استاد: فاطمه منصوری نیمسال دوم ۱۴۰۲-۱۴۰۳

الگوریتم k - نزدیک‌ترین همسایه

یادگیری عمیق

گردآورنده: سینا جعفری

۱ مقدمه

الگوریتم k - نزدیک‌ترین همسایه یا به اختصار kNN [۱] ، به عنوان یکی از ساده‌ترین و سبک‌ترین الگوریتم‌ها در حوزه یادگیری یاد می‌شود که اولین بار توسط اویلن [۲] و همکاران در سال ۱۹۵۱ ارائه شد [۱]. در این جلسه قرار است الگوریتم kNN را مورد بررسی قرار دهیم و ویژگی‌های مختلف این الگوریتم اعم از نحوه کارکرد، پیچیدگی زمانی، مزایا و معایب آن را به طور کامل تجزیه و تحلیل کنیم.

۲ الگوریتم kNN

الگوریتم kNN یک روش از نوع یادگیری نظارت شده است و جز الگوریتم‌های بدون‌ـ پارامتر [۳] و همچنین *lazy* به حساب می‌آید. منظور از بدون‌ـ پارامتر بودن این الگوریتم آن است که در طول فرآیند آموزش و آزمایش، به هیچ پارامتری برای بروزرسانی نیاز ندارد و همچنین منظور از *lazy* بودن این الگوریتم نیز آن است که اصلاً فرآیند آموزش ندارد. درواقع مرحله آموزش این الگوریتم، تنها شامل ذخیره کردن داده‌ها درون حافظه است و همین امر موجب آن شده است که این الگوریتم تبدیل به یکی از سریع‌ترین و ساده‌ترین الگوریتم‌های حوزه یادگیری شود. این الگوریتم، در مسائل مختلفی همچون دسته‌بندی و رگرسیون می‌تواند استفاده شود که اغلب از آن برای دسته‌بندی استفاده می‌شود، که در ادامه هر دو مورد را توضیح خواهیم داد.

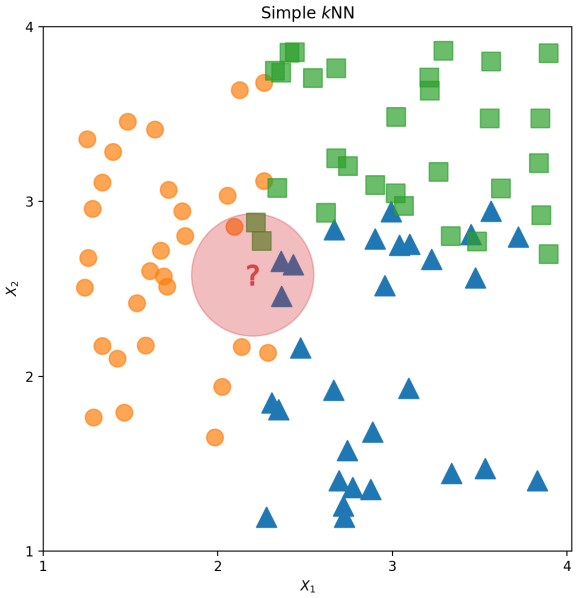
۱.۲ دسته‌بندی

هدف اصلی الگوریتم kNN در مسائل دسته‌بندی به این صورت است که باید برچسب [۴] یک داده آزمایشی [۵] را پیش‌بینی کند. به این نحو که ابتدا فاصله اقلیدسی داده آزمایشی را با تمامی داده‌های آموزشی [۶] محاسبه کرده، سپس این فواصل را به ترتیب صعودی مرتب کرده و k تا داده آموزشی اول که کمترین فواصل را با آن داده آزمایشی دارند را انتخاب کرده و بین برچسب‌های این k تا داده آموزشی، رأی‌گیری [۷] انجام داده و آن دسته‌ای که بیشترین حضور را در آن k تا داده آموزشی دارد را به عنوان برچسب داده آزمایشی انتخاب می‌کند (شکل ۱). همچنین اگر در مرحله آخر یعنی رأی‌گیری، به تعداد یکسانی داده از هر دسته موجود بود، به تصادف یکی از آن‌ها را انتخاب می‌کنیم. درنهایت، الگوریتم kNN ، یک الگوریتم قطعی بوده و به ازای هر داده آزمایشی جدید، می‌تواند یک برچسب برای آن انتخاب کند.

شبه کد kNN در الگوریتم ۱ قرار داده شده است. ورودی‌های این الگوریتم داده‌های آموزشی، یک داده آزمایشی و مرتبه یا همان مقدار k و خروجی این الگوریتم داده آزمایشی خواهد بود که برچسب‌گذاری شده‌اند. در خط ۳ این الگوریتم، یک مجموعه تهی تعریف شده

[1] k-Nearest Neighbor
[2] Evelyn
[3] non-parametric
[4] target
[5] test data
[6] training data
[7] votting

است که قرار است فاصله اقلیدسی تمامی نقاط آموزشی از نقطه آزمایشی را درون خود نگهداری کند، که همان مرحله آموزش مدل است که در قبل اشاره شد. در خط ۴ تعداد داده‌های آموزشی را درون متغیری به نام N قرار داده و خط ۵ و ۶ نیز شامل ذخیره کردن فواصل اقلیدسی نقاط آموزشی از نقطه آزمایشی می‌باشد. سپس در خط ۷ آن را به صورت صعودی مرتب کرده و در خط ۸ نیز دسته‌ای که بیشترین حضور را درون k تا عضو اول مجموعه S دارد را نیز انتخاب کرده و خروجی را برمی‌گرداند.



شکل ۱ : قرار گرفتن داده آزمایشی درون داده‌های آموزشی و تخصیص برچسب برای آن.

Algorithm ۱: KNN(D، x^{test}، k)

Data: D is equal to the training data and x^{test} is equal to the test data، respectively. The value of k is also equal to the order of the algorithm.

Result: test data with their labels.

۱ **begin**
۲ $S \longleftarrow \{\}$
۳ $N \longleftarrow |D|$
۴ **for** $i = 1 \rightarrow N$ **do**
۵ $S \longleftarrow S \cup \{\langle i, d(x^{test}, D_i)\rangle\}$
۶ $S \longleftarrow Sort(S)$
۷ $t \longleftarrow \underset{c \in [1,...,T]}{\arg\max} [\sum_{i=1}^{k} S_i^c]$ ▷ s.t. T is number of classes.
۸ **return** $\langle x^{test}, t\rangle$

۲.۲ رگرسیون

استفاده از الگوریتم kNN در مسائل رگرسیون، همانند مسائل دسته‌بندی است با این تفاوت که در مرحله آخر این الگوریتم، به‌جای محاسبه کردن دسته‌ای با بیشترین حضور، از مقدار برچسب تمامی آن k تا عضو، میانگین گرفته و خروجی را برمی‌گرداند. بنابراین باید به‌جای خط ۷ الگوریتم ۱ ، دستور زیر را قرار داد:

$$t \longleftarrow \frac{1}{K} \sum_{i=1}^{K} D_i^t$$

که در اینجا، D_i^t برچسب داده iام و عددی پیوسته است.

این نکته قابل ذکر است که در این الگوریتم، معیار فاصله الزاماً فاصله اقلیدسی نیست و می‌تواند انواع فاصله‌های دیگر مانند فاصله مینکوفسکی [۸] نیز باشد.

۳ پیچیدگی زمانی

اگر بخواهیم جستجوی کامل یا ساده‌ترین راه را برای این الگوریتم درنظر بگیریم، مرتبه زمانی این الگوریتم برابر با $O(n.m)$ است، جایی‌که n تعداد رکوردها یا نمونه‌های داده آموزشی و مقدار m برابر با تعداد ویژگی‌ها می‌باشد. از آنجایی که در بیشتر مواقع تعداد رکوردهای داده آموزشی بسیار بیشتر از ابعاد آن است ($n \gg m$)، بنابراین مرتبه زمانی این الگوریتم را $O(n)$ در نظر می‌گیریم.

۴ مزایا و معایب

همان‌طور که در قبل نیز اشاره شد، یکی از نقاط مثبت الگوریتم kNN، سریع بودن آن است. این امر در برخورد با داده‌های حجیم با ابعاد بسیار بالا دیده می‌شود و این الگوریتم را به یکی از سبک‌ترین الگوریتم‌های موجود در حوزه یادگیری تبدیل کرده است. یکی دیگر از نقاط مثبت این الگوریتم، فهم آسان آن است. اغلب مدل‌ها و الگوریتم‌های حوزه یادگیری که در فصل‌های بعد مطالعه خواهید کرد، همگی دارای پیچیدگی‌های زیادی هستند و به همین دلیل فهم و تفسیر کردن چنین مدل‌هایی کار بسیار سختی است. این درحالی است که الگوریتم kNN بسیار فهم آسان و قابل تفسیری دارد. همچنین یکی از جاهایی که این الگوریتم بسیار استفاده می‌شود، در مسائلی است که فضای ویژگی داده‌های آن از جنس فاصله باشند، مانند داده‌های نقشه‌ای.
اما این الگوریتم سریع، معایبی نیز دارد که باعث شده است در مسائل سخت‌تر و بزرگ‌تر کمتر از آن استفاده شود. احتمالاً این نکته برایتان سؤال شده است که چطور مقدار بهینه k را پیدا کنیم. یکی از چالش‌های الگوریتم kNN، پیدا کردن مقدار مناسب k است که اغلب با بزرگ و پیچیده شدن داده‌ها کار راحتی نیست. یکی دیگر از نقاط ضعف این الگوریتم، استفاده بیش از حد از فضای حافظه در گام آموزش است. اما مهم‌ترین نقطه ضعف این الگوریتم برمی‌گردد به ماهیت و تفسیر فاصله در فضاهای با بُعد بالاتر. درواقع نکته‌ای که قابل توجه است، آن است که فاصله دو نقطه در فضای دو بُعدی برابر با فاصله آن دو نقطه در فضای n بُعدی نخواهد بود [۲]. این امر موجب آن می‌شود که با افزایش بُعد داده‌ها، عملکرد این الگوریتم به شدت کاهش پیدا کند و دیگر نتایج حاصل شده از آن، قابل اتکا نباشد.

مراجع

[1] E. Fix and J. L. Hodges, "Discriminatory analysis. nonparametric discrimination: Consistency properties," *International Statistical Review/Revue Internationale de Statistique*, vol. 57, no. 3, pp. 238–247, 1989.

[2] H. Daumé, *A course in machine learning*. Hal Daumé III, 2017.

[8] Minkowski