یادگیری ماشین | دکتر منصور رزقی

تمرین سوم | دانشکده علوم ریاضی ، گروه علوم کامپیوتر، گرایش داده‌کاوی

سبحان عیوضی پور کفشدوزی | ۴۰۴۵۲۵۲۱۰۰۴

تمرین ۱:

برای هر تابع بولی، جدول درستی و درخت تصمیم را رسم می‌کنیم.

- ۱) $Y = A \wedge \neg B$:

A	B	Y
T	T	F
T	F	T
F	T	F
F	F	F

$$p_T = \frac{1}{4}, \quad p_F = \frac{3}{4}$$

$$Entropy(S) = -p_T\, log_2\, p_T - p_F\, log\, p_F = -\frac{1}{4}log\frac{1}{4} - \frac{3}{4}log\frac{3}{4} \approx 0.811$$

$$Entropy(S|A=T) = -p(Y=T|A=T)\, log_2\, p(Y=T|A=T)$$
$$-p(Y=F|A=T)\, log_2\, p(Y=F|A=T) = -\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2} = 1$$

$$Entropy(S|A=F) = -p(Y=T|A=F)\, log_2\, p(Y=T|A=F)$$
$$-p(Y=F|A=F)\, log_2\, p(Y=F|A=F) = -\frac{0}{2}log_2\frac{0}{2} - \frac{2}{2}log_2\frac{2}{2} = 0$$

$$Entropy(S|B=T) = -p(Y=T|B=T)\, log_2\, p(Y=T|B=T)$$
$$-p(Y=F|B=T)\, log_2\, p(Y=F|B=T) = -\frac{0}{2}log_2\frac{0}{2} - \frac{2}{2}log_2\frac{2}{2} = 0$$

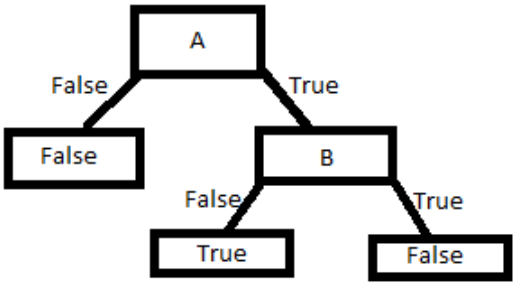
$$Entropy(S|B=F) = -p(Y=T|B=F)\, log_2\, p(Y=T|B=F)$$
$$-p(Y=F|B=F)\, log_2\, p(Y=F|B=F) = -\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2} = 1$$

$$Gain(S|A) = Entropy(S) - [\frac{N_{A=T}}{N}Entropy(S|A=T) + \frac{N_{A=F}}{N}Entropy(S|A=F)]$$
$$= 0.811 - [\frac{2}{4} \times 1 + \frac{2}{4} \times 0] = 0.811 - [0.5 + 0] = 0.811 - 0.5 = 0.311$$

$$Gain(S|B) = Entropy(S) - [\frac{N_{B=T}}{N} Entropy(S|B=T) + \frac{N_{B=F}}{N} Entropy(S|B=F)]$$

$$= 0.811 - [\frac{2}{4} \times 0 + \frac{2}{4} \times 1] = 0.811 - [0 + 0.5] = 0.811 - 0.5 = 0.311$$

چون $Gain(S|A) = Gain(S|B)$، فرقی نمی‌کند کدام مولفه را به عنوان گره ریشه درخت تصمیم درنظر بگیریم. به دلخواه، مولفه A را به عنوان گره ریشه انتخاب می‌کنیم. اگر $A = F$ باشد، $Y = F$ خواهد بود. در غیر این صورت، B تعیین‌کننده است. اگر $B = F$ باشد $Y = T$ خواهد بود و در غیر این صورت، $Y = F$ خواهد بود. درخت تصمیم نهایی، بدین صورت است:



- ۲) $Y = A \vee (B \wedge C)$:

A	B	C	Y
T	T	T	T
T	T	F	T
T	F	T	T
T	F	F	T
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	F

$$p_T = \frac{5}{8}, \quad p_F = \frac{3}{8}$$

$$Entropy(S) = -p_T \, log_2 \, p_T - p_F \, log \, p_F = -\frac{5}{8} log \frac{5}{8} - \frac{3}{8} log \frac{3}{8} \approx 0.954$$

$$Entropy(S|A=T) = -p(Y=T|A=T) \, log_2 \, p(Y=T|A=T)$$
$$-p(Y=F|A=T) \, log_2 \, p(Y=F|A=T) = -\frac{4}{4} log_2 \frac{4}{4} - \frac{0}{4} log_2 \frac{0}{4} = 0$$

$$Entropy(S|A=F) = -p(Y=T|A=F) \, log_2 \, p(Y=T|A=F)$$
$$-p(Y=F|A=F) \, log_2 \, p(Y=F|A=F) = -\frac{1}{4} log_2 \frac{1}{4} - \frac{3}{4} log_2 \frac{3}{4} \approx 0.811$$

$$Entropy(S|B=T) = -p(Y=T|B=T)\ log_2\ p(Y=T|B=T)$$
$$-p(Y=F|B=T)\ log_2\ p(Y=F|B=T) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4} \approx 0.811$$

$$Entropy(S|B=F) = -p(Y=T|B=F)\ log_2\ p(Y=T|B=F)$$
$$-p(Y=F|B=F)\ log_2\ p(Y=F|B=F) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1$$

$$Entropy(S|C=T) = -p(Y=T|C=T)\ log_2\ p(Y=T|C=T)$$
$$-p(Y=F|C=T)\ log_2\ p(Y=F|C=T) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4} \approx 0.811$$

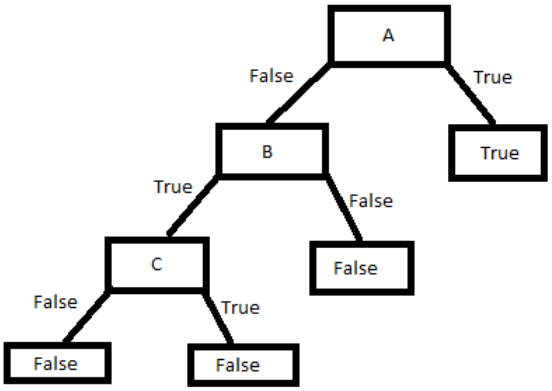
$$Entropy(S|C=F) = -p(Y=T|C=F)\ log_2\ p(Y=T|C=F)$$
$$-p(Y=F|C=F)\ log_2\ p(Y=F|C=F) = -\frac{2}{4}log_2\frac{2}{4} - \frac{2}{4}log_2\frac{2}{4} = 1$$

$$Gain(S|A) = Entropy(S) - [\frac{N_{A=T}}{N}Entropy(S|A=T) + \frac{N_{A=F}}{N}Entropy(S|A=F)]$$
$$= 0.954 - [\frac{4}{8} \times 0 + \frac{4}{8} \times 0.811] = 0.954 - [0 + 0.406] = 0.954 - 0.406 = 0.548$$

$$Gain(S|B) = Entropy(S) - [\frac{N_{B=T}}{N}Entropy(S|B=T) + \frac{N_{B=F}}{N}Entropy(S|B=F)]$$
$$= 0.954 - [\frac{4}{8} \times 0.811 + \frac{4}{8} \times 1] = 0.954 - [0.406 + 0.5] = 0.954 - 0.906 = 0.048$$

$$Gain(S|C) = Entropy(S) - [\frac{N_{C=T}}{N}Entropy(S|C=T) + \frac{N_{C=F}}{N}Entropy(S|C=F)]$$
$$= 0.954 - [\frac{4}{8} \times 0.811 + \frac{4}{8} \times 1] = 0.954 - [0.406 + 0.5] = 0.954 - 0.906 = 0.048$$

چون $Gain(S|A)$ بیشترین است، مولفه A را به عنوان گره ریشه انتخاب می‌کنیم. اگر $A = T$ باشد، $Y = T$ خواهد بود. در غیر این صورت، باید گره ریشه زیردرخت مرتبط با این حالت را تعیین کنیم. چون ترکیب $B \wedge C$ عطفی است. تفاوتی در انتخاب گره بعدی وجود ندارد. به دلخواه B را انتخاب می‌کنیم. اگر $A = F$ و $B = F$ باشد، $Y = F$ خواهد بود، اما اگر $A = F$ و $B = T$ باشد، مقدار نهایی Y، برابر مقدار C خواهد بود. بدین ترتیب، درخت بهینه را تعیین نمودیم.

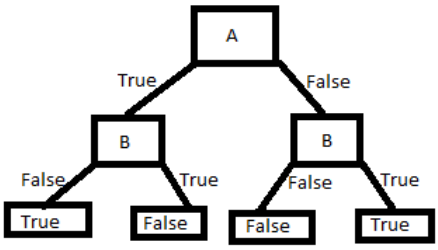


• ۳) $Y = A \oplus B$:

A	B	Y
T	T	F
T	F	T
F	T	T
F	F	F

$$p_T = \frac{1}{2}, \quad p_F = \frac{1}{2}$$

چون نتایج برای A و B مشابه هستند، فرقی نمی‌کند کدام گره را اول انتخاب کنیم. همچنین، هر گره‌ای را انتخاب کنیم، نتیجه Y بر اساس گره دیگر تعیین می‌شود . پس درخت بهینه، درخت دودویی متوازن، پر و کامل است.

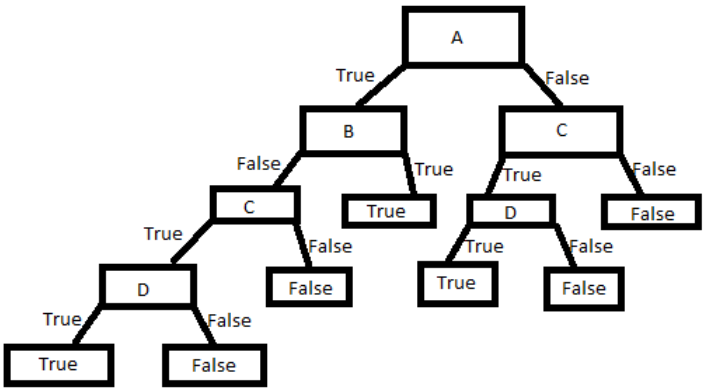


• ۴) $Y = (A \wedge B) \vee (C \wedge D)$:

A	B	C	D	Y
T	T	T	T	T
T	T	T	F	T
T	T	F	T	T
T	T	F	F	T
T	F	T	T	T
T	F	T	F	F
T	F	F	T	F
T	F	F	F	F
F	T	T	T	T
F	T	T	F	F
F	T	F	T	F
F	T	F	F	F
F	F	T	T	T
F	F	T	F	F
F	F	F	T	F
F	F	F	F	F

$$p_T = \frac{7}{16}, \quad p_F = \frac{9}{16}$$

این ترکیب فصلی دو ترکیب عطفی دارای تقارن است. فرقی نمی‌کند کدام مولفه را اول انتخاب کنیم. به دلخواه A را انتخاب می‌کنیم. اگر $A = T$ باشد، بر اساس آنچه در بخش ۲ توضیح داده شد، زیردرخت، قابل تعیین است. زیرا نتایج نهایی و احتمالات به شرط $A = T$، کاملاً یکسان هستند. اگر $A = F$ باشد، کل ترکیب عطفی $A \wedge B$ نادرست است. پس B را کنار گذاشته و به سراغ C و D می‌رویم. چون نتایج برای هر دو یکسان است، فرقی نمی‌کند کدام گره را انتخاب کنیم. مهم این است که هر دو درست باشند تا عبارت کل هم درست شود. بدین ترتیب، درخت بهینه تعیین شد.



تمرین ۲:

ابتدا آنتروپی کل را حساب می‌کنیم:

$$Entropy(S) = -p(well-behaved)log_2\, p(well-behaved) - p(dangerous)log_2\, p(dangerous)$$
$$= -\frac{4}{7}log_2\frac{4}{7} - \frac{3}{7}log_2\frac{3}{7} \approx -0.571 \times (-0.807) - 0.429 \times (-1.222) \approx 0.461 + 0.524 = 0.985$$

در ادامه، $iGain$ را برای هر ویژگی حساب می‌کنیم:

- Size :

$$Entropy(S|Size = small) = -p(well-behaved|small)log_2\, p(well-behaved|small)$$
$$-p(dangerous|small)log_2\, p(dangerous|small) = -\frac{3}{4}log_2\frac{3}{4} - \frac{1}{4}log_2\frac{1}{4}$$
$$\approx -0.75 \times (-0.415) - 0.25 \times (-2) \approx 0.311 + 0.5 = 0.811$$

$$Entropy(S|Size = big) = -p(well-behaved|big)log_2\, p(well-behaved|big)$$
$$-p(dangerous|big)log_2\, p(dangerous|big) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3}$$
$$\approx -0.333 \times (-1.585) - 0.667 \times (-0.585) \approx 0.528 + 0.390 = 0.918$$

آنتروپی را به ازای مقادیر مختلف $Size$ حساب کردیم. در ادامه، $Gain(S, Size)$ را حساب می‌کنیم:

$$Entropy(S) - [\frac{N_{small}}{N}Entropy(S|Size = small) + \frac{N_{big}}{N}Entropy(S|Size = big)]$$
$$= 0.985 - [\frac{4}{7} \times 0.811 + \frac{3}{7} \times 0.918] \approx 0.985 - [0.463 + 0.393] = 0.985 - 0.856 = 0.129$$

- Fur :

$$Entropy(S|Fur = ragged) = -p(well{-}behaved|ragged)log_2\, p(well{-}behaved|ragged)$$
$$-p(dangerous|ragged)log_2\, p(dangerous|ragged) = -\frac{2}{3}log_2\frac{2}{3} - \frac{1}{3}log_2\frac{1}{3}$$
$$\approx -0.667 \times (-0.585) - 0.333 \times (-1.585) \approx 0.390 + 0.528 = 0.918$$

$$Entropy(S|Fur = smooth) = -p(well{-}behaved|smooth)log_2\, p(well{-}behaved|smooth)$$
$$-p(dangerous|smooth)log_2\, p(dangerous|smooth) = -\frac{0}{2}log_2\frac{0}{2} - \frac{2}{2}log_2\frac{2}{2} = 0$$

$$Entropy(S|Fur = curly) = -p(well - behaved|curly)log_2\, p(well - behaved|curly)$$
$$-p(dangerous|curly)log_2\, p(dangerous|curly) = -\frac{2}{2}log_2\frac{2}{2} - \frac{0}{2}log_2\frac{0}{2} = 0$$

آنتروپی را به ازای مقادیر مختلف Fur حساب کردیم. در ادامه، $Gain(S, Fur)$ را حساب می‌کنیم:

$$Entropy(S) - [\frac{N_{ragged}}{N}Entropy(S|Fur = ragged) + \frac{N_{smooth}}{N}Entropy(S|Fur = smooth)$$
$$+\frac{N_{curly}}{N}Entropy(S|Fur = curly)] = 0.985 - [\frac{3}{7} \times 0.918 + \frac{2}{7} \times 0 + \frac{2}{7} \times 0]$$
$$\approx 0.985 - [0.393 + 0 + 0] = 0.985 - 0.393 = 0.592$$

- Color :

$$Entropy(S|Color = brown) = -p(well{-}behaved|brown)log_2\, p(well{-}behaved|brown)$$
$$-p(dangerous|brown)log_2\, p(dangerous|brown) = -\frac{1}{1}log_2\frac{1}{1} - \frac{0}{1}log_2\frac{0}{1} = 0$$

$$Entropy(S|Color = black) = -p(well - behaved|black)log_2\, p(well - behaved|black)$$
$$-p(dangerous|black)log_2\, p(dangerous|black) = -\frac{1}{3}log_2\frac{1}{3} - \frac{2}{3}log_2\frac{2}{3}$$
$$\approx 0.333 \times (-1.585) + 0.666 \times (-0.585) \approx 0.528 + 0.390 = 0.918$$

$$Entropy(S|Color = white) = -p(well{-}behaved|white)log_2\, p(well{-}behaved|white)$$
$$-p(dangerous|white)log_2\, p(dangerous|white) = -\frac{1}{2}log_2\frac{1}{2} - \frac{1}{2}log_2\frac{1}{2}$$
$$= -0.5 \times (-1) - 0.5 \times (-1) = 1$$

$$Entropy(S|Color = red) = -p(well-behaved|red)log_2\, p(well-behaved|red)$$
$$-p(dangerous|red)log_2\, p(dangerous|red) = -\frac{1}{1}log_2\frac{1}{1} - \frac{0}{1}log_2\frac{0}{1} = 0$$

آنتروپی را به ازای مقادیر مختلف $Color$ حساب کردیم. در ادامه، $Gain(S, Color)$ را حساب می‌کنیم:

$$Entropy(S) - [\frac{N_{brown}}{N}Entropy(S|Color = brown) + \frac{N_{black}}{N}Entropy(S|Color = black)$$
$$+\frac{N_{white}}{N}Entropy(S|Color = white) + \frac{N_{red}}{N}Entropy(S|Color = red)] = 0.985$$
$$-[\frac{1}{7}\times 0 + \frac{3}{7}\times 0.918 + \frac{2}{7}\times 1 + \frac{1}{7}\times 0] \approx 0.985 - [0+0.393+0.286+0] = 0.985 - 0.679 = 0.306$$

حال، $iGain$ ها را با هم مقایسه می‌کنیم:

$$Gain(S, Fur) > Gain(S, Color) > Gain(S, Size)$$

پس Fur، گره ریشه را تعیین می‌کند. در ادامه، زیردرخت‌ها را به ازای مقادیر مختلف Fur، به دست می‌آوریم:

- اگر $Fur = smooth$ باشد، $Class = dangerous$ خواهد بود.
- اگر $Fur = curly$ باشد، $Class = well-behaved$ خواهد بود.
- در ادامه، گره ریشه زیردرخت مرتبط با $Fur = ragged$ را تعیین می‌کنیم.
  بدین منظور، می‌بایست $iGain$ها را محاسبه و مقایسه کنیم:
  - $Gain(S_{ragged}, Size)$ را محاسبه می‌کنیم:

$$Entropy(S|Fur = ragged) - [\frac{N_{small,ragged}}{N_{ragged}}Entropy(S|small, ragged)$$
$$+\frac{N_{big,ragged}}{N_{ragged}}Entropy(S|big, ragged)] = 0.918 - [\frac{1}{3}\times 0 + \frac{2}{3}\times 1]$$
$$\approx 0.918 - [0 + 0.667] = 0.918 - 0.667 = 0.251$$

  - $Gain(S_{ragged}, Color)$ را محاسبه می‌کنیم:

$$Entropy(S|Fur = ragged) - [\frac{N_{brown,ragged}}{N_{ragged}}Entropy(S|brown, ragged)$$
$$+\frac{N_{black,ragged}}{N_{ragged}}Entropy(S|black, ragged) + \frac{N_{red,ragged}}{N_{ragged}}Entropy(S|red, ragged)]$$
$$= 0.918 - [\frac{1}{3}\times 0 + \frac{1}{3}\times 0 + \frac{1}{3}\times 0] = 0.918$$

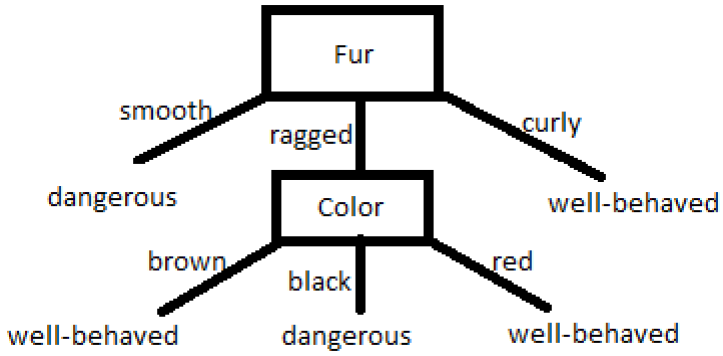
با انجام مقایسه، می‌توان دریافت:

$$Gain(S_{ragged}, Color) > Gain(S_{ragged}, Size)$$

پس $Color$، گره ریشه زیردرخت مرتبط با $Fur = ragged$ را تعیین می‌کند.

- اگر $Color = brown$ و $Fur = ragged$ باشد، $Class = well - behaved$ خواهد بود.
- اگر $Color = black$ و $Fur = ragged$ باشد، $Class = dangerous$ خواهد بود.
- اگر $Color = red$ و $Fur = ragged$ باشد، $Class = well - behaved$ خواهد بود.

بدین ترتیب، درخت تصمیم را به دست آوردیم.



تمرین ۳:

- (الف): جست‌وجو با راهبرد حریصانه یک پارادایم الگوریتمی است که در هر مرحله انتخاب بهینه محلی را انجام می‌دهد، به این امید که بتواند بهینه سراسری را بیابد. در ساخت درخت تصمیم، با بهره گیری از رویکرد بالا به پایین، از ریشه شروع کرده و داده‌ها را به صورت بازگشتی تقسیم می‌کند. همچنین، بهینه‌سازی آن از نوع محلی است؛ یعنی، در هر گره، ویژگی‌ای را انتخاب می‌کند که داده‌ها را به بهترین شکل تقسیم کند. این شاخص انتخاب ویژگی می‌تواند بر اساس معیارهای مختلفی مانند $Information\ Gain$، $Gini\ Index$ و غیره، صورت می‌گیرد. همچنین، خاصیت رویکرد حریصانه این است که به عقب برنمی‌گردد و پس از انجام تقسیم، هرگز بازنگری صورت نمی‌گیرد.
در گره t، با مجموعه داده D_t، برای تمامی ویژگی‌های A_i، اساس معیار $splitting$، یا همان $S(A_i, D_t)$، تمامی ویژگی‌های کاندید را ارزیابی می‌کند. ویژگی بیشینه‌کننده معیار $splitting$ را انتخاب کرده و بر اساس آن، مجموعه داده D_t را افراز می‌کند. این روند را برای مجموعه‌های حاصل‌شده، ادامه می‌دهد تا زمانی که شرط توقف برآورده شود. کارایی محاسباتی این روش، معمولاً $O(n.m.log\ n)$ می‌باشد.
از مهم‌ترین مزایای آن، می‌توان به این موارد اشاره کرد:

  - الگوریتم بازگشتی سرراست و پیاده‌سازی ساده است.
  - اغلب راه‌حل‌های خوبی می‌یابد و در عمل، عملکرد خوبی دارد.
  - می‌تواند داده‌های $numerical$ و $categorical$، را به خوبی مدیریت کند.
  - هر تقسیم به صورت محلی بهینه و قابل درک است.

همچنین، مهم‌ترین معایب آن، عبارت هستند از:

  - نادیده گرفتن پیامدهای آتی تقسیم‌های فعلی بر اساس رویکرد حریصانه
  - امکان ناتوانی در یافتن درخت بهینه سراسری
  - عدم بازگشت و اصلاح تقسیم‌های ضعیف اولیه
  - حساسیت به ترتیب داده
  - تمایل به بیش‌برازش

زمانی راهبرد حریصانه مفید است که نیازی به بهینگی کامل نیست، تفسیرپذیری مهم است، جست‌وجوی جامع برای مجموعه‌های بزرگ امکان‌پذیر نیست، نیاز به پیش‌بینی سریع و یا اینکه مسئله زیرساخت بهینه دارد؛ یعنی بهینه محلی منجر به بهینه سراسری می‌شود.
به عنوان جایگزین این رویکرد، می‌توان از چنین روش‌هایی استفاده کرد:

- $Ensemble\ Methods$: مانند $Random\ Forest$ که چندین درخت با زیرمجموعه‌های تصادفی ویژگی‌ها ایجاد می‌کند و عملکرد آنها را مورد بررسی قرار می‌دهد. می‌توان با بهره‌گیری ایده $boosting$، اصلاح خطاهای درخت‌های قبلی به صورت متوالی را دنبال نمود.
- $Post-pruning$: به جهت بهبود عملکرد مدل، بر اساس پیچیدگی، هزینه محاسباتی یا میزان خطا، $ReducedError\ Pruning$ یا $Cost-Complexity\ Pruning$ را انجام دهیم تا درخت بهتری داشته باشیم.

- (ب):

- الگوریتم $Candidate\ Elimination$، یادگیری فضای نسخه را دنبال می‌کند به گونه‌ای که جستجوی خاص به عام صورت گیرد. سوگیری استقرایی را در این روش توضیح می‌دهیم:
۱- سوگیری بازنمایی: فرض می‌کند مفهوم هدف به صورت ترکیب عطفی محدودیت‌های ویژگی قابل بازنمایی است و نمی‌تواند مفاهیم فصلی را بازنمایی کند.
۲- سوگیری جست‌وجو: اگر دو بخش خاص‌ترین و عام‌ترین را به ترتیب با S و G نمایش دهیم، مرزهای این دو بخش را نگه می‌دارد تا به سمت تعمیم‌های مینیمال از مثال‌های مثبت و به سمت تخصیص‌های مینیمال از مثال‌های منفی جهت‌دار شود.
۳- سوگیری ترتیب: فرضیه‌های سازگار با تمام مثال‌های مشاهده‌شده را ترجیح می‌دهد. بدون سوگیری ترجیح عمل می‌کند و همه فرضیه‌ها در فضای نسخه به یک اندازه محتمل هستند و همچنین، بدون سوگیری محدودیت، فقط ساختار فضای فرضیه یادگیری را محدود می‌کند
اساس منطقی این روش، بدین صورت است که فرض می‌کند مفهوم هدف درون فضای فرضیه اولیه قرار دارد و به صورت ترکیب عطفی محدودیت‌ها قابل بازنمایی است.
- الگوریتم $ID3$، یادگیری درخت تصمیم و جست‌وجوی حریصانه بالا به پایین را دنبال می‌کند. سوگیری استقرایی را در این روش توضیح می‌دهیم:
۱- سوگیری ترجیح: درخت‌های کوتاه‌تر، ویژگی‌های با سود اطلاعاتی بالا که مرزهای تصمیم ساده‌تر دارند، ترجیح داده می‌شوند. همچنین ویژگی‌های مهم در سطوح بالاتر قرار می‌گیرند.
۲- سوگیری بازنمایی: می‌تواند مفاهیم فصلی را از طریق مسیرهای چندگانه بازنمایی کند. مرزهای تصمیم، برای ویژگی‌های پیوسته، موازی با محورها هستند و تعاملات سلسله‌مراتبی ویژگی‌ها رعایت می‌شود.
۳- سوگیری جستجو: جست‌وجو به شکل حریصانه و بر اساس موقعیت محلی انجام می‌شود. بهترین تقسیم فوری انتخاب می‌شود و نه بهترین تقسیم سراسری. همچنین، پس از انجام تقسیم، هرگز بازنگری صورت نمی‌گیرد.
اساس منطقی این روش، بدین صورت است که فرضیه‌های ساده‌تر را ترجیح می‌دهد و فرض می‌کند ویژگی‌ها می‌توانند به طور مستقل برای تقسیم ارزیابی شوند.

- (ج): فرض کنید n مثال آموزش و m ویژگی داشته باشیم. در هر گره، ارزیابی تمام ویژگی‌ها به $O(m)$ زمان نیاز دارد. برای هر ویژگی، محاسبه سود اطلاعات بر اساس نوع آن متفاوت است. اگر دسته‌ای باشد، شمارش فراوانی کلاس‌ها $O(n)$ طول می‌کشد و اگر عددی پیوسته باشد، $O(n.log\ n)$ برای مرتب‌سازی ضروری است. مجموعاً برای هر گره $O(m.n.log\ n)$ طول می‌کشد. اگر تعداد گره‌های داخلی T باشد، پیچیدگی به صورت $O(T.m.n.log\ n)$ خواهد بود اما چون عموماً T خیلی کمتر از n است، پس پیچیدگی کل را می‌توانیم به صورت $O(m.n.log\ n)$ درنظر بگیریم.

تمرین ۴:

- (الف) بیش‌برازش زمانی رخ می‌دهد که یک مدل یادگیری ماشین نه تنها الگوهای اساسی در داده‌های آموزش، بلکه نویز، نوسانات تصادفی و جزئیات نامربوط را نیز یاد می‌گیرد. این منجر به مدلی می‌شود که روی داده‌های آموزش عملکرد استثنایی دارد اما روی داده‌های دیده‌نشده (داده آزمون یا داده دنیای واقعی) ضعیف عمل می‌کند. در درخت‌های تصمیم، شاخه‌های بیش از حد که نمونه‌های آموزشی خاص را ثبت می‌کنند، گره‌های برگ با نمونه‌های بسیار کم (اغلب فقط ۱-۲ نمونه) و ثبت نویز به عنوان الگو به دلیل عمق زیاد، می‌توانند از نشانه‌های بیش‌برازش باشند.

- (ب)هر مدل یادگیری ماشین، به منظور خاصی طراحی می‌گردد. جهت رسیدن به هدف مربوطه، مدل نیاز به آموزش دارد تا یاد بگیرد دقیقاً چه کاری را انجام دهد. بدین جهت، نیاز به داده‌های آموزشی داریم. همچنین، باید اطمینان پیدا کنیم که مدل طراحی شده، خاصیت تعمیم‌پذیری را رعایت می‌کند و برای داده‌های دیده‌نشده نیز به خوبی عمل می‌کند. به علاوه، برای ما بسیار مهم است که مدل صرفاً داده آموزش را حفظ نکرده باشد بلکه الگوی آنها را یاد گرفته است. پس به مجموعه داده تست، برای ارزیابی عملکرد مدل نیاز داریم. نکته مهم این است که تقسیم‌بندی به خودی خود از بیش‌برازش جلوگیری نمی‌کند و فقط به تشخیص آن کمک می‌کند. بدین منظور، می‌توان از چندین تکنیک پیشگیری موثر استفاده کرد که در درخت‌های تصمیم نیز قابل استفاده هستند:

  - هرس کردن به جهت کنترل خطای کاهش‌یافته، پیچیدگی و هزینه
  - منظم‌سازی با لحاظ کردن حداقل نمونه در هر برگ یا حداکثر عمق
  - معیارهای $Early\ Stopping$
  - بهره‌گیری از $Ensemble\ Method$ها مانند $Random\ Forest$ و $Bagging$

- (ج) نحوه عملکرد راهبردهای مختلف هرس کردن را توضیح می‌دهیم.

  - $Reduced-Error\ Pruning$: رشد کامل درخت تا زمانی که همه برگ‌ها خالص شوند یا معیار حداقل برآورده شود انجام می‌شود. برای هر گره غیربرگ، موقتاً هرس می‌کند تا به برگ با کلاس اکثریت تبدیل شود. ارزیابی دقت روی مجموعه $validation$انجام می‌شود و در صورتی که دقت کاهش نیابد، هرس را نگه می‌دارد. این روند، تا زمانی که بهبودی حاصل نشود، تکرار می‌شود. پیاده‌سازی این روش، ساده است و معمولاً به بهبود $generalization$ در مدل کمک می‌کند؛ اما از نظر محاسباتی می‌تواند پرهزینه باشد. همچنین، نیاز به مجموعه $validation$ جداگانه داریم و رویکرد حریصانه این روش، می‌تواند باعث شود که هرس‌های بهتر را از دست بدهیم.
  - $Rule\ Post-Pruning$: درخت به مجموعه معادل قواعد تبدیل می‌شود به طوری که هر مسیر از ریشه به برگ، یک قاعده محسوب می‌شود. هر قاعده را مستقل هرس می‌کنیم. $antecedents$ را حذف می‌کنیم و ارزیابی دقت روی مجموعه $validation$انجام می‌شود و در صورتی که دقت کاهش نیابد، هرس را نگه می‌دارد. قواعد را بر اساس دقت تخمینی مرتب می‌کنیم و سپس، به ترتیب بر اساس بهترین بودن، اعمال می‌کنیم. در این روش، ترتیب قوانین بسیار مهم است. اگرچه تفسیرپذیری بهتری دارد و می‌تواند قواعدی که هم‌پوشانی دارند را مدیریت کند، اما به دلیل پیاده‌سازی سطوح مختلف هرس به ازای هر قاعده، پیاده‌سازی پیچیده‌تری دارد و ممکن است باعث شود، بخشی از آثار مثبت مرتبط با ساختار درخت را از دست بدهیم.
  - $Cost-Complexity\ Pruning$: به غیر از دو راهبرد قبل که در سوال ذکر شده بود، این رویکرد نیز می‌توان مد نظر قرار گیرد. فرمول کلیدی آن، بدین صورت است:

$$R_\alpha(T) = R(T) + \alpha|\tilde{T}|$$

برای درخت T، عبارت $R(T)$ برابر $Misclassification\ rate$ است، $|(\tilde{T})|$ برابر تعداد گره‌های برگ و α، برابر پارامتر پیچیدگی هستند. با شروع از درخت پُر، دنباله درخت‌های بعدی را به صورت

زیرمجموعه درخت قبلی باشد، تولید می‌کند. در حقیقت، برای هر درخت، زیردرخت مینیمم‌کننده عبارت R_α را می‌یابد. در نهایت، بهترین درخت، با استفاده از $cross-validation$ تعیین می‌گردد.

- (د):
  - (۱) زمان آموزش کوتاه معمولاً منجر به کم‌برازش می‌شود، نه بیش‌برازش. بنابراین نادرست است.
  - (۲) عموماً درخت‌های کوچک‌تر تعمیم‌پذیری بهتری دارند. بنابراین، این گزاره را با احتیاط، درست درنظر می‌گیریم.
  - (۳) مجموعه آموزش تعیین می‌کند که چه الگوهایی یاد گرفته می‌شود، ساختار درخت و تقسیم‌ها چگونه باشد و چطور ثبت توزیع واقعی داده، صورت گیرد. بنابراین، درست است.
  - (۴) نظریه اطلاعات به انتخاب تقسیم‌ها کمک می‌کند اما نمی‌تواند برای داده ناکافی یا سوگیری نمونه‌گیری، کارایی داشته باشد و جبران کند. بنابراین نادرست است.

- (ه): $Dropout$، یک تکنیک منظم‌سازی است که در طول آموزش استفاده می‌شود. به صورت تصادفی واحدهای عصبی (نورون‌ها) را در هر مرحله آموزش غیرفعال می‌کند. همچنین، از وابستگی بیش از حد نورون‌ها جلوگیری می‌کند. در درخت تصمیم بر مبنای هرس، تکنیک پس‌پردازشی برای ساده‌سازی مدل صورت می‌گیرد؛ بدین صورت که شاخه‌های غیرضروری یا کم‌اهمیت درخت را حذف می‌کند، از بیش‌برازش جلوگیری کرده و تعمیم‌پذیری را بهبود می‌بخشد. چند شباهت مهم، بین این دو رویکرد برقرار است.
  - هدف مشترک هر دو، جلوگیری از بیش‌برازش است تا وابستگی‌های خاص کاهش یابد.
  - هر دو، $Robustness$ را مدنظر قرار می‌دهند.
  - هر دو، به دنبال بهبود تعمیم‌پذیری هستند. در هرس کردن، درخت‌های مختلف (زیردرخت‌ها) در فرآیند هرس ارزیابی می‌شوند و درخت نهایی نماینده‌ای از فضای جست‌وجو می‌باشد. در $Dropout$، در هر مرحله آموزش، یک معماری شبکه متفاوت آموزش می‌بیند و در تست، از میانگین تمام این معماری‌ها استفاده می‌شود.

  اگر مقایسه دقیقی داشته باشیم، می‌توانیم تفاوت‌های این دو رویکرد را بدین صورت بیان کنیم:
  - زمان اعمال روش $Dropout$، در حین آموزش است و در هرس کردن، پس از آموزش است.
  - نحوه اجرای الگوریتم در $Dropout$، تصادفی و پویاست و در هرس کردن، قطعی و استاتیک است.
  - $Dropout$ موقتاً ساختار را تغییر می‌دهد، اما در هرس کردن، تغییر ساختار، دائمی است.

  به طور کلی می‌توان گفت هدف هر دو ساده کردن مدل، جهت بهبود تعمیم پذیری است. یکی با غیرفعال کردن موقت اجزا و دیگری، با هرس کردن، این هدف را دنبال می‌کنند.

تمرین ۵:

- (الف): در توقف زودهنگام، توقف رشد درخت هنگامی که معیارها برآورده شوند، رخ می‌دهد. مهم‌ترین معایب آن:
  - نمی‌تواند ببیند که یک تقسیم "بد" ممکن است به تقسیم‌های خوب بعدی منجر شود.
  - تصمیم‌های حریصانه ممکن از یافتن بهینه سراسری جلوگیری کنند.
  - تنظیم آستانه یا تعیین معیارهای توقف بهینه دشوار است.
  - تعامل بین بخش‌های مختلف درخت را در نظر نمی‌گیرد.

درحالي که اگر بگذاریم درخت کامل رشد کند و سپس، هرس را انجام دهیم:

- کل ساختار را قبل از تصمیم‌گیری هرس می‌بینیم.
- معمولاً از پیش‌هرس بهتر عمل می‌کنیم.
- می‌توانیم با انعطاف، شاخه‌های مختلف را به سطوح مختلف هرس کنیم.

- (ب): اگر مجموعه داده موجود را به سه بخش $train$، بخش $validation$ و بخش $test$ تقسیم کنیم، کاربرد آنها بدین صورت خواهد بود:

- $train$: هدف، ساخت درخت تصمیم اولیه است. برای هر $splitting$، معیار $inforamtion\ gain$ را محاسبه می‌کنیم. بر اساس کلاس اکثریت، پیش‌بینی گره را مشخص می‌کنیم و رشد درخت را تا رسیدن به معیارت توقف، ادامه می‌دهیم. این مجموعه، معمولاً ۶۰ الی ۷۰ درصد کل داده‌ها را شامل می‌شود.
- $validation$: برای هرس و اعتبارسنجی به کار می‌رود. هدف، انتخاب درخت هرس‌شده بهینه از کاندیدهاست. سطوح مختلف هرس را ارزیابی می‌کنیم. انتخاب پارامتر پیچیدگی را انجام می‌دهیم. همچنین، می‌توانیم عمکرد حالات هرس‌شده و هرس‌نشده را مقایسه کنیم. این مجموعه، معمولاً ۱۵ الی ۲۰ درصد کل داده‌ها را شامل می‌شود.
- $test$: هدف، انجام آزمون و ارزیابی نهایی بی‌طرفانه است. میزان تعمیم‌پذیری مدل را ارزیابی می‌کنیم و معیارهای ارزیابی نهایی را گزارش می‌کنیم. این مجموعه هم معمولاً ۱۵ الی ۲۰ درصد کل داده‌ها را شامل می‌شود.

بدین ترتیب، می‌توان در بخش $train$ داده‌ها به ساخت درخت تصمیم و بخش $validation$ داده‌ها، به انتخاب یک درخت تصمیم از میان مجموعه‌ای از نامزدهای هرس پرداخت.

تمرین ۶:

- (الف): ابتدا به ناخالصی گره ریشه می‌پردازیم. چون احتمال همه کلاسها یکسان است:

$$p(c_i) = \frac{1}{4}, \quad i = 1, 2, 3, 4$$

بنابراین:

$$Misclassification\ rate : p_{misclass} = 1 - max_i p(c_i) = 1 - \frac{1}{4} = \frac{3}{4}$$

$$Gini\ Index : p_{misclass} = 1 - \Sigma_{i=1}^{4} p(c_i)^2 = 1 - 4(\frac{1}{4})^2 = \frac{3}{4}$$

بر اساس شکل موجود در سوال، $splitting$ چپ را ابتدا بررسی می‌کنیم. به دلیل تقسیم متقارن هر گره به دو گروه، برای هر بچه، بر اساس $Misclassification$، داریم:

$$y_{misclass_i} = 1 - 0.5 = 0.5 \implies \frac{1}{2}\Sigma_i^2(0.5 + 0.5) = 0.5$$

$$\Delta_{misclass} = 0.75 - 0.5 = 0.25$$

همچنین طبق معیار $Gini$، برای هر بچه می‌توان نوشت:

$$y_{misclass_i} = 1 - (0.5^2 + 0.5^2) = 0.5 \implies \frac{1}{2}\Sigma_i^2(0.5 + 0.5) = 0.5$$

$$\Delta_{misclass} = 0.75 - 0.5 = 0.25$$

در ادامه، $splitting$ راست را بررسی می‌کنیم. برای هر بچه، یک کلاس با احتمال $\frac{1}{2}$ و دو کلاس با احتمال $\frac{1}{4}$، داریم. بر اساس $Misclassification$، داریم:

$$y_{misclass_i} = 1 - 0.5 = 0.5 \implies \frac{1}{2}\Sigma_i^2(0.5 + 0.5) = 0.5$$

$$\Delta_{misclass} = 0.75 - 0.5 = 0.25$$

همچنین طبق معیار $Gini$، برای هر بچه می‌توان نوشت:

$$y_{misclass_i} = 1 - (0.25^2 + 0.25^2 + 0.5^2) = 0.625 \implies \frac{1}{2}\Sigma_i^2(0.625 + 0.625) = 0.625$$

$$\Delta_{misclass} = 0.75 - 0.625 = 0.125$$

- (ب): برای معیار $Misclassification$، دو $splitting$، شرایط مشابهی دارند؛ اما، معیار $Gini$، به دلیل کمتر بودن $splitting$ چپ، آن را ترجیح می‌دهد. چون گره‌های بچه‌ی $pure$تری ایجاد می‌کند. با توجه به نتایج حاصل‌شده، می‌توان دریافت که $Gini$ در مقایسه با معیار دیگر، حساسیت بیشتری نسبت به ترکیب کلاس‌ها دارد.

- (ج): این دو ماتریس را در نظر بگیرید:

$$c_L = \begin{bmatrix} 0 & 10 & 1 & 1 \\ 10 & 0 & 1 & 1 \\ 1 & 1 & 0 & 10 \\ 1 & 1 & 10 & 0 \end{bmatrix}, \quad c_R = \begin{bmatrix} 0 & 1 & 10 & 10 \\ 1 & 0 & 10 & 10 \\ 10 & 10 & 0 & 1 \\ 10 & 10 & 1 & 0 \end{bmatrix}$$

c_R، به گونه‌ای طراحی شده است که تقسیم سمت چپ، هزینه گزافی برای اشتباهاتش داشته باشد و تقسیم سمت راست ترجیح داده شود. این در حالی است که ماتریس c_L، هزینه مورد انتظار پایینی دارد و تقسیم چپ را ترجیح می‌دهد.

تمرین ۷:

- (لف): مجموعه آموزش بدین صورت است:

id	Color	Size	Points	Eatability
1	red	small	yes	toxic
2	brown	small	no	eatable
3	brown	large	yes	eatable
4	green	small	no	eatable
5	red	large	no	eatable

در شکل موجود در درخت تصمیم یک‌سطحی برای طبقه‌بندی قارچ‌ها، تقسیم داده‌ها بر اساس ویژگی $size$ صورت می‌گیرد و لیبل، یکی از مقادیر $toxic$ یا $eatable$ است. بر اساس تابع هزینه معرفی‌شده در صورت سوال، به ارائه راه‌حل می‌پردازیم. بر اساس $size$ تقسیم‌بندی می‌کنیم:

- $small$: بیشتر کلاس‌ها $eatable$ هستند. پس طبق اکثریت، لیبل را $eatable$ در نظر می‌گیریم و خطای طبقه‌بندی اشتباه را برای مثال شماره ۱ ، برابر ۱ در نظر می‌گیریم.

- $large$: بیشتر کلاس‌ها $eatable$ هستند و طبق اکثریت، لیبل را هم $eatable$ در نظر می‌گیریم. خطای طبقه‌بندی اشتباه نداریم.

بدین ترتیب هزینه کل طبقه‌بندی اشتباه برابر ۱ خواهد بود.

- (ب) این‌گونه در نظر می‌گیریم که اگر اشتباهاً کلاس $toxic$ را $eatable$ در نظر بگیریم، به دلیل خطرناک بودن، هزینه بالایی به همراه داشته باشد. بدین ترتیب اگر احیاناً کلاس $eatable$ را $toxic$ بگیریم، هزینه ۱ و اگر کلاس $toxic$ را $eatable$ بگیریم، هزینه ۱۰۰ در نظر می‌گیریم.

  - برای قارچ‌های $small$، فقط یک مورد $toxic$ است و دو مورد دیگر $eatable$ هستند. اگر لیبل را $toxic$ بگیریم، طبقه‌بندی اشتباه دو موردی که این لیبل را ندارند، مجموعاً هزینه ۲ دارد. در حالی که اگر لیبل را $eatable$ بگیریم، هزینه طبقه‌بندی اشتباه موردی که این لیبل را ندارد، ۱۰۰ خواهد بود که بسیار بالاست. پس لیبل را $toxic$می‌گیریم.
  - برای قارچ‌های $large$، مورد $toxic$ نداریم. اگر لیبل را $toxic$ بگیریم، طبقه‌بندی اشتباه دو موردی که این لیبل را ندارند، مجموعاً هزینه ۲ دارد. در حالی که اگر لیبل را $eatable$ بگیریم، هزینه طبقه‌بندی اشتباه نداریم. پس لیبل را $eatable$ می‌گیریم.

بدین ترتیب:

- قارچ‌های $small$، لیبل $toxic$ می‌گیرند.
- قارچ‌های $large$، لیبل $eatable$ می‌گیرند.
- هیچ قارچ $toxic$، لیبل $eatable$ نمی‌گیرد.

تمرین ۸:

- (الف): این تقسیم آستانه‌های مطرح‌شده در صورت سوال $axis - aligned$ هستند.

  - تقسیم به فرم $x_1 < c$، یک برش عمودی است.
  - تقسیم به فرم $x_2 < c$، یک برش افقی است.

هر تقسیم، صفحه را فقط تحت یک محور تقسیم می‌کند. البته، الگوی صفحه شطرنجی، رنگ‌ها در هر واحد به طور متناوب تغییر می‌کنند و باید هر ۱۶ مربع یکه به شکل صحیح تفکیک شوند. تحت هر محور ۳ برش داریم که ۴ ناحیه ایجاد می‌کند. پس تعداد کل نواحی مجزای ایجاد شده، برابر $4 \times 4 = 16$ هستند. جهت تفکیک این ۱۶ مربع، ۳ برش در جهت محور اول و ۳ برش در جهت محور دوم، همه مربع‌ها پوشش داده می‌شوند. بنابراین حداقل عمق برابر ۶ خواهد بود.

- (ب): اگر فقط وابسته به یک محور باشد، می‌توانیم از زوج یا فرد بودن استفاده کنیم، درست مشابه آنچه الگوی شطرنجی به آن وابسته است. بنابراین حداقل عمق برابر ۲ خواهد بود.

- (ج): اجازه استفاده از هر ویژگی مشترک دو محور داده می‌شود. این عبارت را در نظر بگیرید:

$$(\lfloor x_1 \rfloor + \lfloor x_2 \rfloor) mod\, 2 = 0$$

کل مسئله با یک شرط بولی حل می‌شود؛ بنابراین، حداقل عمق درخت را برابر ۱ است.

تمرین ۹:

$AdaBoost$ خطای آموزشی را به‌صورت نمایی کاهش می‌دهد.

$$TrainErr(H) \leq \prod_{t=1}^{T} Z_t$$

$$Z_t = 2\sqrt{\epsilon_t(1-\epsilon_t)}$$

چون $\epsilon_t \leq \frac{1}{2} - \gamma$:

$$Z_t \leq \sqrt{1-4\gamma^2}$$

با استفاده از نامساوی زیر:

$$\sqrt{1-x} \leq e^{-\frac{x}{2}}$$

می‌توان نوشت:

$$Z_t \leq e^{-2\gamma^2}$$

کران نهایی خطا را تعیین می‌کنیم:

$$TrainErr(H) \leq e^{-2\gamma^2 T}$$

صفرشدن خطا را بررسی می‌کنیم. اگر:

$$e^{-2\gamma^2 T} < \frac{1}{m}$$

آنگاه هیچ نمونه‌ای اشتباه طبقه‌بندی نمی‌شود. با گرفتن لگاریتم:

$$T > \frac{ln\, m}{2\gamma^2}$$

حکم ثابت می‌گردد.

تمرین ۱۰:

احتمال حذف یک نمونه خاص، برابر است با:

$$(1 - 1\frac{1}{m})^m$$

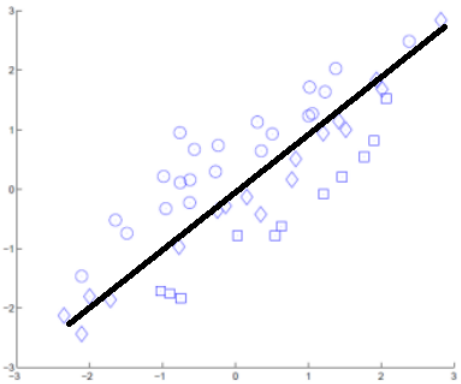
کسر مورد انتظار نمونه‌های حذف شده برابر است با:

$$\mathbb{E} = (1 - 1\frac{1}{m})^m$$

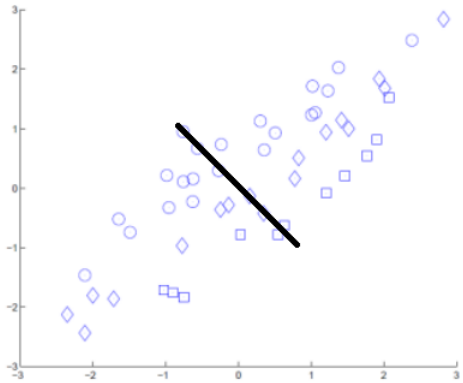
اگر m به بی‌نهایت میل کند، کسر فوق برابر e^{-1} خواهد بود. یعنی حدود ۳۷ درصد داده‌ها.

تمرین ۱۱:

هدف روش PCA این است که واریانس داده پراجکت‌شده را بیشینه کند و هدف روش فیشر، یافتن جهت پراجکشنی است که نسبت واریانس بین کلاسی به واریانس درون کلاسی داده پراجکت‌شده را بیشینه کند. شکل اول مرتبط با نتیجه روش PCA است:



شکل دوم مرتبط با روش فیشر است:



برای روش فیشر در حالت چند کلاسه (حالت خاص در اینجا ۳ کلاسه)، داده‌ها دو بعدی باقی می‌مانند اما محورها عوض می‌شوند. به مراکز کلاس‌ها توجه می‌کند. توجه می‌کنیم که در چه جهتی فاصله بین این سه میانگین بیشینه است. محور اول را طوری تعیین می‌کند که یک کلاس تقریباً کامل جدا شود و دو کلاس دیگر تقریباً در هم تنیده باشند. سپس به سراغ دو کلاس باقی‌مانده رفته و آنها را با محور دیگر تفکیک می‌کنیم. بدین ترتیب چون دایره‌ها، خالصانه‌تر جدا شده‌اند، مشابه آنچه در شکل فوق ترسیم شده است، با یک محور آنها را نسبت به باقی داده‌ها جدا می‌کنیم و سپس با یک محور مربع‌ها و لوزی‌ها را از هم تفکیک می‌کنیم.

تمرین ۱۲:

- (الف) ابتدا میانگین را محاسبه می‌کنیم:

$$\mu = \frac{1}{3}[\begin{bmatrix}-1\\-1\end{bmatrix} + \begin{bmatrix}0\\0\end{bmatrix} + \begin{bmatrix}1\\1\end{bmatrix}] = \begin{bmatrix}0\\0\end{bmatrix}$$

ماتریس $center$ شده را محاسبه می‌کنیم:

$$Z = D - \mu.1^T = \begin{bmatrix}-1 & 1\\0 & 0\\1 & 1\end{bmatrix}$$

ماتریس کوواریانس را محاسبه می‌کنیم:

$$\Sigma = \frac{1}{n-1}Z^TZ = \frac{1}{2}\begin{bmatrix}-1 & 0 & 1\\-1 & 0 & 1\end{bmatrix}\begin{bmatrix}-1 & 1\\0 & 0\\1 & 1\end{bmatrix} = \frac{1}{2}\begin{bmatrix}2 & 2\\2 & 2\end{bmatrix} = \begin{bmatrix}1 & 1\\1 & 1\end{bmatrix}$$

بردارهای ویژه و مقادیر ویژه آن را محاسبه می‌کنیم. باید $\Sigma v = \lambda v$ را حل کنیم. پس قرار می‌دهیم:

$$det(\Sigma - \lambda I) = \begin{bmatrix} 1-\lambda & 1 \\ 1 & 1-\lambda \end{bmatrix} = (1-\lambda)^2 - 1 = \lambda^2 - 2\lambda = \lambda(\lambda - 2) = 0$$

مقادیر ویژه، برابر $\lambda_1 = 2$ و $\lambda_2 = 0$ هستند. برای $\lambda_1 = 2$:

$$\begin{bmatrix} 1-2 & 1 \\ 1 & 1-2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

با توجه به معادله $-v_1 + v_2 = 0$ می‌توان دریافت: $v_1 = v_2$. بردار ویژه یکه را تعیین می‌کنیم:

$$v_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$$

این بردار، همان مولفه اصلی اول است که جهت آن در امتداد خط $y = x$ است.

- (ب) با توجه به فرمول $a = v_1^T x$، برای هر نقطه، مختصات جدید را محاسبه می‌کنیم:

$$P_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix} \implies a_1 = v_1^T P_1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} -1 \\ -1 \end{bmatrix} = -\frac{1}{\sqrt{2}} - \frac{1}{\sqrt{2}} = -\sqrt{2}$$

$$P_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \implies a_2 = v_1^T P_2 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \end{bmatrix} = 0$$

$$P_3 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \implies a_3 = v_1^T P_3 = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} = \sqrt{2}$$

پس مختصات تصویرشده، بدین صورت هستند:

$$a_1 = -\sqrt{2}, \;\; a_2 = 0, \;\; a_3 = \sqrt{2}$$

واریانس داده تصویرشده را حساب می‌کنیم:

$$\mu_P = \frac{1}{3}[\sqrt{2} + 0 - \sqrt{2}] = 0$$

$$\sigma_P^2 = \frac{1}{2}[(\sqrt{2} - 0)^2 + (0 - 0)^2 + (\sqrt{2} - 0)^2] = \frac{1}{2}[2 + 0 + 2] = 2$$

می‌بینیم که $\sigma_P^2 = \lambda_1$ برقرار است.

- (ج) بر اساس $\hat{x}_i = a_i v_1$، عملیات $reconstruction$ را انجام می‌دهیم:

$$\hat{P}_1 = a_1 v_1 = -\sqrt{2} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$\hat{P}_2 = a_2 v_1 = 0 \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\hat{P}_3 = a_3 v_1 = \sqrt{2} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

چون دقیقاً همان نقاط اولیه تولید شده‌اند، خطای $reconstruction$ برابر صفر است.

تمرین ۱۳:

تمامی این روش‌ها هدفی مشترک دارند و آن، نمایش داده‌های پربُعد با تعداد کمی متغیر پنهان است. تفاوت آن‌ها در مواردی مانند قیود ریاضی، نوع تابع خطا، قابلیت تفسیر و نوع داده (ماتریس یا تانسور) می‌باشد که در ادامه، درباره آنها توضیح می‌دهیم.

- روش SVD، پایه جبری تمامی این روش‌هاست که برای ماتریس $X \in \mathbb{R}^{n\times d}$دارای چنین تجزیه‌ای است:

$$X = U\Sigma V^T$$

در این تجزیه، Uو V ماتریس‌های متعامد هستند که به ترتیب، مقادیر تکین چپ و راست ماتریس را نشان می‌دهند و Σ بیانگر مقادیر تکین است. مسئله بهینه‌سازی آن، بهترین تقریب مرتبه پایین با رتبه kرا بدین صورت می‌یابد :

$$min \ \ ||X - Y||_F^2$$

$$s.t. \ \ rank(Y) = k$$

این روش، پایه روش PCA است. همچنین، می‌توان مشاهده نمود که این روش، فاقد مدل احتمالاتی بوده و اساساً جبری است.

- هدف روش PCA، بیشینه‌سازی واریانس است. اگر داده X یک داده $center$شده باشد، مسئله بهینه‌سازی آن، بدین صورت مطرح می‌گردد:

$$max_W \ \ Var(XW)$$

$$s.t. \ \ W^TW = I$$

می‌توان به صورت معادل، مسئله را به این صورت نیز مطرح نمود:

$$max_{W,Z} \ \ ||X - ZW^T||_F^2$$

$$s.t. \ \ W^TW = I$$

در حقیقت، روش PCA، معادل روش SVD است که بر ماتریس داده $center$ شده اجرا می‌شود. مولفه‌های اصلی، معادل بردارهای تکین راست هستند و واریانس توصیف‌شده، معادل مربع مقادیر تکین است. اگرچه این روش بر اساس واریانس، بهینه عمل می‌کند، اما به نسبت تفسیرپذیری پایینی دارد. همچنین این روش، برخلاف $PPCA$، فاقد نگاه احتمالاتی است.

- روش $PPCA$، نسخه احتمالاتی روش PCA است که به شکل یک مدل متغیر پنهان به آن نگاه می‌کند. می‌توان بر اساس مدل‌های مولد، چنین توصیفی از این روش ارائه نمود:

$$x = Wz + \mu + \epsilon \quad , Z \sim \mathcal{N}(0, I) \quad , \epsilon \sim \mathcal{N}(0, \sigma^2 I)$$

مسئله بهینه‌سازی آن بدین صورت مطرح می‌گردد:

$$max_W \ log \ p(X|W, \sigma^2)$$

در حقیقت، PCA حالت خاصی از $PPCA$ است؛ زیرا در حقیقت، هنگامی که $\sigma^2 \to 0$، روش $PPCA$ به PCA تبدیل می‌شود.همچنین این روش، شرایط داده‌های گمشده را لحاظ می‌کند و یک بسط بیزی ارائه می‌دهد تا بتوانیم میزان عدم قطعیت را تخمین بزنیم. می‌توان جهت تبدیل PCA به $PPCA$، جواب PCA را به عنوان برآورد درست‌نمایی بیشینه $PPCA$ در نظر گرفت.

- روش NMF، یک تجزیه قابل تفسیر ارائه می‌دهد که به فرم زیر است:

$$X \approx WH$$
$$s.t.\ \ W \geq 0,\ \ H \geq 0$$

مسئله بهینه‌سازی آن بدین صورت مطرح می‌گردد:

$$min_{W,H\geq 0}\ ||X - WH||_F^2$$

همچنین، می‌توان از همگرایی KL استفاده نمود.
این روش چند تفاوت مهم با روش‌های PCA/SVD دارد:

  - خاصیت متعامدی ندارد.
  - عناصر منفی امکان‌پذیر نیستند.
  - تجزیه یکتا نیست.
  - برخلاف دو روش دیگر، تفسیرپذیری بسیاربالایی دارد.
  - PCA، جهت‌های متعامد حداکثر واریانس را می‌یابد و NMF، بردارهای پایه غیرمنفی (مخروط محدب) را می‌یابد که در حقیقت، جوابهایش در همان فضای PCA قرار دارند اما به ربع غیرمنفی محدود شده‌اند.

نکته مهم این است که NMF، بهینگی را فدای تفسیرپذیری می‌کند. دارای اجزای افزایشی است و مناسب داده‌های واقعی مانند تصویر و متن می‌باشد. برای داده غیرمنفی، می‌توان جواب PCA را با قیود NMF، با بهینه‌سازی متناوب و قیود تعامد اضافی تقریب زد.

- تمامی روش‌های مذکور را می‌توان در قالب یک عبارت واحد معرفی نمود:

$$min_{W,H}\ ||X - WH|| + \text{قیود}$$

قیود و تابع هدف برای هر روش بدین صورت مطرح می‌گردد:

  - SVD: قیود آن، متعامد بودن و تابع هدف آن، کمینه‌سازی نرم فربنیوس است.
  - PCA: قیود آن، متعامد بودن و $center$ بودن داده و تابع هدف آن، بیشینه‌سازی واریانس است.
  - $PPCA$: قیود آن، رعایت $prior$ گاوسی و تابع هدف آن، بیشینه‌سازی درست‌نمایی است.
  - NMF: قیود آن، نامنفی بودن و تابع هدف آن، کمینه‌سازی خطای مبتنی بر اجزا می‌باشد.

- روش تجزیه تاکر در داده‌های چندبعدی و تانسوری کاربرد زیادی دارد. در حقیقت، این تجزیه کمک می‌کند تا بتوانیم تجزیه و عامل‌بندی را از ماتریس‌ها به تانسورها تعمیم دهیم. برای تانسور $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$، می‌توانیم چنین تجزیه‌ای ارائه کنیم:

$$\mathcal{X} \approx \mathcal{G} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_N U^{(N)}$$

در این تجزیه، $\mathcal{G}$ تانسور هسته یا همان تانسور مرکزی است و $U^{(i)}$ها عوامل ماتریسی هستند. با این روش، می‌توان تجزیه روش‌های SVD/PCA را به داده‌های مرتبه بالاتر تعمیم داد.

  - تاکر حالت خاص PCA برای برای ماتریس‌های ۲بعدی است که فقط حاصل‌ضرب‌های حالت ۱ و حالت ۲ را لحاظ می‌کند. تاکر PCA را به داده‌های چندظریقه تعمیم می‌دهد.
  - اگر $\mathcal{X}$ یک ماتریس باشد، تاکر به SVD/PCA تقلیل می‌یابد.
  - تاکر ساختار انعطاف‌پذیرتری نسبت به NMF ساده برای داده تانسوری ارائه می‌دهد. می‌توانیم با اعمال قیود غیرمنفی بر ماتریس‌های عامل، حالت خاصی را ایجاد کنیم که برای داده‌های چندطریقه غیرمنفی مانند داده‌های طیفی و تصاویر RGB مفید باشد.
  - نسخه احتمالاتی این روش، معادل نسخه تانسوری $PPCA$ است که مدل‌های تانسوری بیزی را لحاظ می‌کند تا متغیرهای پنهان $multilinear$ را مورد بررسی قرار دهد.