Школа анализа данных
Машинное обучение, часть 1
Теоретическое домашнее задание №2

Кузина Е.М.

10 апреля 2020

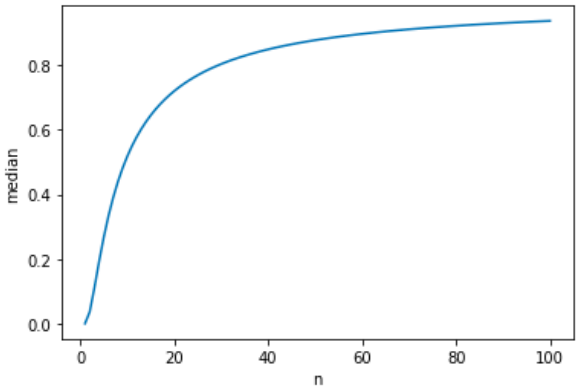
Решите предложенные задачи. Решения необходимо оформить в виде PDF документа. Каждая задача должна быть подробно обоснована, задачи без обоснования не засчитываются. Решения пишутся в свободной форме, однако так, чтобы проверяющие смогли разобраться. Если проверяющие не смогут разобраться в решении какой-нибудь задачи, то она автоматически не засчитывается.

Задача 1 (1.5 балла). Метрические методы, kNN, проклятие размерности.

Рассмотрим l точек, распределенных равномерно по объему n-мерного единичного шара с центром в нуле. Предположим, что мы хотим применить метод ближайшего соседа для точки начала координат. Зададимся вопросом, на каком расстоянии будет расположен ближайший объект. Для ответа на этот вопрос выведите выражение для **медианы** расстояния от начала координат до ближайшего объекта. Чтобы проинтерпретировать полученный результат, подставьте в формулу конкретные значения: $l = 500$ и $n = 10$. Покажите, как будет меняться значение медианы при дальнейшем увеличении размерности пространства при фиксированном количестве точек и постройте график этой зависимости. Поясните, в чем состоит проклятие размерности и почему полученная для медианы формула наглядно его демонстрирует. Для размерности n посчитайте, сколько точек $l = f(n)$ необходимо взять, чтобы побороть проклятье размерности.

Решение: Объем n-мерного единичного шара равен $c * 1^n$, где c-некоторая константа. Пусть r-радиус внутреннего шара с центром в начале координат. Тогда вероятность того, что все точки выборки лежат вне внутреннего шара равна $p = \left(\frac{c*1^n - c*r^n}{c*1^n}\right)^l = (1 - r^n)^l => r = (1 - p^{\frac{1}{l}})^{\frac{1}{n}}$. Тогда медиана расстояния от начала координат до ближайшего объекта равна $r = (1 - (\frac{1}{2})^{\frac{1}{l}})^{\frac{1}{n}}$.

При $l = 500$ и $n = 10$: $r = 0.51779$. Таким образом, можно сказать, что ближайший сосед будет находится примерно в середине шара.

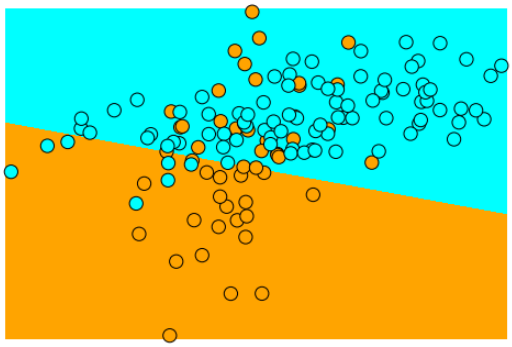


На графике видно, что при увеличении размерности медиана расстояния от начала координат до ближайшего соседа будет приближаться к 1, это означает, что выборка концентрируется возле границы шара.

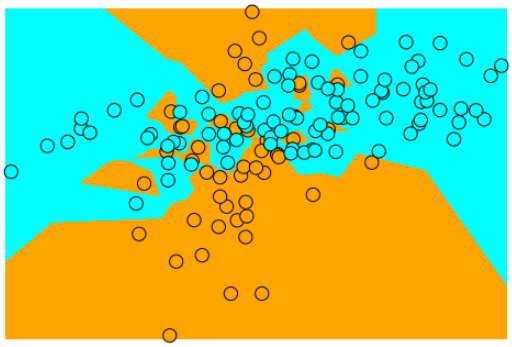
Проклятие размерности связано с экспоненциальным возрастанием количества данных из-за увеличения размерности пространства. В метрических классификаторах вычисляется расстояние между объектами. По графику видно, что точки концентрируются возле границы, то есть расстояние до них будет одинаковым, что в свою очередь ведет к неинформативности расстояния. Но об этом и говорит проклятие размерности: чтобы сохранить информативность, нужно увеличивать размер выборки (то есть с увеличением размерности пространства возрастает количество данных).

$l = f(n) = -\frac{1}{log_2(1-r^n)} = -\frac{1}{log_2(1-(\frac{1}{2})^n)}$ элементов нужно, чтобы побороть проклятие размерности.

Задача 2 (1 балл). Число степеней свободы алгоритма обучения.



Линейная модель



Метод ближайших соседей

Известно, что чем больше у метода машинного обучения настраиваемых параметров, тем больше он склонен к переобучению. Действительно, склонность к переобучению свидетельствует о «гибкости» модели, а «гибкость» говорит о большом количестве «степеней свободы» модели или, другими словами, параметров. Здесь под «гибкостью» будем неформально подразумевать способность алгоритма подстроиться под любые данные.

Рассмотрим несколько моделей. Линейные алгоритмы классификации имеют порядка n настраиваемых параметров (вектор весов), где n — размерность признакового пространства объектов. На левом рисунке показан результат работы линейного алгоритма для случая бинарной классификации в двумерном пространстве. Метод k ближайших соседей (kNN) имеет один настраиваемый параметр k, число соседей. На правом рисунке показан результат работы обученного kNN на тех же данных.

На этих примерах можно легко видеть, что kNN куда более «гибок», чем линейная модель. Однако kNN обладает всего одним параметром (число соседей), а линейная модель целым набором из n параметров. Почему так происходит? Что не так с приведенными выше рассуждениями?

Решение: Гибкость модели определяется не только параметрами модели, но и её "свойствами". Приведу свои объяснения на данных рисунках. Линейная классификация разбивает пространство на две части с помощью прямой так, чтобы сумма ошибок классификации была минимальной. В методе k ближайших соседей объект классифицируется к тому классу, представителей которого окажется больше среди его k ближайших соседей, т.е. он в отличие от линейной классификации учитывает те объекты, которые расположены рядом с ним.

Понятно, что в зависимости от данных одни алгоритмы работают лучше других. Некоторые из них будут лучше учитывать "особенности"выборки, другие-хуже. В данном примере KNN лучше

подстраивается под данные (более гибок) из-за особенной метода (следует рассмотреть ближайшие k соседей), чем линейная классификация. Следовательно, в приведенных рассуждениях постановки задачи не правильно понимать, что гибкость зависит только от количества параметров модели.

Задача 3 (1.5 балла) Кривые ROC и Precision-Recall.

Как известно, ROC и Precision-Recall (PR) кривые строятся по точкам, соответствующим результатам классификации при различных значениях порога, и каждая из кривых соединяет две диагонально противоположные точки квадрата $[0, 1]^2$. Опишите более детально поведение этих кривых, а именно:

1. (0.25 балла) Всегда ли ROC-кривая не убывает?

   Решение: Да. Если посмотреть на алгоритм построения ROC кривой, можно увидеть, что кривая строится горизонтально вправо или вертикально вверх ($TPR_i = TPR_{i-1}$ или $TPR_i = TPR_{i-1} + \frac{1}{l_+}$).

2. (0.25 балла) Всегда ли PR-кривая не возрастает?

   Решение: PR кривая строится в precision(точность)-recall(полнота) координатах

   PR-кривая может как возрастать, так и убывать:

   (a) если объект принадлежит классу 1, то точность и полнота увеличиваются => кривая поднимается вправо вверх

   (b) если объект принадлежит классу -1, то точность уменьшается, полнота не изменяется => кривая опускается строго вниз

3. (0.5 балла) В каких случаях предпочтительнее использовать ROC, а в каких PR кривую?

   Решение: ROC кривую лучше использовать, когда классы сбалансированы, иначе лучше использовать PR кривую. ссылка

   Проблема с ROC связана с тем, что если положительный класс существенно меньше по размеру, то он может давать не адекватную оценку. Например, если алгоритм правильно распознает 95 объектов из 100 положительного класса и не верно распознает 50000 из 1000000, то $TPR = 0.95$ и $FPR = 0.05$. Оценки близки к идеальными, но не отражают действительность.

4. (0.5 балла) Предложите алгоритм, который за $O(n \log n)$ вычисляет площадь под PR-кривой (иногда эту величину называют Average Precision (AP), хотя не все согласны с такой терминологией).

   Решение:

$$Precision = \frac{\sum_{i=1}^{n} [a(x_i) = 1][y_i = 1]}{\sum_{i=1}^{n} [a(x_i) = 1]}$$

$$Recall = \frac{\sum_{i=1}^{n} [a(x_i) = 1][y_i = 1]}{\sum_{i=1}^{n} [y_i = 1]}$$

   Алгоритм:

   $n_+ = \sum_{i=1}^{n} [y_i = 1]$ $O(n)$

Упорядочить выборку X^l по убыванию $f(x_i, w)$ $O(n\log n)$

$(Precision_0, Recall_0) = (0, 0)$

$count = 0$

$AP = 0$

for $i = 1, \ldots, n : O(n)$

  if $y_i = 1$:

    $count = count + 1$

    $Precision_i = \frac{count}{i}$

    $Recall_i = Recall_{i-1} + \frac{1}{n_+}$

    $AP = AP + \frac{1}{n_+} \frac{Precision_i + Precision_{i-1}}{2}$

  else:

    $Precision_i = \frac{count}{i}$

    $Recall_i = Recall_{i-1}$

Асимптотика алгоритма: $O(n) + O(n\log(n)) + O(n) = O(n\log(n))$

Задача 4 (1 балл) ROC AUC.

Определим понятие доли дефектных пар ответов классификатора. Пусть дан классификатор $a(x)$, который возвращает оценки принадлежности объектов классам: чем больше ответ классификатора, тем более он уверен в том, что данный объект относится к классу «+1». Отсортируем все объекты по неубыванию ответа классификатора a: $x_{(1)}, \ldots, x_{(\ell)}$. Обозначим истинные ответы на этих объектах через $y_{(1)}, \ldots, y_{(\ell)}$. Тогда доля дефектных пар записывается как

$$\mathrm{DP}(a, X^\ell) = \frac{2}{\ell(\ell-1)} \sum_{i<j}^{\ell} \left[y_{(i)} > y_{(j)}\right].$$

Как данный функционал связан с AUC (площадью под ROC-кривой)? Ответ должен быть дан в виде формулы, связывающей DP и AUC.

Решение:

Если рассмотреть алгоритм вычисления AUC, можно заметить, что ROC-кривая состоит из прямоугольников с шириной $\frac{1}{l_-}$ и высотой $\frac{1}{l_+} \sum\limits_{j=i+1}^{l} [y_{(j)} = 1]$.

$$AUC(a, X^l) = \frac{1}{l_-} \sum_{i=1}^{l} [y_{(i)} = -1] \frac{1}{l_+} \sum_{j=i+1}^{l} [y_{(j)} = 1] = \frac{1}{l_- l_+} \sum_{i=1}^{l} [y_{(i)} = -1] \sum_{j=i+1}^{l} [y_{(j)} = 1] = \frac{1}{l_- l_+} \sum_{i<j}^{l} [y_{(i)} < y_{(j)}] =$$

$$= \frac{1}{l_- l_+} \sum_{i<j}^{l} \Big(1 - [y_{(i)} > y_{(j)}] - [y_{(i)} = y_{(j)}]\Big) = \frac{1}{l_- l_+} \sum_{i<j}^{l} \Big(1 - [y_{(i)} = y_{(j)}]\Big) - \frac{1}{l_- l_+} \sum_{i<j}^{l} [y_{(i)} > y_{(j)}] =$$

$$= \frac{1}{l_- l_+} \sum_{i<j}^{l} [y_{(i)} \neq y_{(j)}] - \frac{1}{l_- l_+} \sum_{i<j}^{l} [y_{(i)} > y_{(j)}] = \frac{l_- l_+}{l_- l_+} - \frac{1}{l_- l_+} \sum_{i<j}^{l} [y_{(i)} > y_{(j)}] = 1 - \frac{l(l-1)}{2 l_- l_+} DP(a, X^l)$$

$$DP(a, X^l) = \frac{2 l_- l_+ (1 - AUC(a, X^l))}{l(l-1)}$$

Задача 5 (2 балла). Метрики качества.

Для каждой задачи машинного обучения есть множество различных метрик, оценивающих качество ее решения. При этом на практике иногда оказывается, что по одной метрике алгоритм может иметь более высокое качество, а по другой — более низкое. Предположим, что мы хотим сравнить между собой два алгоритма бинарной классификации, предсказывающих вероятность принадлежности одному из классов. Качество алгоритмов оценивается на тестовой выборке из **пяти** объектов. Оказывается, что по метрике A первый алгоритм превосходит второй, а по метрике B второй алгоритм превосходит первый. Приведите примеры тестовых меток y и векторов предсказаний $\tilde{y}_1$ и $\tilde{y}_2$ в случае каждой пары метрик A и B:

1. A — auc-roc, B — log-loss;

   Решение: Чем больше auc-roc, тем лучше алгоритм. Чем меньше log-loss, тем лучше алгоритм.

   $y_{true} = [1, 0, 0, 0, 1], y1 = [0.5, 0.4, 0.4, 0.4, 0.55], y2 = [0.3, 0.5, 0.3, 0.2, 0.8]$

   $roc - auc_1 = 1, roc - auc_2 = 0.75$

   $log - loss_1 = 0.5646922105227075, log - loss_2 = 0.5400164062906067$

   Таким образом, roc-auc лучше у первого алгоритма, а log-loss - у второго.

2. A — precision (при пороге 0.5), B — auc-roc;

   Решение: Чем больше precision, тем лучше алгоритм.

   $y_{true} = [1, 1, 0, 0, 1]$

   $y1_{prob} = [0.3, 0.4, 0.48, 0.45, 0.8], y1_{pred} = [0, 0, 0, 0, 1]$

   $y2_{prob} = [0.45, 0.55, 0.52, 0.4, 0.55], y2_{pred} = [0, 1, 1, 0, 1]$

   $precision_1 = 1.0, precision_2 = 0.6666666666666666$

   $roc - auc_1 = 0.3333333333333333, roc - auc_2 = 0.8333333333333333$

   Таким образом, precision лучше у первого алгоритма, а roc-auc - у второго.

3. A — precision (при пороге 0.5), B — recall (при пороге 0.5);

   Решение: Чем лучше recall, тем лучше.

   На тех же самых значениях из пункта 2 получаем:

   $recall_1 = 0.3333333333333333, recall_2 = 0.6666666666666666$

   Тогда первый алгоритм лучше по метрике precision, а второй - по recall.

4. A — F1-score (при пороге доставляющем максимум), B — auc-roc.

   Решение: Чем выше $F1 - score$, тем лучше алгоритм.

   $y_{true} = [1, 0, 0, 1, 1]$

   $y1 = [0.3, 0.8, 0.2, 0.9, 0.8], y2 = [0.5, 0.4, 0.4, 0.4, 0.55]$

   $f1 - score_1 = 0.8571428571428571, \text{ порог} - 0.29$

   $f1 - score_2 = 0.8, \text{ порог} - 0.49$

   $roc - auc_1 = 0.75, roc - auc_2 = 0.8333333333333333$

   Получаем, что первый алгоритм лучше по метрике f1-score, а второй - по roc-auc.

Задача 6 (1.5 балла). Метрики качества.

Костя участвует в конкурсе по анализу данных, в котором нужно решить задачу бинарной классификации, в которой для оценивания качества используется функционал ошибки Mean Absolute Error (MAE):

$$Q(\boldsymbol{y}, \tilde{\boldsymbol{y}}) = \frac{1}{l}\sum_{i=1}^{l} |y_i - \tilde{y}_i|, \quad \boldsymbol{y} \in \{0,1\}^l, \quad \tilde{\boldsymbol{y}} \in [0,1]^l,$$

где l — количество обучающих объектов, $\boldsymbol{y}$ — вектор истинных классов объектов, $\tilde{\boldsymbol{y}}$ — вектор предсказанных «степеней принадлежности» классу 1, качество которого и оценивается. Костя заметил, что качество предсказания на скрытой выборке, которая доступна только организаторам конкурса, часто улучшается, если сдвинуть прогноз $\tilde{y}_i$ в один из концов отрезка $[0,1]$ для каждого объекта i. Объясните, почему Костины действия привели к улучшению качества предсказания. При каких обстоятельствах Костя мог провернуть такой трюк? Всегда ли такое возможно?

Подсказка: Сделайте предположение, что объект с номером i принадлежит классу 1 с истинной вероятностью p_i.

***** Оцениваться будут рассуждения, каким условиям должны удовлетворять предсказания, чтобы Костина идея сработала, а также подтверждающие и/или опровергающие примеры.

Решение: Найдем матожидание функционала ошибки:

$$E(Q(y,\hat{y})|x) = E(|y_i - \hat{y_i}|\big|x_i) = |1-\hat{y_i}|p(y_i=1|x_i) + |0-\hat{y_i}|(1-p(y_i=1|x_i)) = (1-\hat{y_i})p(y_i=1|x_i) + \hat{y_i}(1-p(y_i=1|x_i)),$$

где $p(y_i = 1|x_i)$-истинная вероятность того, что объект с номером i принадлежит классу 1

$$\frac{\partial E(Q(y,\hat{y})|x)}{\partial \hat{y_i}} = 1 - 2p(y_i=1|x_i) = 0 => p(y_i=1|x_i) = \frac{1}{2}$$

Если $p(y_i = 1|x_i) = \frac{1}{2}$, то $E(Q(y,\hat{y})) = \frac{1}{2}$. Получаем, что для любого предсказания вероятности матожидание функционала ошибки равна $\frac{1}{2}$, поэтому будут предсказаны не верные вероятности.

Если истинная вероятность $p(y_i = 1|x_i) \neq \frac{1}{2}$, тогда минимум матожидания будет достигаться на концах отрезка $[0,1]$, т.е. $minE(Q(y,\hat{y})|x) = min(p(y_i|x_i), 1 - p(y_i|x_i))$.

Можно сделать следующие выводы:

1. данный функционал ошибки не позволяет предсказывать верные вероятности принадлежности к классу 1
2. Костя смог улучшить свои предсказания, сдвинув их в один из концов отрезка $[0,1]$, т.к. истинные вероятности не равны $\frac{1}{2}$ и его модель не предсказывает верные вероятности в силу неверно выбранного функционала ошибки

Приведем примеры:

1. $y_{true} = [1,1,1,0,0], y_{prob} = [0.4, 0.5, 0.4, 0.3, 0.1]$ (предсказанные вероятности)

   MAE в данном случае будет равен 0.42. Теперь попробуем сдвинуть все предсказания в 1, тогда MAE равен 0.4. => оценка улучшилась
2. Аналогично можно построить пример, когда все предсказания сдвигаются в 0 и MAE уменьшается.
3. $y_{true} = [1,1,1,0,0,0], y_{prob} = [0.7, 0.5, 0.6, 0.4, 0.3, 0.6]$ (предсказанные вероятности)

   MAE в данном примере увеличивается с 0.417 до 0.5, если сдвинуть предсказания в 0 или 1.

4. $y_{true} = [1, 1, 0, 0, 0], y_{prob} = [0.4, 0.5, 0.4, 0.3, 0.1]$

   MAE = 0.38, MAE для всех предсказаний, равных 1, равен 0.6, а для всех предсказаний, равных 0, равен 0.4

Из примеров можно сделать вывод, что у Кости могла быть тестовая выборка, в которой объектов одного класса больше, поэтому MAE **может** уменьшится (например, в 4 примере данная тактика не работает, а в 1 работает). Также следует отметить, что для правильных предсказаний вероятностей не нужно использовать данный функционал ошибки.

Задача 7 (1.5 балла). SVM, функция потерь.

Классический линейный метод опорных векторов приводит к задаче квадратичного программирования:

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n \to \min_{\boldsymbol{w},\boldsymbol{\xi}}, \quad \text{s.t.} \quad y_n\boldsymbol{w}^T\boldsymbol{x}_n \geq 1 - \xi_n, \quad \xi_n \geq 0 \quad \forall n.$$

Какую функцию потерь оптимизирует линейный метод опорных векторов? Постройте задачу квадратичного программирования аналогичную SVM, но для оптимизации функции потерь:

$$\mathcal{L}(M) = \begin{cases} (M-1)^2, & M < 1 \\ 0, & M \geq 1, \end{cases}$$

где $M = y(\boldsymbol{w}^T\boldsymbol{x} - \boldsymbol{w}_0)$ — отступ объекта $\boldsymbol{x}$. Выпишите двойственную к ней задачу.

Решение:

Линейный метод опорных векторов оптимизирует функцию потерь $(1 - M)_+$, где $M = yw^Tx$ - отступ объекта x.

Теперь построим задачу квадратичного программирования (ЗКП), аналогичную SVM для оптимизации функции потерь $\mathcal{L}$. Можно заметить, что:

$$\mathcal{L}(M) = \begin{cases} (1-M)^2, & M < 1 \\ 0, & M \geq 1, \end{cases}$$

Тогда ЗКП:

$$\begin{cases} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n^2 \to \min_{\boldsymbol{w},\boldsymbol{w_0},\boldsymbol{\xi}} \\ M_n(\boldsymbol{w}, \boldsymbol{w_0}) \geq 1 - \xi_n \quad \forall n \\ \xi_n \geq 0 \quad \forall n \end{cases}$$

Теперь построим двойственную к ней задачу. Запишем Лагранжиан:

$$L(\boldsymbol{w}, \boldsymbol{w_0}, \boldsymbol{\xi}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n^2 - \sum_{n=1}^{N}\lambda_i(M_n(\boldsymbol{w}, \boldsymbol{w_0}) - 1 + \xi_n) - \sum_{n=1}^{N}\mu_n\xi_n$$

Условия Каруша-Куна-Такера (ККТ):

$$\begin{cases} \frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{w} - \sum_{n=1}^{N}\lambda_n y_n x_n = 0 => \boldsymbol{w} = \sum_{n=1}^{N}\lambda_n y_n x_n \\ \frac{\partial L}{\partial \boldsymbol{w_0}} = \sum_{n=1}^{N}\lambda_n y_n = 0 \\ \frac{\partial L}{\partial \xi_n} = 2C\xi_n - \lambda_n - \mu_n = 0 => \xi_n = \frac{\lambda_n + \mu_n}{2C} \\ \xi_n \geq 0, \lambda_n \geq 0, \mu_n \geq 0 \quad \forall n \\ \lambda_n = 0 \text{ либо } M_n(\boldsymbol{w}, \boldsymbol{w_0}) = 0 \quad \forall n \\ \mu_n = 0 \text{ либо } \xi_n = 0 \quad \forall n \end{cases}$$

Тогда двойственная задача выглядит следующим образом:

$$\begin{cases} -L(\boldsymbol{w}, \boldsymbol{w_0}, \boldsymbol{\xi}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = -\sum_{n=1}^{N} \lambda_n + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j \langle x_i, x_j \rangle + \sum_{n=1}^{N} \frac{(\lambda_n + \mu_n)^2}{4C} \to \min_{\lambda, \mu} \\ \lambda_n \geq 0, \mu_n \geq 0 \quad \forall n \\ \sum_{n=1}^{N} \lambda_n y_n = 0 \end{cases}$$