تمرین سوم درس هوش مصنوعی

دانشکده مهندسی برق و کامپیوتر

فربد سیاهکلی ۸۱۰۱۹۸۵۱۰

درخت تصمیم

قسمت اول) ابتدا آنتروپی اولیه را حساب کرده و سپس آنتروپی وزن‌دار را برای هر فیچر محاسبه می‌کنیم. سپس Information Gain هرکدام نسبت به آنتروپی اولیه را مقایسه می‌کنیم تا گره اول درخت را تعیین کنیم.

$$I(Ham, Spam) = -\frac{Ham}{Ham + Spam}\log_2(Ham) - \frac{Spam}{Ham + Spam}\log_2(Spam)$$

$$E(Detect) = I(5,9) = 0.94$$

$$AE(Domain) = \frac{6}{14}I(2,4) + \frac{4}{14}I(2,2) + \frac{4}{14}I(1,3) = \frac{6}{14}(0.918) + \frac{4}{14}(1) + \frac{4}{14}(0.811)$$

$$AE(Format) = \frac{8}{14}I(2,6) + \frac{6}{14}I(3,3) = \frac{8}{14}(0.811) + \frac{6}{14}(1)$$

$$AE(Length) = \frac{5}{14}I(2,3) + \frac{5}{14}I(3,2) + \frac{4}{14}I(0,4) = \frac{5}{14}(0.971) + \frac{5}{14}(0.971) + \frac{4}{14}(0)$$

$$AE(Complain) = \frac{7}{14}I(4,3) + \frac{7}{14}I(1,6) = \frac{7}{14}(0.985) + \frac{7}{14}(0.592)$$

آنتروپی وزن دار فیچرها به شرح زیر است:

$$AE(Domain) = 0.911$$

$$AE(Format) = 0.892$$

$$AE(Length) = 0.694$$

$$AE(Complain) = 0.788$$

در اینجا مشاهده می‌شود که بیشترین Information gain را فیچر طول ایمیل دارد در نتیجه گره اولیه را طول ایمیل در نظر می‌گیریم. برای طول بلند داریم:

$$E(Long) = 0$$

برای طول متوسط داریم:

$$E(Medium) = I(3,2)$$

$$AE(Domain) = \frac{2}{5}I(1,1) + \frac{2}{5}I(2,0) + \frac{1}{5}I(0,1) > 0$$

$$AE(Format) = \frac{3}{5}I(2,1) + \frac{2}{5}I(1,1) > 0$$

$$AE(Complain) = \frac{3}{5}I(3,0) + \frac{2}{5}I(0,2) = 0$$

در نتیجه نرخ شکایت را که دارای بیشترین Information Gain است، به عنوان گره بعدی این شاخه در نظر می‌گیریم.

برای نرخ شکایت بالا داریم:

$$E(High) = I(3,0) = 0$$

که آنتروپی آن صفر شده و این گره قابلیت جداسازی کامل را دارد. حال برای نرخ شکایت پایین داریم:

$$E(Low) = I(0,2) = 0$$

برای طول کوتاه نیز خواهیم داشت:

$$AE(Format) = \frac{3}{5}I(0,3) + \frac{2}{5}I(2,0) = 0$$

در نتیجه گره را فرمت در نظر می‌گیریم:

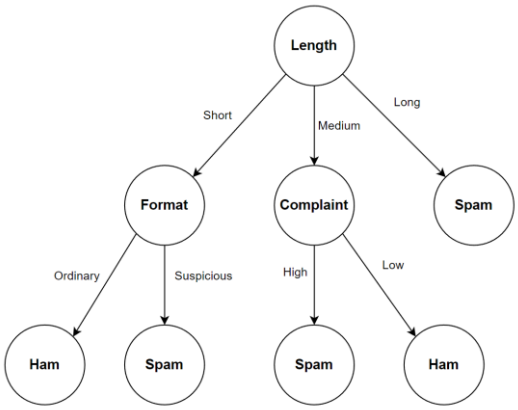
برای فرمت مشکوک داریم:

$$E(Sus) = I(2,0) = 0$$

برای فرمت نامشکوک نیز داریم:

$$E(Norm) = I(0,3) = 0$$

در نتیجه آنتروپی گره ها صفر شده و درخت تصمیم گیری با دقت ۱۰۰ درصدی بر روی دیتای آموزشی بدست آمد.



قسمت دوم) حال بر روی دیتای تست ارزیابی درخت را انجام می‌دهیم:

Number	Actual	Predicted
1	Ham	Spam
2	Spam	Ham
3	Spam	Spam
4	Spam	Spam
5	Ham	Ham
6	Spam	Spam

قسمت سوم) ماتریس Confusion را برای نتایج ارزیابی خود محاسبه و ترسیم می‌کنیم.

		Actual	
		Spam	Ham
Predicted	Spam	3	1
	Ham	1	1

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} = \frac{4}{6} = 67\%$$

$$Precision = \frac{TP}{TP + FP} = \frac{3}{3 + 1} = 75\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{3}{3 + 1} = 75\%$$

قسمت چهارم)

۱. تفاوت اصلی بین روش‌های یادگیری ماشین Random Forest و درخت تصمیم در نحوه پیش‌بینی و مدیریت پیچیدگی داده‌ها نهفته است.

- درخت تصمیم: درخت تصمیم یک ساختار فلوچارت مانند است که در آن هر گره داخلی یک ویژگی، هر شاخه یک قانون تصمیم گیری، و هر گره برگ نشان دهنده یک نتیجه یا پیش بینی است. به صورت بازگشتی داده ها را بر اساس ویژگی های انتخاب شده برای ایجاد یک مدل درخت مانند تقسیم می شود. درختان تصمیم گیری مستعد Overfitting هستند، زیرا می توانند نویز و نقاط پرت را در داده ها ثبت کنند.
- الگوریتم Random Forest: جنگل تصادفی یک روش یادگیری گروهی است که چندین درخت تصمیم را برای پیش‌بینی ترکیب می‌کند. با انتخاب تصادفی زیرمجموعه هایی از داده ها و ویژگی های آموزشی، یک "جنگل" از درختان تصمیم ایجاد می کند. هر درخت تصمیم در جنگل به طور مستقل از یک زیرمجموعه تصادفی از داده ها یاد می گیرد و یک پیش بینی ارائه می دهد. پیش‌بینی نهایی از طریق رای‌گیری یا میانگین‌گیری پیش‌بینی تک درختان به دست می‌آید. Random Forest با میانگین‌گیری پیش‌بینی‌های چند درخت، Overfitting را کاهش می‌دهند که منجر به نتایج قوی‌تر و دقیق‌تر می‌شود.

۲. استفاده از روش Random Forest:

- Random Forest در درجه اول برای کارهای طبقه بندی و رگرسیون استفاده می شوند.
- طبقه‌بندی: Random Forest می‌توانند مشکلات طبقه‌بندی باینری و چند طبقه‌ای را مدیریت کنند. آنها در مدیریت مجموعه داده های پیچیده و با ابعاد بالا موثر هستند و می توانند روابط غیرخطی بین ویژگی ها و متغیر هدف را ثبت کنند.
- رگرسیون: Random Forest همچنین می توانند برای کارهای رگرسیونی استفاده شوند، جایی که خروجی پیوسته ارائه می دهند. آن‌ها می‌توانند روابط غیرخطی بین ویژگی‌ها و متغیر هدف را مدیریت کنند و در برابر نقاط پرت و نویز در داده‌ها مقاوم هستند.

مزایای Random Forest:

- جنگل‌های تصادفی با ترکیب چندین درخت و میانگین‌گیری پیش‌بینی‌های آن‌ها، Overfitting را در مقایسه با درختان تصمیم‌گیری فردی کاهش می‌دهند.
- آنها قادر به مدیریت مجموعه داده های بزرگ با ابعاد بالا و تعامل ویژگی های پیچیده هستند.
- جنگل‌های تصادفی معیارهای اهمیت ویژگی را ارائه می‌کنند و به کاربران اجازه می‌دهند اهمیت و تأثیر ویژگی‌های مختلف بر پیش‌بینی را درک کنند.
- آنها از نظر محاسباتی کارآمد هستند، زیرا درختان تصمیم گیری فردی را می توان به صورت موازی آموزش داد.

۳. بایاس و واریانس دو معیاری هستند که برای ارزیابی عملکرد و تعمیم مدل‌های یادگیری ماشین استفاده می‌شوند.

- بایاس: بایاس تفاوت بین مقادیر پیش بینی شده توسط مدل و مقادیر واقعی در داده های آموزشی را اندازه گیری می کند. یک سوگیری زیاد نشان می‌دهد که مدل مشکل را بیش از حد ساده می‌کند و نمی‌تواند الگوهای بنیادی را به تصویر بکشد، و در نتیجه عدم تناسب دارد. درختان تصمیم گیری تمایل به تعصب بالاتری دارند، زیرا آنها فقط می توانند مرزهای تصمیم گیری نسبتا ساده ای ایجاد کنند.
- واریانس: واریانس تغییرپذیری یا ناپایداری پیش‌بینی‌های مدل را در مجموعه‌های آموزشی مختلف اندازه‌گیری می‌کند. میزان حساسیت مدل به نوسانات داده های آموزشی را کمیت می کند. واریانس بالا نشان دهنده Overfitting است، جایی که مدل بسیار پیچیده است و نویز یا نوسانات تصادفی در داده های آموزشی را ضبط می کند. جنگل‌های تصادفی با جمع‌آوری درخت‌های تصمیم‌گیری متعدد، واریانس را کاهش می‌دهند که منجر به پیش‌بینی‌های پایدارتر می‌شود.

دلیل تفاوت در معیارهای بایاس و واریانس بین درخت‌های تصمیم و جنگل‌های تصادفی این است که درخت‌های تصمیم تمایل به سوگیری بالا و واریانس کم دارند، در حالی که جنگل‌های تصادفی هدفشان کاهش واریانس است. این Tradeoff به جنگل‌های تصادفی اجازه تعمیم بهتر و ارائه پیش‌بینی‌های دقیق‌تر در مورد داده‌های دیده نشده را در مقایسه با درخت‌های تصمیم فردی می‌دهد.

شبکه عصبی

قسمت اول)

$$H_1 = w_1x_1 + w_2x_2 + b_1 = 0.05 \times 0.15 + 0.10 \times 0.20 + 0.35 = 0.3775$$

$$Sig(H_1) = \frac{1}{1 + e^{-H_1}} = 0.5933$$

$$H_2 = w_3x_1 + w_4x_2 + b_1 = 0.05 \times 0.25 + 0.10 \times 0.30 + 0.35 = 0.3925$$

$$Sig(H_2) = \frac{1}{1 + e^{-H_2}} = 0.5969$$

$$Y_1 = w_5H_1 + w_6H_2 + b_2 = 0.5933 \times 0.4 + 0.5969 \times 0.45 + 0.6 = 1.1059$$

$$Sig(Y_1) = \frac{1}{1 + e^{-Y_1}} = 0.7514$$

$$Y_2 = w_7H_1 + w_8H_2 + b_2 = 0.5933 \times 0.5 + 0.5969 \times 0.55 + 0.6 = 1.2249$$

$$Sig(Y_2) = \frac{1}{1 + e^{-Y_2}} = 0.7729$$

$$E_{total} = \frac{1}{2}(0.01 - 0.7514)^2 + \frac{1}{2}(0.99 - 0.7729)^2 = 0.2984$$

$$\frac{\partial E_{total}}{\partial w_5} = \frac{\partial E_{total}}{\partial sig(Y_1)} \frac{\partial sig(Y_1)}{\partial Y_1} \frac{\partial Y_1}{\partial w_5}$$

$$\frac{\partial E_{total}}{\partial sig(Y_1)} = \frac{\partial \left[\frac{1}{2}(t_1 - sig(Y_1))^2 + \frac{1}{2}(t_2 - sig(Y_2))^2\right]}{\partial sig(Y_1)} = (-1)\big(t_1 - sig(Y_1)\big) + 0$$

$$\frac{\partial E_{total}}{\partial sig(Y_1)} = 0.7414$$

$$\frac{\partial sig(Y_1)}{\partial Y_1} = \frac{\partial \left[\frac{1}{1 + e^{-Y_1}}\right]}{\partial Y_1} \frac{e^{-Y_1}}{(1 + e^{-Y_1})^2} = e^{-Y_1} \times [sig(Y_1)^2] = sig(Y_1) \times \big(1 - sig(Y_1)\big)$$
$$= 0.1868$$

$$\frac{\partial Y_1}{\partial w_5} = \frac{\partial(w_5 H_1 + w_6 H_2 + b_2)}{\partial w_5} = H_1 = 0.5969$$

$$\rightarrow Error_{w_5} = \frac{\partial E_{total}}{\partial w_5} = 0.7414 \times 0.1868 \times 0.5969 = 0.0822$$

$$\rightarrow w_5^+ = w_5^- - \alpha \times \frac{\partial E_{total}}{\partial w_5} = 0.4 - 0.5 \times 0.0822 = 0.3589$$

قسمت دوم)

دو تابع فعال سازی محبوب مورد استفاده در شبکه های عصبی عبارتند از: واحد خطی اصلاح شده (ReLU) و تابع سیگموئید.

(ReLU):

مزایا:

- ReLU یک تابع فعال سازی ساده است که از نظر محاسباتی کارآمد است و پیاده سازی آن آسان است.
- به دلیل توانایی آن در کاهش مشکل گرادیان ناپدید شدن، که با سایر توابع فعال سازی مانند سیگموئید یا tanh رایج است، به طور گسترده در یادگیری عمیق مورد استفاده قرار گرفته است.
- ReLU با صفر کردن مقادیر منفی، فعال‌سازی‌های پراکنده را ایجاد می‌کند، که می‌تواند با کاهش تعداد نورون‌های فعال، پراکندگی را افزایش داده و کارایی شبکه را افزایش دهد.
- این امکان همگرایی آموزشی سریعتر را فراهم می کند زیرا دارای یک فعال سازی خطی برای مقادیر مثبت است و از مسائل اشباع مرتبط با سایر توابع فعال سازی جلوگیری می کند.

معایب:

- ReLU می تواند از مشکل "ReLU در حال مرگ" رنج ببرد، جایی که برخی از نورون ها به طور دائم غیرفعال می شوند و دیگر هرگز شلیک نمی کنند. این زمانی اتفاق می‌افتد که ورودی یک نورون منفی می‌شود و باعث می‌شود که گرادیان صفر شود و نورون دیگر در طول تمرین به‌روزرسانی نشود.
- ReLU برای مدل‌هایی که نیاز به فعال‌سازی منفی دارند، مناسب نیست زیرا ورودی‌های منفی را صفر می‌کند و به‌طور موثر هر مقدار منفی را حذف می‌کند.

تابع سیگموئید:

مزایا:

- تابع سیگموئید دارای یک خروجی صاف و محدود بین ۰ و ۱ است که آن را برای مسائل طبقه بندی باینری که در آن خروجی احتمالات را نشان می دهد مناسب است.
- در طول تاریخ در شبکه های عصبی به عنوان یک تابع فعال سازی و به عنوان جزئی از مدل رگرسیون لجستیک استفاده شده است.

- تابع سیگموئید مشتق خوبی دارد که آن را برای الگوریتم های بهینه سازی مبتنی بر گرادیان مانند انتشار پس زمینه مفید می کند.

معایب:

- تابع سیگموئید از مشکل گرادیان ناپدید شدن رنج می برد، به ویژه برای ورودی های با بزرگی زیاد، که می تواند منجر به همگرایی کند در طول تمرین شود.
- این تابع ورودی ها را به یک محدوده محدود (۰ تا ۱) نگاشت می کند، و زمانی که ورودی دور از ۰ است، باعث "اشباع" می شود. این اشباع منجر به شیب های کوچکی می شود و توانایی مدل برای یادگیری را کاهش می دهد و روند آموزش را کند می کند.
- خروجی های تابع سیگموئید صفر محور نیستند، که می تواند یادگیری موثر لایه های زیر را چالش برانگیز کند، به خصوص در شبکه های عمیق.

شایان ذکر است که چندین activation function دیگر نیز وجود دارد که هر کدام مزایا و معایب خاص خود را دارند و انتخاب تابع فعال سازی به وظیفه خاص و معماری شبکه بستگی دارد.