frac{TP}{TP+FN}$

16) Метрика AUC - area under the curve - считает площадь под кривой. Метрика ROC-AUC - площадь под ROC-кривой. ROC-кривая строится в координатах TPR(FPR), где $TPR = \frac{TP}{TP+FN}$, $FPR = \frac{FP}{FP+TN}$. ROC-кривая проходит через точки (0, 0) (если алгоритм всегда выдает 0, то TP = FP = 0) и (1, 1) - (если алгоритм всегда выдаёт 1, то FN = TN = 0 и TPR = FPR = 1).

17) В контексте про фамилии строится линейный классификатор. Переобозначим классы как {1, -1} (1 - фамилия, -1 - не фамилия). Тогда этот линейный классификатор можно записать в виде $a(x) = \text{sign}(\langle w, x \rangle - w_0)$. Пусть после обучения веса алгоритма - \hat{w} . Заметим, что при $w_0 > \max_x (\langle \hat{w}, x \rangle)$ $a(x)$ дает -1 на всех элементах выборки, значит TPR = FPR = 0, при $w_0 < \min_x (\langle \hat{w}, x \rangle)$ $a(x) = 1$ на любом x , и TPR = FPR = 1. Для построения ROC-кривой будем варьировать w_0 от $\min_x (\langle \hat{w}, x \rangle)$ до $\max_x (\langle \hat{w}, x \rangle)$, при каждом w_0 считаем TPR и FPR: TPR = число правильно угаданных фамилий / общее число фамилий в тесте; FPR = число не фамилий, на которых алгоритм ответил, что это фамилия / общее число не фамилий.

3.2 Вероятностный смысл для регуляризатора

Пусть $p(x, y|w)$ - вероятностная модель данных, $p(w, \gamma)$ - априорное распределение в пространстве параметров модели (выборка порождена каждой плотностью $p(x, y|w)$ с вероятностью $p(w, \gamma)$), где γ - гиперпараметр, который можно варьировать. Запишем совместное правдоподобие данных и модели: $p(X^l, w) = p(X^l|w) \cdot p(w, \gamma)$. Тогда принцип максимального правдоподобия:

$$\ln(p(X^l, w)) = \sum_{i=1}^l \ln(p(x_i, y_i|w)) + \ln(p(w, \gamma)) \rightarrow \max_w$$

где $\ln(p(w, \gamma))$ - регуляризатор, не зависящий от данных и накладывающий ограничения на w .

Пусть вектор w имеет нормальное распределение, его компоненты независимы и имеют равные дисперсии σ . Запишем логарифм правдоподобия в этом случае:

$$\ln(p(w, C)) = \ln \left(\frac{1}{(2\pi\sigma)^{\frac{n}{2}}} \exp\left(-\frac{\sum_{i=1}^l w_i^2}{2\sigma}\right) \right) = \text{const}(w) - \frac{1}{2} \sum_{i=1}^l w_i^2$$

То есть l2-регуляризатор порождён нормальным распределением.

Пусть вектор w имеет распределение Лапласа с плотностью $\frac{C}{2} \exp(-C|w|)$

$$\ln(p(w, C)) = \ln \left(\frac{C^n}{2^n} \exp(-C \sum_{i=1}^l |w_i|) \right) = \text{const}(w) - C \cdot \sum_{i=1}^l |w_i|$$

l1-регуляризатор порождён распределением Лапласа.

3.3 SVM и максимизация разделяющей полосы

Пусть вектора принадлежат классам +1, -1 и разделены плоскостью $\langle w, x \rangle + w_0$. Нормируем векторы w, w_0 так, чтобы было выполнено $\min_i y_i (\langle w, x_i \rangle + w_0) = 1$. Пусть точки x_+, x_- - ближайшие к полосе точки классов +1 и -1 соответственно. Ширина полосы равна $\langle (x_+ - x_-), \frac{w}{\|w\|} \rangle$.

$$\left\langle (x_+ - x_-), \frac{w}{\|w\|} \right\rangle = \frac{\langle x_+, w \rangle - \langle x_-, w \rangle}{\|w\|} = \frac{w_0 + 1 - (w_0 - 1)}{\|w\|} = \frac{2}{\|w\|}$$

Получаем задачу оптимизации:

$$\begin{cases} \langle w, w \rangle \rightarrow \min \\ y_i (\langle w, x_i \rangle + w_0) \geq 1 \end{cases} \quad (2)$$

В случае, когда выборка линейно неразделима, вводим штраф за неправильную классификацию. Тогда задача оптимизации примет вид:

$$\begin{cases} \langle w, w \rangle + C \sum_{i=1}^l \xi_i \rightarrow \min \\ y_i (\langle w, x_i \rangle + w_0) \geq 1 - \xi_i \\ \xi_i \geq 0, i = 1, 2, \dots, l \end{cases} \quad (3)$$

3.4 Kernel Trick

Пусть есть разделяющая поверхность $x_1^2 + 2x_2^2 = 3$.

Рассмотрим функцию $K(a, b) = \langle a, b \rangle^2 = (a_1b_1 + a_2b_2)^2 = a_1^2 + b_1^2 + 2a_1a_2b_1b_2 = (a_1^2, \sqrt{2}a_1a_2, a_2^2) \begin{pmatrix} b_1^2 \\ \sqrt{2}b_1b_2 \\ b_2^2 \end{pmatrix}$

Заметим, что $K(x, w)$ задаёт ядро, называемое квадратичным ядром, оно сопоставляет вектору (x_1, x_2) вектор $(x_1^2, \sqrt{2}x_1x_2, x_2^2)$.

Разделяющая поверхность, построенная линейным классификатором задаётся уравнением $\langle x, w \rangle + w_0 = 0$. В случае квадратичного ядра: $(x_1^2, \sqrt{2}x_1x_2, x_2^2) \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} + w_0 = 0$, что при $w_0 = -3, w_1 = 1, w_2 = 0, w_3 = 2$ даёт исходную поверхность.

3.6 Было в номере 3.1