Московский физико-технический институт (ГУ)

Физтех-школа прикладной математики и информатики

Методы оптимизаций
Исследовательский проект

Kernel Trick

Преподаватель: Анна Руденко

Студент: Ален Алиев

Специальность: Прикладная математика и информатика

Семестр: Осень 2021, 5 семестр

E-Mail: aliev.ae@phystech.edu

Долгопрудный, 2021

Содержание

Вступление

Метод Kernel Trick используется в широком спектре задач - от геостатистики и биоинформатики до распознавания рукописного почерка.

Это один из самых популярных методов оптимизации, используемых при классификации данных и детекции аномалий, основанный на вполне естественных предположениях, к тому же несложен в реализации и применении.

План

1. Объяснить мотивацию, лежащую за методом Kernel Trick.
2. Изучить различные применения Kernel Trick в задачах классификации.
3. Проанализировать возможные преимущества использования метода при решении задач SVM и логистической регресии.

Пререквизиты

Введем основные понятия, которыми далее будем оперировать.

1. **Постановка задачи**

   На вход подаются данные в формате $X \in \mathcal{R}^{n\times d}$, где n - размер выборки, d - размерность пространства признаков и $y \in \mathcal{R}^n$ - таргетный признак. Тогда наша задача - построить модель, которая будет предсказывать таргетный признак по входу из вектора размера d. В случае, когда таргетный признак является категориальным, будем говорить, что это задача классификации, если же класса всего 2 - то бинарной классификации.

2. **Используемые методы**

   - Линейная регрессия

     Предполагаем, что зависимость таргета от признаков линейная и вводим функцию штрафа: $Q(\theta) = \|X\theta - y\|^2 + \lambda G(\theta) \to \min$, где $G(\theta)$ - регуляризатор. Если $G(\theta) \equiv \|\theta\|^2$, то будем говорить о Ridge-регресии.

   - Метод опорных векторов(SVM)

     Хотим разделить выборку гиперплоскостью, но понимаем, что это скорее всего невозможно. Определим задачу следующим образом: $\begin{cases} \frac{1}{2}\|\theta\|^2 + C\sum\limits_{i=1}^{d}\epsilon_i \to \min\limits_{\theta,b,\epsilon} \\ 0 \leq \epsilon_i \\ y_i(\langle\theta, x_i\rangle + b) \geq 1 - \epsilon_i \end{cases}$.

Здесь $\frac{2}{\|\theta\|}$ - ширина разделяющей полосы, $\frac{1}{\|\theta\|}$ - мягкий отступ, ϵ - штраф за попадание внутрь разделяющей полосы. Чем больше C, тем сильнее мы ориентируемся на обучающую выборку.

Эта задача является выпуклой и имеет единственное решение, а еще она удовлетворяет условиям Каруша-Куна-Таккера, чем мы не постеснялись воспользоваться.

- Ядра

  Ядро - скалярное произведение в некотором пространстве. Согласно теореме Мерсера, функция $K(x, z)$ является ядром тогда и только тогда, когда она симметрична и неотрицательно определена. Проверять эти условия на практике зачастую сложно, поэтому обычно пользуются конструктивными признаками:

  Theorem 1 *Пусть $K_1(x, z)$ и $K_2(x, z)$ — ядра, заданные на множестве X, $f(x)$ - вещественная функция на X, $\phi : X \to \mathbb{R}^N$ - векторная функция на X, K_3 - ядро, заданное на $\mathbb{R}^N$. Тогда следующие функции являются ядрами:*

  (a) $K(x, z) = K_1(x, z) + K_2(x, z)$,

  (b) $K(x, z) = \alpha K_1(x, z)$, $\alpha > 0$,

  (c) $K(x, z) = K_1(x, z)K_2(x, z)$,

  (d) $K(x, z) = f(x)f(z)$,

  (e) $K(x, z) = K_3(\phi(x), \phi(z))$.

- Основные типы ядер

  (a) Линейное

  $K(x, z) = \langle x, z\rangle$

  Базовое ядро

  (b) Полиномиальное

  $K(x, z) = (\langle x, z\rangle + R)^m$

  Соответсвует переводу набора признаков в мономы степени не более m, R регулирует вес признаков.

  (c) Radial Basis Function(RBF) или Гауссово

  $K(x, z) = exp(-\frac{\|x - z\|^2}{2\sigma^2})$

  (d) Sigmoid

  $K(x, z) = \tanh(a\langle x, z\rangle + R)$

- Random Fourier Features

  Из комплексного анализа известно, что любое непрерывное ядро вида $K(x, z) = K(x - z)$ является преобразованием Фурье некоторого вероятностного распределения (теорема Бохнера):

$$K(x - z) = \int_d p(\theta)e^{i\theta^T(x-z)}d\theta.$$

Преобразуем интеграл:

$$\int_d p(\theta)e^{i\theta^T(x-z)}d\theta = \int_d p(\theta)\cos(\theta^T(x-z))d\theta + i\int_d p(\theta)\sin(\theta^T(x-z))d\theta = \\ = \int_d p(\theta)\cos(\theta^T(x-z))dw.$$

Поскольку значение ядра $K(x-z)$ всегда вещественное, то и в правой части мнимая часть равна нулю, а значит, остаётся лишь интеграл от косинуса $\cos\theta^T(x-z)$. Мы можем приблизить данный интеграл методом Монте-Карло:

$$\int_d p(\theta)\cos(\theta^T(x-z))d\theta \approx \frac{1}{n}\sum_{j=1}^{n}\cos(\theta_j^T(x-z)),$$

где векторы $\theta_1,\ldots,\theta_n$ генерируются из распределения $p(\theta)$. Используя эти векторы, мы можем сформировать аппроксимацию преобразования $\phi(x)$:

$$\tilde{\phi}(x) = \frac{1}{\sqrt{n}}(\cos(\theta_1^T x),\ldots,\cos(\theta_n^T x),\sin(\theta_1^T x),\ldots,\sin(\theta_n^T x)).$$

Данная оценка является несмещённой для $K(x,z)$ в силу свойств метода Монте-Карло. Более того, с помощью неравенств концентрации меры можно показать, что дисперсия данной оценки достаточно низкая. Заметим, что синусы можно заменить на косинусы со сдвигом, что немного упрощает реализацию метода.

Отчет

- **Общая мотивация**

  Рассматривается выборка, сгенерированная с помощью make_circles - 2 вложенных круга с небольшим шумом.

  Из здравого смысла несложно видеть разделяющее пространство - круг между нашими двумя. Тем не менее, обычные линейные методы очевидно затрудняются построить хорошее приближение.

  Добавим всего один признак, уже визуально отчетливо видно, что выборка в новом пространстве легко разделяется гиперплоскостью - то, чего мы и добивались. Оказывается, эту идею перехода к новому пространству признаков можно обобщить и на менее тривиальные ситуации.

- **Анализ результатов SVM с различными ядрами**

  Рассматривается выборка ijcnn1. Этот датасет использовался в рамках конкурса нейросетей *IJCNN*.

  Данные были разбиты случайно на трейн и тест в соотношении 7:3.

Полученные результаты следующие - на тестовой выборке размера ≈ 35000 примерно в 2 раза быстрее обучались SVM с ядрами $poly$, rbf, и показали заметно лучший перформанс относительно $linear$:
увеличение $ROC-AUC$ метрики на ≈ 0.08 и $Precision-Recall-Area$ на ≈ 0.3.
$Sigmoid$ ядро провалилось по всем статьям - долго обучалось и показало наихудший результат, что свидетельствует о том, что этот метод наиболее чувствителен к исходной задаче и подбору гиперпараметров.

- **Приближение ядер с помощью RFF**

  Рассматривается выборка fashion_mnist из пакета Tensorflow.
  Так как прямое использование ядерных функций может быть затратно по времени и памяти, иногда помогает метод аппроксимации ядер через построение случайных признаков на основе исходных.
  Реализованное приближение гауссовского ядра через RFF показало хороший результат - обучается быстрее, чем библиотечный подсчет напрямую, при этом не проседая в эффективности. Мы явно воспользовались тем, что для гауссовых ядер легко построить распределение, позволяющее применить RFF напрямую.
  Снова увидели неустойчивость $sigmoid$ ядра, которое и здесь оказалось неудачным выбором.

Результаты на тестовой выборке	
Метод	Accuracy
SVM, gaussian kernel, RFF	0.8715
LogReg, gaussian kernel, RFF	0.8574
SVM, linear kernel	0.8464
SVM, polynomial kernel	0.863
SVM, rbf kernel	0.8828
SVM, sigmoid kernel	0.4321

Использованная литература

R. Ravinder, Ramadevi Yellasiri, K.V.N. Sunitha:
Anomaly Detection using Feature Selection and SVM Kernel Trick, 2015

Christian Bauckhage:
Lecture Notes on Machine Learning: The Kernel Trick

Myung-Hoe Huh:
Kernel-Trick Regression and Classification

Varlam Kutateladze:
The Kernel Trick for Nonlinear Factor Modeling

Evgeniy Sokolov:
Lecture Notes on Machine Learning: The Kernel Trick, 2020 HSE