دانشگاه صنعتی خواجه نصیرالدین طوسی تهران
مبانی سیستم‌های هوشمند (پاییز ۱۴۰۴)، مهدی علیاری شوره‌دلی

مینی پروژه شماره دو

۱ پرسش یک: سخت‌حاشیه، حاشیهٔ هندسی، دوگان و بردارهای پشتیبان

آ. فرم اولیهٔ SVM سخت‌حاشیه[۱] را بنویسید.

ب. نشان دهید عرضِ حاشیه برابر $\frac{2}{\|w\|}$ است.

ج. دوگان را استخراج کنید و توضیح دهید چرا تنها نمونه‌های با $\alpha_i > 0$ (بردارهای پشتیبان) در تابعِ تصمیم[۲] ظاهر می‌شوند.

۲ پرسش دو: حاشیهٔ نرم با نُرم ℓ_1 و هینج‌لاس

آ. فرم اولیهٔ نرم‌حاشیه[۳] را با متغیرهای شُل‌شدگی[۴] ξ_i بنویسید.

ب. ثابت کنید این فرم معادل کمینه‌سازیِ $\frac{1}{2}\|w\|^2 + C\sum_i \max(0, 1 - y_i f(x_i))$ است.

ج. با استدلال روشن توضیح دهید پارامتر C چگونه بر تعمیم‌پذیری و تعداد بردارهای پشتیبان اثر می‌گذارد.

۳ پرسش سه: نُرم ℓ_2 (شُل‌شدگی مربعی) و تفاوت دوگان با ℓ_1

آ. فرم اولیهٔ soft margin-ℓ_2[۵] را بنویسید.

ب. دوگان متناظر را به‌دست آورید.

ج. تفاوت کران‌های α_i با حالت ℓ_1 و اثر آن بر تعداد و وزن SVها را توضیح دهید.

آ. تعریف «کرنل معتبر»[۶] را بنویسید و ارتباط آن با وجودِ نگاشتِ ویژگی[۷] را توضیح دهید.

ب. با استفاده از خواص بستار[۸]، نشان دهید کرنل‌های RBF و چندجمله‌ای معتبرند.

ج. حالات KKT را به‌صورت مرتب بنویسید و نقش هر حالت را توضیح دهید.

[1] Hard-Margin SVM
[2] Decision Function
[3] Soft-Margin
[4] Slack Variables
[5] ℓ_2-Soft Margin (Quadratic Slack)
[6] Valid/PSD Kernel
[7] Feature Map
[8] Closure Properties

۴ پرسش پنج: محاسبهٔ دستیِ SVM سخت‌حاشیه در $\mathbb{R}^2$ با نمونه‌های بیشتر

در $\mathbb{R}^2$ کلاس‌ها به‌صورت زیر داده شده‌اند. کلاس مثبت:

$$(2,0),\ (3,0),\ (2,1),\ (2,-1)$$

و کلاس منفی:

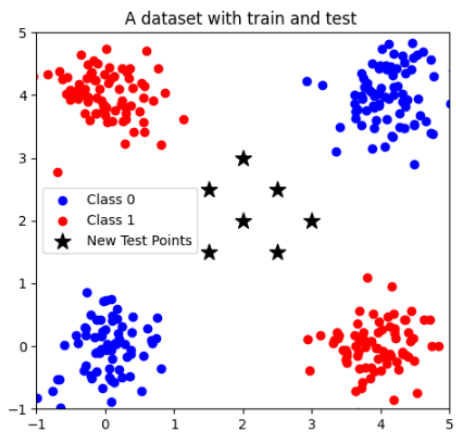
$$(-2,0),\ (-3,0),\ (-2,-1),\ (-2,1).$$

آ. بر مبنای هندسهٔ داده‌ها، ابرصفحهٔ حداکثرحاشیه را حدس بزنید و معادلهٔ مرز تصمیم را بنویسید.

ب. ضرایب $w \in \mathbb{R}^2$ و $b \in \mathbb{R}$ را به‌صورت دستی محاسبه کنید.

ج. بردارهای پشتیبان را مشخص کنید و مقدار حاشیهٔ هندسی و نیز فاصلهٔ دو ابرصفحهٔ $w^\top x + b = \pm 1$ را دقیق محاسبه نمایید.

۵ پرسش شش : طبقه بندی دسته بندی غیرخطی



دیتاست دوکلاسه‌ی شکل ۵ را در نظر بگیرید که در آن داده‌های آموزشی اولیه متعلق به دو کلاس متمایز بوده و چندین نقطه‌ی جدید (به‌صورت ستاره‌های سیاه) در ناحیه‌ی مرکزی به آن افزوده شده‌اند. هدف، طراحی و پیشنهاد یک مدل مناسب برای طبقه‌بندی صحیح این داده‌ها است.

آ) اگر بخواهیم با استفاده از شبکه‌های عصبی این مجموعه‌داده را مدل کنیم، کدام نوع شبکه عصبی را پیشنهاد می‌دهید؟ (برای مثال: MLP، RBF، Sigmoid، و غیره)

ب) به‌صورت تحلیلی توضیح دهید که ضعف سایر مدل‌های شبکه‌های عصبیِ آموزش‌داده‌شده تا کنون (مانند MLP یا شبکه‌های با توابع فعال‌سازی سراسری) در ایجاد مرز تصمیم در این شکل چیست؟ چرا برخی مدل‌ها در تفکیک خوشه‌های محلی این داده‌ها عملکرد ضعیف‌تری دارند؟

۶ سوال پیاده سازی RBF

هدف این سوال، مدل‌سازی یک سیستم دینامیکی ساده با استفاده از یک شبکه عصبی با تابع شعاعی پایه[۹] است. شما با پیاده‌سازی یک RBFNN استاندارد با ساختاری ایستا[۱۰] شروع خواهید کرد (بخش اول) و سپس یک شبکه پیشرفته‌تر وتطبیقی[۱۱] (بخش دوم) را توسعه خواهیدداد که ساختار خود را بر اساس شبکه M-RAN که در مقاله ارائه‌شده بحث شده‌است، بهینه‌سازی می‌کند.

[۹] Network Neural RBF
[۱۰] Static
[۱۱] adaptive

قسمت اول:

۱.۱ بخش اول: شناسایی سیستم با شبکه‌های عصبی RBF

برای این سوال ابتدا یک سیستم ساده‌شده از نوع ball and beam را مدل‌سازی خواهیم‌کرد. موقعیت توپ در زمان t، که با $y(t)$ نشان داده می‌شود، تابعی از دو موقعیت قبلی خود یعنی $y(t-1)$ و $y(t-2)$ و ورودی قبلی به سیستم یعنی $u(t-1)$ است. این رابطه توسط معادله تفاضلی[۱۲] زیر تعریف می‌شود:

$$y(t) = \frac{y(t-1)}{1+y(t-2)} + u(t-1)^3$$

در ادامه شما باید یک شبکه RBFNN ایستا بسازید که به عنوان "شناساگر"[۱۳] عمل می‌کند و خروجی سیستم ($y(t)$) را بر اساس بردار ورودی ($x(t) = [y(t-1), y(t-2), u(t-1)]^T$) پیش‌بینی می‌کند.

RBFNN ایستا در این بخش، شما یک RBFNN خواهید ساخت که در آن تعداد و ویژگی‌های هایneuron مخفی قبل از آموزش تعیین می‌شوند. سپس از روش "حداقل مربعات خطی"[۱۴] برای آموزش وزن‌های خروجی استفاده خواهید کرد.

ایجاد دیتاست ball: and beam

۱.۱.۱

سیگنال $u(t)$ یک موج سینوسی ساده خواهد بود تا دینامیک سیستم را تحریک کند. کدی برای تولید ۱۰۰۰ sample (جفت‌های ورودی خروجیX /Y) به عنوان داده های دیتاست بنویسید. برای این کار از سیگنال $u(t) = \sin(2\pi t/250)$ استفاده کنید. $y(0) = y(1) = 0$ را به عنوان مقادیر اولیه در نظر بگیرید. از $t = 2$ تا $t = 999$ تکرار کنید تا دنباله $y(t)$ تولید شود. مجموعه داده نهایی شما شامل بردارهای ورودی $x(t) = [y(t-1), y(t-2), u(t-1)]^T$ و target متناظر آن‌ها یعنی $y(t)$ خواهدبود.

۲.۱.۱

داده ها را به دو قسمت برای آموزش و تست تقسیم کنید. برای این کار از ۷۰۰ نقطه اول برای آموزش و ۳۰۰ نقطه باقی‌مانده برای آزمایش (تست) استفاده کنید.

۳.۱.۱

توضیح دهید چرا این مسئله یک "مسئله ی تقریب تابع"[۱۵] است. تابع مجهولی که RBFNN در تلاش برای یادگیری آن خواهد بود، چیست؟

۲.۱ بخش دوم: پیاده سازی RBFNN ایستا

یک RBFNN معمولاً دارای یک لایه ورودی، یک لایه مخفی با نورون های RBF، و یک لایه خروجی خطی است. خروجی شبکه به صورت زیر محاسبه می‌شود (برای سادگی α_0 را صفر در نظر گرفته‌ایم):

$$f(x) = \sum_{k=1}^{K} \alpha_k \phi_k(x)$$

---

[۱۲] equation difference
[۱۳] identifier
[۱۴] Squares Least Linear
[۱۵] problem approximation function

که در آن K تعداد نورون‌های مخفی، α_k وزن‌های خروجی و $\phi_k(x)$ تابع فعال‌سازی RBF گوسی[۱۶] (بر اساس مقاله) برای نورون k ام است. همچنین $\phi_k(x)$ به صورت زیر تعریف میشود: (علامت $\|\|$ نشان‌دهنده‌ی نُرم اقلیدسی است)

$$\phi_k(x) = \exp\left(-\frac{1}{\sigma_k^2}\|x-\mu_k\|^2\right)$$

۱.۲.۱

نقش مرکز نورون (μ_k) و عرض (σ_k) آن را توضیح دهید. کم و زیاد شدن هر پارامتر چگونه بر پاسخ نورون به یک ورودی تأثیر می‌گذارد؟

۲.۲.۱

تابعی برای یک نورون تعریف کنید که فعال‌ساز RBF گوسی را با گرفتن بردار ورودی x، مرکز μ_k و عرض σ_k محاسبه کند. (این تابع را با ورودی و مرکز یکسان $[1,1]$ و عرض ۱ تست کنید) خروجی فعال ساز چه مقدار را نشان میدهد؟ توضیح دهید که این خروجی به چه معناست؟

۳.۱ بخش سوم: آموزش از طریق حداقل مربعات خطی (LLS)

با ثابت بودن مراکز و عرض‌ها، خروجی RBFNN یک ترکیب خطی از فعال‌سازهای RBF است. این بدان معناست که شما می‌توانید برای محاسبه وزن‌های خروجی بهینه ($\alpha = [\alpha_1, \cdots, \alpha_k]^T$) از LLS استفاده کنید. بواقع شما باید دستگاه معادلات خطی $\Phi\alpha = Y$ را حل کنید، که در آن Y بردار target و Φ ماتریس ای به صورت $\Phi_{ij} = \phi_j(x_i)$ است.

۱.۳.۱

با نوشتن کدی، ماتریس Φ را برای داده های آموزش خود بسازید. شبکه rbf ایستا را بوسیله LLS آموزش (روی مجموعه آموزش) دهید. تعداد مراکز انتخاب شده و مقدار عرض انتخاب شده را گزارش کنید. بردار وزنهای بهینه خروجی α، و خروجی ها را برای مجموعه ی آموزش بدست آورید. مدل را روی مجموعه تست آزمایش کنید و خروجی های پیش بینی شده را بدست آورید.

توجه کنید که:

- K مرکز (μ_k) را با انتخاب تصادفی K داده از مجموعه آموزشی خود انتخاب کنید.
- تمام عرض‌های (σ_k) را روی یک مقدار ثابت تنظیم کنید. یک نقطه شروع خوب میتواند میانگین فاصله بین مراکز انتخاب شده‌باشد.

همچنین تابعی برای ارزیابی شبکه آموزش‌دیده خود بنویسید. از ریشه میانگین مربعات خطا[۱۷] به عنوان معیار ارزیابی عملکرد شبکه خود استفاده کنید. مقدار RMSE را برای هر دو مجموعه تست و آموزش گزارش دهید.

۲.۳.۱

خروجی های پیش‌بینی شده در مجموعه آموزش را با خروجی های واقعی در یک نمودار بر اساس اندیس نمونه ها نمایش دهید (محور x اندیس نمونه ها و محور y مقادیر خروجی).

۳.۳.۱

نمودار مشابه قسمت قبلی برای مجموعه تست ایجاد کنید. این نمودار را تحلیل کنید. بیشترین خطا در کدام قسمت رخ داده است؟ چرا؟ تحلیل کنید.

---

[۱۶] Gaussian

[۱۷] Error Squared Mean Root

۴.۳.۱

توضیح دهید چرا این فرآیند آموزش بوسیله ی LLS بسیار سریع‌تر از backpropagation" مبتنی بر گرادیان کاهشی" است که در پرسپترون‌های چندلایه[۱۸] استفاده می‌شود.

۵.۳.۱

همچنان σ را ثابت در نظر بگیرید این بار شبکه‌هایی برای مقادیر مختلف $K = 10, 20, 50, 100, 200$ آموزش دهید. مدل ها را روی داده های تست ارزیابی کنید سپس RMSE تست را به صورت تابعی از K رسم کنید. بهترین مقدار k را گزارش کنید.

۶.۳.۱

توصیف کنید که اگر K خیلی کوچک یا خیلی بزرگ باشد چه اتفاقی می‌افتد. این موضوع را به "کم‌برازش"[۱۹] و "بیش‌برازش"[۲۰] ربط دهید.

۷.۳.۱

K را بر روی بهترین مقداری که پیدا کردید ثابت کنید و اکنون شبکه را با مقادیر مختلف σ آموزش داده و روی داده های تست ارزیابی کنید ($\sigma = 0.5, 1, 5, 7, 10$).

همچنین RMSE تست را به صورت تابعی از σ رسم کنید. بهترین مقدار σ را گزارش‌کنید. همچنین خروجی‌های بدست آمده از شبکه با بهترین σ (و بهترین k) روی مجموعه تست ارزیابی کرده و مقادیر پیش‌بینی شده را با مقادیر خروجی واقعی مقایسه‌کنید و در نموداری مانند نمودار ۳.۳.۱ نشان دهید.

۸.۳.۱

توصیف کنید که وقتی σ خیلی کوچک یا خیلی بزرگ است چه اتفاقی می‌افتد؟ این موضوع چگونه بر توانایی "تعمیم پذیری"[۲۱] شبکه تأثیر می‌گذارد؟

۹.۳.۱

در انتخاب "بهترین" ساختار شبکه (شبکه با بهترین k و σ) با چه چالش‌هایی روبرو شدید؟ توضیح دهید.

۴.۱ بخش چهارم: تلاش برای حل چالش ها

۱.۴.۱

در قسمتهای قبلی شما از روش استاندارد کمترین مربعات خطی (LLS) برای یافتن وزن‌های بهینه استفاده کرده‌اید. زمانی که K بسیار بزرگ باشد میتواند سبب مشکل در تعمیم پذیری شود. یکی از روش های حل این موضوع آموزش با LLS همراه با Regularization L۲ است. تابعی برای آموزش مدل RBFNN Static با استفاده از Regularization L۲ با انتخاب K مرکز (μ_k) برای یک مقدار بزرگ K و عرض‌های (σ) ثابت تعریف‌کنید و بردار وزن های بهینه و خروجی ها را بدست آورید. مدل آموزش دیده را روی مجموعه تست ارزیابی کنید و خروجی های پیش بینی شده را با خروجی های واقعی مقایسه کنید (میتوانید ضریب رگولاریزیشن را یکی از اعداد$0.001, 0.01, 0.1, 1, 10$قرار دهید). تفاوت خروجی حاصل با و بدون Regularization برای مقدار σ و k انتخابی مقایسه‌کنید.

---

[۱۸] MLPs
[۱۹] underfitting
[۲۰] overfitting
[۲۱] generalize

۲.۴.۱

بردار وزن‌های بهینه α زمانی که از تابع Regularization L2 استفاده می‌کنیم از رابطه ی زیر بدست می‌آید که در آن λ ضریب رگولاریزیشن است:

$$(\Phi^T\Phi + \lambda I)\alpha = \Phi^T Y$$

نقش پارمتر λ را توضیح دهید؟ با افزایش یا کاهش زیاد این پارامتر چه اتفاقی می‌افتد؟

۵.۱ بخش پنجم

۱.۵.۱

بار دیگر مقدار σ را ثابت در نظر بگیرید و قسمت ۱.۳.۱ را با یافتن k مرکز (μ_k) با استفاده از clustering K-Means[۲۲] بجای انتخاب رندوم حل کنید و وزن های بهینه را بدست آورید. مقدار RMSE تست را گزارش دهید.

همچنین مجددا خروجی های پیش بینی شده توسط مدل روی داده های تست و خروجی های واقعی را در نموداری نمایش داده و مقایسه کنید (توجه کنید که k حتما بزرگتر از ۱۰ انتخاب شود) نتایج را با زمانی که از K مرکز رندوم استفاده می‌کردید مقایسه کنید.

۲.۵.۱

حال با استفاده از مقادیر مراکز بدست آمده از سوال قبل و استفاده از استراتژی Nearest-Neighbor[۲۳] تابعی بنویسید که عرض σ برای هر مرکز محاسبه کند. برای به کار بردن این روش عرض σ هر نورون را بر اساس میانگین فاصله تا p نزدیکترین مراکز بدست آورد یعنی برای بدست آوردن عرض σ هر نورون، فاصله اقلیدسی مرکز آن نورون تا تمام مراکز (بجز خودش) را بدست آورید. فواصل به دست آمده را به صورت صعودی مرتب کنید و میانگین فاصله تا p تا از نزدیکترین مراکز را بدست آورید (میانگین p فاصله ی کوچکتر)؛ این مقدار عرض آن نورون می باشد. با استفاده از این مراکز بار دگر وزن های بهینه را بدست آورده و گزارش کنید ($p = 2$ قرار دهید).

مقادیر RMSE روی داده‌های تست و آموزش را گزارش کنید. خروجی های پیش بینی شده روی داده‌های تست و خروجی های واقعی را در نموداری نمایش داده و مقایسه کنید. نتایج این قسمت را با نتایج قسمت قبل (زمانی که از σ ثابت استفاده کردید) مقایسه کنید.

۳.۵.۱

توضیح دهید چرا clustering K-Means از نظر تئوری در مقایسه با انتخاب تصادفی، روش بهتری برای انتخاب مراکز RBF است؟

قسمت دوم: پیاده‌سازی RBFNN تطبیقی با الهام از M-RAN

قسمت اول سوال نقطه ضعف کلیدی هایRBFNN ایستا را نشان‌داد؛ یافتن توپولوژی[۲۴] مناسب چنین شبکه‌ای دشوار و ناکارآمد است.

در مقاله ارائه‌شده راه حلی برای این موضوع پیشنهاد شده‌است؛ شبکه‌ای که بر اساس نیاز می‌تواند نورون‌ها و لایه‌های پنهان را افزایش‌دهد و یا هرس‌کند در این بخش نسخه‌ی ساده شده‌ای از M-RAN پیشنهاد شده در مقاله را پیاده‌سازی می‌کنید.

در این روش برای آموزش مدل به‌جای استفاده همزمان از کل مجموعه آموزشی، اکنون داده‌ها به صورت ترتیبی، یعنی هر بار یک داده برای پردازش به مدل داده‌خواهدشد. شبکه با صفر نورون پنهان شروع می‌شود و فقط در صورت نیاز نورون‌ها اضافه

---

[۲۲] clustering K-Means
[۲۳] Nearest-Neighbor
[۲۴] topology

می‌شوند. طبق مقاله، یک unit hidden جدید برای یک نمونه (x_n, y_n) اضافه می‌شود، اگر داده جدید[۲۵] باشد و خطا قابل توجه[۲۶] باشد.

پس معیارهای رشد شبکه شما عبارتند از:

۱. تازگی[۲۷]: ورودی x_n از مراکز موجود دور است:

$$\|x_n - \mu_n\| > \varepsilon$$

که μ_n نزدیکترین مرکز به x_n است.

۲. خطا: خطای شبکه بزرگ باشد؛ یعنی:

$$|y_n - f(x_n)| > e_{\min}$$

۱.۲ بخش اول: یادگیری ترتیبی[۲۸] و معیار رشد

۱.۱.۲

ابتدا تابعی بنویسید که یک نمونه جدید (x_n, y_n)، مقدار پیش‌بینی شده، و آستانه ε و $e_{\min}$ را دریافت می‌کند. اگر نیاز به اضافه‌کردن نورون وجود داشته‌باشد True و در غیر این صورت False را برمیگرداند. (میتوانید با مقادیر $\varepsilon = 1$ و $e_{\min} = 0.5$ و چند مقدار فرضی برای ورودی و خروجی واقعی و مقدار پیش بینی شده تابع را تست کنید)

۲.۱.۲

چرا هر دو معیار برای رشد شبکه ضروری هستند؟ چه اتفاقی می‌افتد اگر فقط از معیار خطا استفاده می‌کردیم؟ چه اتفاقی می‌افتد اگر فقط از معیار تازگی استفاده می‌کردیم؟ افزایش و کاهش ε چه تاثیری در شبکه دارد؟

۲.۲ بخش دوم(امتیازی): پیاده سازی RBFNN تطبیقی با استراتژی هرس[۲۹]

در این قسمت شما باید شبکه RBFNN تطبیقی بر اساس M-RAN گفته‌شده در مقاله را به صورت کامل پیاده‌سازی کنید برای این منظور شما در هر epoch از فرایند آموزش مدل باید تنها یک sample به شبکه بدهید همچنین شما از استراتژی هرس گفته شده در مقاله استفاده می‌کنید که در قسمت زیر توضیح داده شده است.

هرس: برخی از نورون هایی که اضافه می‌شوند ممکن است بعدا نقش کمی در تولید خروجی داشته‌باشند و به واقع redundant طلقی شوند. برای حفظ سادگی و عدم پیچیدگی شبکه ایجادشده الگوریتم M-RAN واحدهای پنهان[۳۰] را که سهم کمی در خروجی شبکه دارند هرس می‌کند. مقاله یک استراتژی قوی برای هرس‌کردن پیشنهاد می‌کند (بخش ۲.۲). این استراتژی هرس، با حذف نورون‌هایی که در یک دوره زمانی تاثیر کمی روی خروجی دارند سبب فشردگی مدل می‌شود.

استراتژی هرس:

- محاسبه مشارکت: برای هر نورون k در نمونه n، مشارکت خروجی آن را پیدا کنید:

$$o_k^n = \alpha_k \times \phi_k(x_n)$$

---

[۲۵] novel
[۲۶] significant
[۲۷] Novelty
[۲۸] Sequential
[۲۹] Pruning
[۳۰] hidden

- نرمال‌سازی مشارکت: حداکثر خروجی مطلق را بیابید: $o^n_{\max} = \max_k(|o^n_k|)$ و خروجی نرمال‌شده را برای هر نورون محاسبه کنید:

$$r^n_k = \frac{|o^n_k|}{o^n_{\max}}$$

- بررسی اهمیت: یک نورون در نمونه n بی‌اهمیت[۳۱] است اگر: $r^n_k < \delta$
- تصمیم هرس: نورون k را حذف کنید اگر برای M مشاهده متوالی «بی‌اهمیت» بوده‌باشد.

برای پیاده‌سازی هرس در کد خود، باید یک شمارنده پایدار[۳۲] برای هر نورون استفاده کنید (مثلاً، در آرایه‌ای به نام prun-ing_counters):

برای پیاده‌سازی در کد خود مراحل زیر را برای نمونه فعلی x_n انجام دهد:

۱. مشارکت خروجی o^n_k را برای همه K نورون محاسبه‌کنید.
۲. خروجی‌های نرمال‌شده r^n_k را برای همه K نورون محاسبه‌کنید.
۳. در یک حلقه، هر نورون k را بررسی‌کنید:
   - اگر $r^n_k < \delta$ (بی‌اهمیت است) pruning_counters[k] را افزایش‌دهید[۳۳]
   - اگر $r^n_k > \delta$ (بااهمیت است) pruning_counters[k] را برابر ۰ قراردهید[۳۴]
۴. تمام اندیس‌های k را که در آن‌ها pruning_counters[k] $> M$ است، پیداکنید و نورون‌ها و آرایه های شمارنده‌ی مربوط به آن‌ها را حذف کنید.
۵. وضعیت شبکه را به‌روزرسانی کنید.

۱.۲.۲

اکنون، منطق رشد (از بخش اول) و منطق هرس پیشرفته (از بخش ۱.۲) را در یک حلقه آموزش واحد ادغام و به صورت کامل پیاده‌سازی کنید و مدل را آموزش دهید.

فرایند آموزش برای هر نمونه (x_n, y_n) در هر epoch باید به صورت زیر انجام شود:

- خروجی شبکه $f(x_n)$ (یا همان $\hat{y}$) و خطا ($e_n = y_n - f(x_n)$) را محاسبه‌کنید.
- معیارهای رشد را بررسی‌کنید.
- در صورت تایید و نیاز نورون جدید را به شبکه اضافه‌کنید.
- اگر نورون جدید اضافه‌شد:
  - مرکز آن را $\mu_{k+1} = x_n$ قراردهید
  - وزن آن را $\alpha_{k+1} = e_n$ تنظیم‌کنید
  - عرض آن را با رابطه: $\sigma_{k+1} = \kappa\|x_n - \mu_n\|$ تنظیم کنید (κ ضریب همپوشانی است و میتوانید آن را در ابتدا ۸.۰ انتخاب کنید)
  - یک عدد ۰ به آرایه pruning_counters اضافه کنید (برای نورون جدید).
- اگر نورون جدیدی به شبکه اضافه نشد:
  - تمام وزن‌های خروجی موجود α_k را با استفاده از قانون "حداقل میانگین مربعات"[۳۵] به‌روزرسانی‌کنید (این قانون به صورت: $\alpha_k(n) = \alpha_k(n-1) + \eta \cdot e_n \cdot \phi_k(x_n)$ می‌باشد که در آن η نرخ یادگیری[۳۶] است.)

[۳۱] insignificant
[۳۲] persistent
[۳۳] increment
[۳۴] reset
[۳۵] LMS
[۳۶] rate learning

- پس از رشد یا به‌روزرسانی هرس را انجام‌دهید.

نکات مهم زیر را برای انجام این قسمت در نظر داشته‌باشید:

توجه‌کنید که برای انجام آموزش شما باید مقادیر مناسبی را برای هایپرپارامترها[۳۷] انتخاب‌کنید ($\kappa, M, \eta, \delta, e_{\min}, \varepsilon$).

می‌توانید برای این کار از روش هایی مثل search grid استفاده‌کنید. با قرار دادن $M = 60$، $\kappa = 0.8$، $\delta = 0.1$ و $\eta = 0.05$ با استفاده از روش search grid مقادیر مناسبی برای $e_{\min}$ و ε روی بازه‌ی دلخواه انتخابی برای ε و $e_{\min} \in \{0.05, 0.1, 0.15, 0.2\}$ بدست‌آورید.

(همچنین می‌توانید شبکه را به صورت تطبیقی نیز پیاده‌سازی کنید یعنی همزمان که وزن‌ها را آموزش می‌دهید، هایپرپارامترها را نیز آموزش‌داده و بهترین هایپرپارامترها را بدست‌آورید)

در پیاده‌سازی شبکه خود کد خود را مجهزکنید تا ذخیره‌ای از تعداد نورون‌های پنهان در هر مرحله از فرآیند آموزش را نگه‌دارد.

۲.۲.۲

در نهایت RBFNN تطبیقی خود را بر روی مجموعه آموزشی، آموزش دهید. سپس، عملکرد نهایی آن را بر روی مجموعه آزمایش ارزیابی‌کنید. نتایج و RMSE آن را گزارش‌دهید. خروجی های پیش‌بینی شده را با خروجی واقعی در یک نمودار نمایش داده و تحلیل‌کنید. همچنین بهترین هایپرپارامتر های پیدا شده را گزارش‌کنید.

۳.۲.۲

نموداری ایجادکنید که نشان‌دهد چگونه تعداد نورون های پنهان در شبکه تطبیقی شما در طول دوره آموزش تغییرکرده‌است. (مانند شکل A۲ یا A۳ در مقاله)

۴.۲.۲

تعداد نهایی نورون ها و RMSE نهایی روی داده‌های تست را برای بهترین RBFNN ایستا و RBFNN تطبیقی خود مقایسه‌کنید.

۵.۲.۲

بر اساس نتایج خود، در مورد مزایا و معایب رویکرد تطبیقی M-RAN در مقایسه با RBFNN ایستا بحث کنید. آیا شبکه تطبیقی موفق شد توپولوژی شبکه حداقلی[۳۸] و در عین حال با دقت بالا را پیدا کند؟ چرا این امر برای کاربردهای دنیای واقعی[۳۹] مهم است؟

۷ کدنویسی - SVM خطی و مطالعهٔ پارامتر C

داده‌ها: Breast Cancer Wisconsin (Diagnostic).
اصل تکرارپذیری: در همهٔ تقسیم‌بندی‌ها و سازنده‌ها، random_state را برابر دو رقم آخر شمارهٔ دانشجویی تنظیم کنید.

آ. آماده‌سازی. داده را بارگذاری کنید؛ به‌صورت لایه‌مند به train، validation و test (پیشنهادی ۶۰/۲۰/۲۰) تقسیم کنید. تنها روی ویژگی‌ها استانداردسازی انجام دهید. یک PCA دوبُعدی صرفاً برای نمایش آماده کنید (مدل‌ها در فضای اصلی آموزش ببینند).

ب. آموزش و گزارش حاشیه. یک SVC(kernel='linear') آموزش دهید. با رابطهٔ $w = \sum_i \alpha_i y_i x_i$ (پس از استانداردسازی)، مقدار $1/\|w\|$ را محاسبه و گزارش کنید. روی validation و test شاخص‌های Accuracy، Precision، Recall، F1 و ROC-AUC را اندازه‌گیری و گزارش کنید.

---

[۳۷] hyperparameters
[۳۸] minimal
[۳۹] applications real-world

ج. **مطالعهٔ C.** برای $C \in \{0.01, 0.1, 1, 10, 100\}$ مدل را بازآموزی کنید. برای هر C، شاخص‌های فوق و تعداد **support vectors** را استخراج کرده و در یک جدول منظم ارائه دهید. تغییرات بایاس/واریانس را تحلیل مکتوب کنید.

د. **نمودارهای ارزیابی.** برای بهترین C (انتخاب بر مبنای validation و تأیید روی test)، منحنی‌های ROC و Precision–Recall را رسم و تفسیر کنید.

در انجام این تمرین حتماً به نکات زیر توجه کنید:

- موعد تحویل این تمرین، ساعت ۱۸:۰۰ روز ۳۰ آبان‌ماه ۱۴۰۴ است.
- برای گزارش لازم است که پاسخ هر سؤال و زیربخش‌هایش به‌ترتیب و به‌صورت مشخص نوشته شده باشند. بخش زیادی از نمره به توضیحات دقیق و تحلیل‌های کافی شما روی نتایج بستگی خواهد داشت.
- لازم است که در صفحهٔ اول گزارشِ خود لینک مخزن گیت‌هاب و گوگل‌کولبِ مربوط به مینی‌پروژهٔ خود را درج کنید. درخصوص گیت‌هاب، یک مخزنِ خصوصی درست کنید و آی‌دی‌های `MJAHMADEE`، `ParisaGhorbani`، `AliBagheriNejad` و `erfanY2AJ` را به‌عنوان Collaborator به مخزن اضافه کنید. پروژه‌های گیت‌هاب می‌بایست در انتهای ترم پابلیک شوند. در مقابل، لینک گوگل‌کولب را در حالتی که دسترسی عمومی دارد به اشتراک بگذارید. دفترچهٔ گوگل‌کولب باید به‌صورت منظم و با بخش‌بندیِ مشخص تنظیم شده باشد و خروجیِ سلول‌های اجراشده قابل مشاهده باشد. در گیت‌هاب نیز یک مخزن برای درس و یک پوشهٔ مجزا برای هر مینی‌پروژه ایجاد کنید.
  (آموزش پرایوت‌کردن مخزن گیتهاب و آموزش افزودن Collaborator به مخزن گیت‌هاب)
- چرا از دفترچهٔ گوگل‌کولب؟ شما نباید به فراخوانیِ فایلی خارج از محیط نیاز داشته باشید؛ مطابق آموزش‌های ارائه‌شده ملزم هستید از دستور gdown استفاده کنید و مسیرهای فایل‌ها را طوری تنظیم کنید که صرفاً با اجرای سلول‌ها، امکان فراخوانی و خواندن فایل‌ها توسط هر کاربری وجود داشته باشد.
- در تمامی مراحلِ تعریف داده و مدل و هر جای دیگری که مطابق آموزش‌های ویدیویی و به لحاظ منطقی نیاز است، Random State را برابر با دو رقمِ آخرِ شمارهٔ دانشجوییِ خود در نظر بگیرید.
- استفاده از ابزارهای هوشمند (مانند ChatGPT) در کمک‌گرفتن برای بهبود کدها مجاز است؛ امّا لازم است تمام جزئیات مواردی که در خروجی‌های مختلفِ گزارش خود عنوان می‌کنید را به‌خوبی خوانده، درک و تحلیل کرده باشید. استفاده از این ابزارهای هوشمند در نوشتنِ گزارش و تحلیل‌ها ممنوع است.
- در جاهایی که با توجه به دو رقمِ آخر شمارهٔ دانشجوییِ خود محدود به انتخاب عددی معین یا داده‌ای خاص شده‌اید، برای تست‌های اضافه‌تر و نمایشِ بهبود در نتایجِ خود، مجاز هستید از مقادیر دیگر هم استفاده کنید.
- رعایت نکات بالا به حرفه‌ای‌تر شدن شما کمک خواهد کرد و اهمیتی معادلِ مطالبِ درسیِ فراگرفته‌شده دارد؛ بنابراین، در صورت عدم رعایت یک یا چند مورد از این نکات، از نمرهٔ تمرین شما کاسته خواهد شد.
- آی‌دیِ پرسشِ هرگونه سؤال درخصوص مینی‌پروژهٔ شمارهٔ ۲