به نام خدا

عنوان:

تکلیف شماره پنج درس یادگیری ماشین

استاد:

دکتر پدرام

دانشجو:

محمدعلی مجتهدسلیمانی

شماره دانشجویی:

4033904504

تاریخ:

۱۴۰۳/۷/۳۰

پاسخ ۱.

یک شبکه عصبی دارای ساختار مشخصی هست که شامل یک سری node به نام نورون ها میشود که از سلول های عصبی الهام گرفته شده است که این node ها در لایه ها قرار دارند. node های ما در واقع یک سری گراف هستند که روی یال های آنها یک سری weight یا وزن وجود دارد و در لایه خروجی، یک تابع activation به خروجی اعمال میشود. شبکه های عصبی برای اینکه بتوانند یاد بگیرند وزن ها و بایاس را به نوعی تنظیم میکنند تا مقدار تخمین آنها به مقدار واقعی که desired output یا target value نام دارد به حداقل برسد به این روند یادگیری شبکه های عصبی گفته میشود که یکی از الگوریتم های معروف الگوریتم back propagation در این راستا هست که سعی میکند با تغییر گرادیان مقدار خطا را کم کند.

پاسخ ۲.

مفهوم تعمیم به این اصل اشاره دارد که شبکه عصبی ما نسبت به داده ای که در زمان آموزش یا تشکیل شبکه ندیده است بتواند به درستی پاسخ بدهد و به خروجی مناسب ورودی جدید را نگاشت کند حتی نسبت به داده ای که مقداری با داده آموزشی خود فرق داشته. اگر شبکه عصبی ما overtrained یا overfit شده باشد قابلیت تعمیم نسبت به داده های جدید را از دست میدهد مثلا نسبت به یک noise از داده های آموزشی یک ویژگی در نظر بگیرد که در تابع واقعی ما این ویژگی درست نباشد. قاعده تعمیم تحت تاثیر ۳ عامل هست: ۱. اندازه مجموعه آموزشی ۲. معماری شبکه عصبی ۳. پیچیدگی مسئله.

پاسخ ۳.

این تکنیک در الگوریتم های بهینه سازی به عنوان مثال در SGD در آموزش شبکه های عصبی به کار میرود که منجر به تسریع همگرایی میشود. این کار با در نظر گرفتن گرادیان های قبلی برای هموار شدن بروز رسانی صورت میگیرد و به فرآیند بهینه سازی کمک میکند.

مومنتوم با ضریب بالا باعث همگرایی سریعتر شود و از local minimum ها عبور کند اما در طرف مقابل ممکن است باعث overshooting شود یعنی نوسان گرادیان ما بسیار زیاد شود. اگر ضریب آن کم باشد بروز رسانی جدید فقط به گرادیان فعلی بستگی پیدا میکند و تاثیر گرادیان های قبلی محو یا کمتر میشود که باعث

طولانی شدن فرآیند یادگیری میشود و ممکن هست در local minimum ها گیر کند. مومنتوم حتی اگر نرخ یادگیری پایین باشد اگر به درستی تنظیم شود شاید بتواند از local minimum فرار کند.

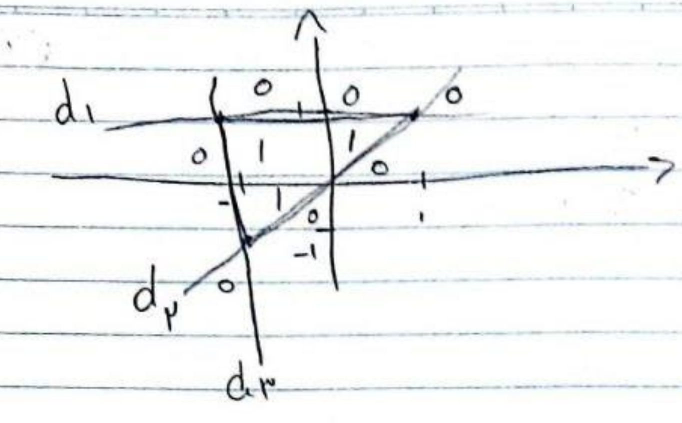
پاسخ ٤.

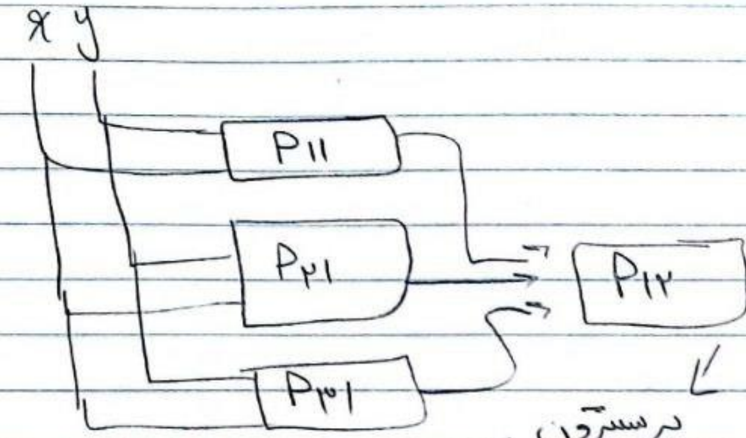
نرخ یادگیری در واقع به ما کمک میکند تا اندازه گام بعدی را (منظور همان تنظیم کردن وزن ها است) در جهت کاهش تابع loss یا رسیدن به نقطه بهینه را بدست بیاوریم. در واقع مشخص میکند که پارامتر هایی مثل وزن چه قدر سریعتر و بزرگتر باید بروز رسانی شوند تا به نقطه بهینه و همگرایی برسیم. نرخ یادگیری اگر بالا باشد ممکن است سریعتر به همگرایی برسیم ولی ممکن است باعث overshooting شود و از روی نقطه بهینه عبور کنیم. اگر نرخ یادگیری کم باشد سرعت همگرایی هم کمتر میشود و روند یادگیری طولانی شود (به این منظور که زمان بیشتری طول میکشد تا به نقطه بهینه برسیم) همچنین باعث میشود که شبکه سخت تر از local minimum عبور کند و از طرف دیگر تعداد iteration ها هم بالاتر برود به همین دلیل.

پاسخ ٥.

برای این سوال ما به ۳ پرسپترون نیاز داریم که برای طراحی این ۳ پرسپترون نیاز به معادله خط داریم و به نوعی طراحی میشوند که اگر نقطه داخل خط بود ۱ و اگر نبود ۰ در نظر گرفته شود و در لایه آخر یک AND قرار میدهیم تا مطمئن شویم از هر ۳ پرسپترون خروجی یک حاصل میشود. ورودی ما مختصات هستند و وزن ها باید به سمت داخل مثلث محاسبه شوند. پس در نهایت به ٤ پرسپترون نیاز داریم.

۵

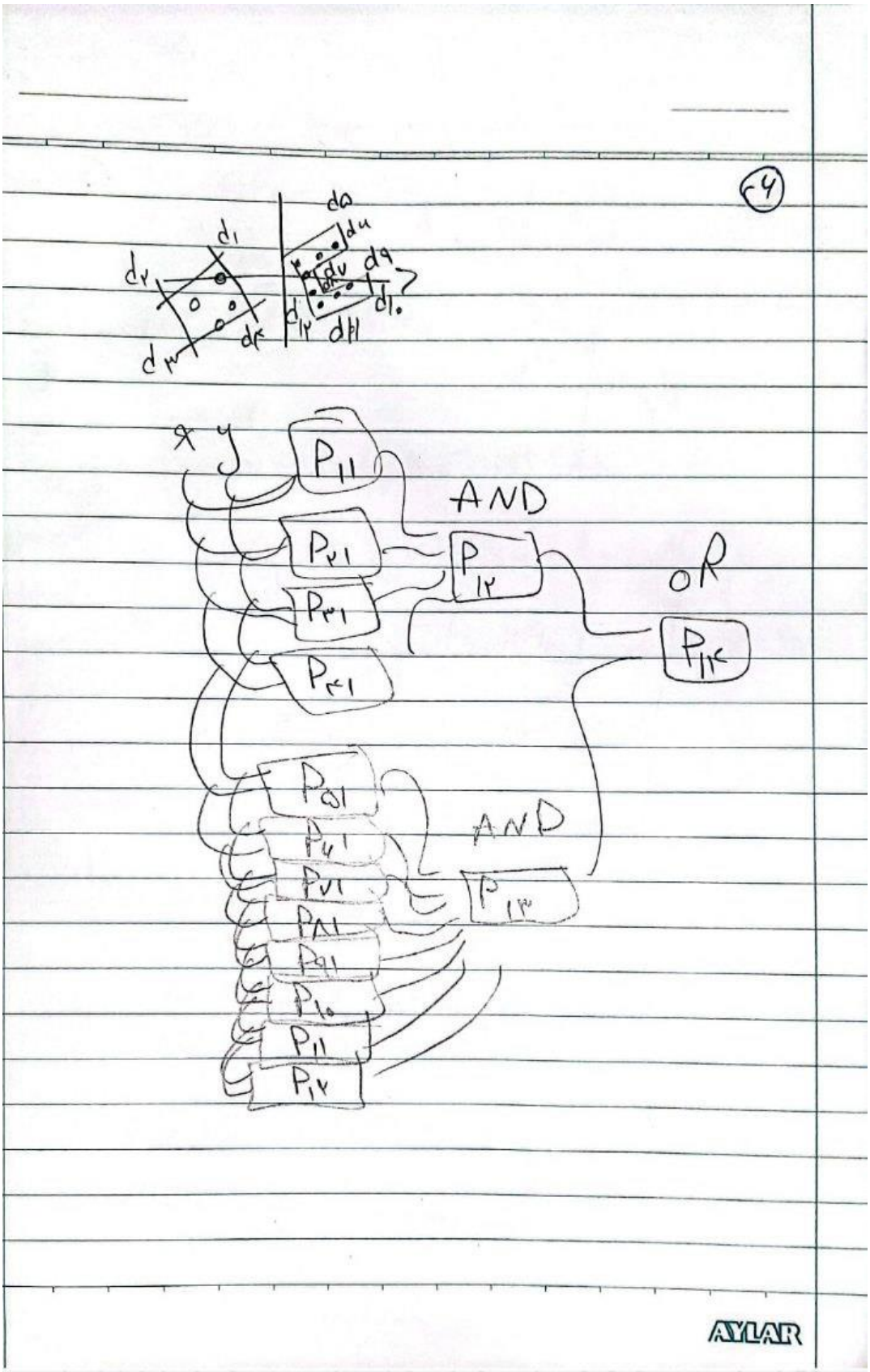




پرسپترون AND که مثبت شوند در همه جهات داخل هستند

پاسخ ٦.

برای تفکیک کلاس نارنجی در مجموع به ۱۲ پرسپترون نیاز داریم ٤ تا برای دسته سمت چپ نمودار و ۸ تا برای دسته سمت راست که همه این ها شامل معادله خط هستند ورودی هم مختصات هستند. بعد از این مرحله به ۲ پرسپترون AND برای هر دسته نیاز داریم که بررسی کنیم آیا نقاط داخل دسته هستند یا خیر و در نهایت یک پرسپترون برای OR که اگر در یکی از این ۲ بود خروجی ۱ شود و گرنه ۰.



پاسخ ۷.

لیست مجموعه فرمول هایی که استفاده شده است.

$$net(P) = \sum w_{ij} y_i(P)$$

$$y_j = f(net_j(P))$$

y_j سیگموید

$$E(P) = \frac{1}{2}[d(P) - y(P)]^2$$

$$E = \sum E(P)$$

$$\frac{\partial E(P)}{\partial w_{ij}} = \frac{\partial E(P)}{\partial net(P)} \cdot \frac{\partial net_j(P)}{\partial w_{ij}}$$

$$= -\delta_j(P) \cdot y_i(P)$$

$$new\ w = old\ w - (\mu \times \frac{\partial E}{\partial w})$$

μ: Learning rate

$$\text{لایه آخر } w_{ij} = [d_j(P) - y_j(P)] \cdot f_j'(net_j(P))$$

$$\text{لایه میانی} = \left[\sum_k \delta_k(P) \cdot w_{jk}\right] \cdot f'(net_j(P))$$

AYLAR

H_1 ۰/۰۰۷۵ ۰/۰۲

net (P) = (۰/۰۵ × ۰/۱۵) + (۰/۲۰ × ۰/۱۰) + ۰/۳۵ = ۰/۳۷۷۵

۰/۰۱۲۵ ۰/۰۱۵

net H_2 = (۰/۰۵ × ۰/۲۵) + (۰/۰۵ × ۰/۳۰) + ۰/۳۵ = ۰/۳۷۷۵

Sigmoid H_1 = ۰/۵۹ $f'(H_1)$ = ۰/۲۴

Sigmoid H_2 = ۰/۵۹ $f'(H_2)$ = ۰/۲۴

۰/۲۳۶ ۰/۲۶۵۵ ۱/۱۰۱۵

net y_1 = (۰/۴۰ × ۰/۵۹) + (۰/۴۵ × ۰/۵۹) + ۰/۶۰ = ۱/۱۰۱۵

۰/۲۹۵ ۰/۳۲۴۵

net y_2 = (۰/۵۰ × ۰/۵۹) + (۰/۵۵ × ۰/۵۹) + ۰/۶۰ =
۱/۲۱۹۵

Sigmoid y_1 = ۰/۷۵ $f'(y_1)$ = ۰/۱۸

Sigmoid y_2 = ۰/۷۷ $f'(y_2)$ = ۰/۱۷

w_8 = δ_j = [۰/۹۹ − ۰/۷۷] · ۰/۱۷ = ۰/۰۳۷۴

$\frac{\partial E(P)}{\partial w_8}$ = $-\delta_j \times y_2$ = −۰/۰۳۷۴ × ۰/۷۷

= ۰/۰۲۸۷۹۸

new w_8 = ۰/۵۵ − (۰/۵ × ۰/۰۲۸۷۹۸) = ۰/۵۳۵۶۰۱ ★★★★

۰/۰۱۴۳۹۹

★★★★★

AYLAR

* چون w_4 لایه میانی هست باید از فرمول سیگما دار حساب بکنیم

$$\text{new } W_4 = ۰/۳۰ - (۰/۵ \times ۰/۰۰۵۵۷۴۷۹۲) = ۰/۲۹۷۲$$

۰/۲۴

$$-۰/۰۵۹۹۴ \qquad ۰/۰۲۰۵۷$$

$$\delta_J = (-۰/۱۳۳۲ \times ۰/۴۵) + (۰/۰۳۷۴ \times ۰/۵۵) \times f'(H_2) = -۰/۰۰۹۴۴۸۸$$

$$-۰/۷۴$$

$$\delta_{k,y_1} = [۰/۰۱ - ۰/۷۵] \times ۰/۱۸ = -۰/۱۳۳۲$$

از لایه بعدی برای حساب گرادیان های لایه بعدی تا لایه میانی حساب شود

$$* \delta_J = \left(\sum \delta_k(p) \cdot w_{Jk}\right) \cdot f'_J(net_J\, p)$$

$$(\delta y_1 \times w_7) + (\delta y_2 \times w_8) \times f'(H_2)$$

$$* \delta y_1 = (target - y_1) \times f'(y_1)$$

$$\frac{\partial E}{\partial w_4} = +۰/۰۰۹۴۴۸۸ \times ۰/۵۹ = ۰/۰۰۵۵۷۴۷۹۲$$

$$\text{new } w_4 = ۰/۲۹۷۲$$

$$\text{new } w_8 = ۰/۵۳۵۴۰۱$$