سهراب پیرهادی

مستندسازی پروژه پایانی درس داده کاوی پیشرفته

اسفند ۹۸

سوال اول)

توضیحات مربوط به سوالات این بخش، در فایل سورس کد به صورت متنی ارائه شده است.(برای مشاهده بهتر، فایلی که خروجی html از پیاده سازی است را مطالعه نمایید.)

سوال دوم)

الگوریتم Naïve Bayes

اگر با یک مساله classification طرف باشیم، یکی از روش‌های طبقه‌بندی Naive Bayes است. در بیشتر مواقع، زمانی که تعداد متغیرها کم ولی مشاهدات زیاد هستند الگوریتم بیز ساده برای تشخیص دسته‌ها مناسب است. پایه و اساس الگوریتم دسته‌بندی بیز، قضیه بیز است.این روش فرض می‌کند که بین پیش‌بین‌ها (Predictors)استقلال وجود دارد. در واقع، یک دسته‌بندی نایو بیز فرض می‌کند که یک ویژگی در یک کلاس، نامرتبط به دیگر موارد است .برای هر مجموعه داده بزرگی، ساخت یک مدل نایو بیز آسان است. نه تنها این مدل بسیار ساده است، بلکه بهتر از بسیاری از روش‌های پیچیده دسته‌بندی کار می‌کند .

در بیشتر تکنیک‌ها و مدل‌های بیز ساده از روش حداکثرسازی تابع Likelihood استفاده می‌شود. هرچند تکنیک دسته‌بند بیز ساده دارای فرضیات محدود و قابل دسترس است ولی به خوبی می‌تواند از عهده حل مسائل واقعی برآید و به عنوان یک رقیب برای روش‌های Random Forest محسوب شود. یکی از مزایای قابل توجه در دسته‌بند بیز ساده، امکان برآورد پارامترهای مدل با اندازه نمونه کوچک به عنوان مجموعه داده آموزشی (Training Data)است.

مدل سازی احتمالاتی:

اگر n متغیر ورودی داشته باشیم یعنی $\chi=(\chi_1 , \chi_2 , \ldots, \chi_n)$ و خروجی y از یک مجموعه K عضوی باشد، هدف از مدل‌سازی پیدا کردن احتمال مشروط هر کدام از این K دسته است یعنی:

$\rho (C_k \mid \chi_1 , \chi_2 , \ldots, \chi_n)$ طبق قانون بیز این احتمال برابر است با

$$p(C_k \mid \mathbf{x}) = \frac{p\left(C_k\,,\,\mathbf{x}\right)}{p(\mathbf{x})} \propto p\left(C_k\,,\,\mathbf{x}\right)$$

به عبارت دیگر احتمال مشروط (χ_1 , χ_2 ,..., χ_n | C_k) ρ به توزیع توأم X و C_k بستگی دارد. طبق قانون زنجیره‌ای این توزیع توأم برابر است با:

$$\begin{aligned}
p(C_k, x_1, \dots, x_n) &= p(x_1, \dots, x_n, C_k) \\
p(C_k, x_1, \dots, x_n) &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2, \dots, x_n, C_k) \\
p(C_k, x_1, \dots, x_n) &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2 \mid x_3, \dots, x_n, C_k)\, p(x_3, \dots, x_n, C_k) \\
p(C_k, x_1, \dots, x_n) &= \dots \\
p(C_k, x_1, \dots, x_n) &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2 \mid x_3, \dots, x_n, C_k) \dots p(x_{n-1} \mid x_n, C_k)\, p(x_n \mid C_k) p(C_k)
\end{aligned}$$

حال اگر فرض کنیم هر متغیری نسبت به متغیرهای دیگر به شرط دسته C_k مستقل است یعنی

ρ (χ_i | χ_{i+1} , …, χ_n , C_k) = ρ(χ_i | C_k) به نتیجه پایین می‌رسیم:

$$\begin{aligned}
p(C_k \mid x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\
p(C_k, x_1, \dots, x_n) &= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\
p(C_k, x_1, \dots, x_n) &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)
\end{aligned}$$

با نرمال‌سازی عبارت قبلی می‌توان توزیع احتمال مشروط را پیدا کرد، در معادله پایین

Z = ρ(χ) =$\sum_k$ ρ(Ck)ρ(X |Ck) همان ضریب نرمال‌سازی است:

$$p(C_k \mid x_1, \dots, x_n) = \frac{1}{Z} p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

اگر هدف پیدا کردن محتمل ترین دسته باشد، به ضریب نرمال‌سازی یعنی Z نیازی نیست:

$$\hat{y} = \underset{k \in \{1, \dots, K\}}{\operatorname{argmax}}\ p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)$$

تخمین پارامترها

برای مدلسازی دسته‌بندی‌کننده بیز ساده برای تمام K ها به تخمین $\rho(C_k)$ و $\rho(\chi_i|C_k)$ نیاز داریم . به سادگی با حساب درصد داده‌هایی که متعلق به کلاس C_k هستند بدست می‌آید.

برای بدست آوردن $\rho(\chi_i|C_k)$ راه‌های مختلفی وجود دارد، تخمین توزیع چند جمله‌ای یا توزیع طبیعی روش‌هایی متداول برای این کار هستند .در روش تخمین توزیع طبیعی، $\rho(\chi_i|C_k)$ را با یک توزیع طبیعی با میانگین $\mu_{i,k}$ و واریانس $\sigma^2_{i,k}$ تخمین می‌زنیم و $\mu_{i,k}$ و $\sigma^2_{i,k}$ را از طریق درست نمایی بیشینه بدست می‌آوریم:

$$p(x_i = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma^2_{i,k}}} \exp\left(-\frac{(v-\mu_{i,k})^2}{2\sigma^2_{i,k}}\right)$$

اگر χ_i گسسته باشد، توزیع $\rho(\chi_i = \upsilon|C_k)$ را می‌توان با یک توزیع چند جمله‌ای تخمین زد.

اگر مشاهدات و داده‌ها از نوع پیوسته باشند، از مدل احتمالی با توزیع گاوسی یا نرمال برای متغیرهای مربوط به شواهد می‌توانید استفاده کنید. در این حالت هر دسته یا گروه دارای توزیع گاوسی است. به این ترتیب اگر k دسته یا کلاس داشته باشیم می‌توانیم برای هر دسته میانگین و واریانس را محاسبه کرده و پارامترهای توزیع نرمال را برای آن‌ها برآورد کنیم. اگر μk میانگین و σ^2_k واریانس دسته kام یعنی Ckباشد، همچنین ν را مشاهدات حاصل از متغیرهای تصادفی Xدر نظر بگیریم،آنگاه از آنجایی که توزیع Xدر هر دسته گاوسی نرمال فرض شده است، خواهیم داشت:

$$p(x = v \mid C_k) = \frac{1}{\sqrt{2\pi\sigma^2_k}} e^{-\frac{(v-\mu_k)^2}{2\sigma^2_k}}$$

دسته بند بیز ساده برنولی(Bernoulli Naive Bayes)

در این مدل در حالت چند متغیره، فرض بر این است که وجود یا ناموجود بودن یک ویژگی در نظر گرفته شود. به این ترتیب مدل تابع درستنمایی متن براساس کلاس های مختلف Ck به شکل زیر نوشته می‌شود:

$$p(\mathrm{x} \mid C_k) = \prod_{i=1}^{n} p_{ki}^{x_i}(1 - p_{ki})^{(1-x_i)}$$

و منظور از p_{ki} احتمال تولید مشاهده xi از کلاس Ck است.

از مزایای این روش می‌توان به موارد زیر اشاره کرد :

- دسته‌بندی کردن داده‌های آزمایشی آسان و سریع است. همچنین زمانی که تعداد دسته‌ها از دو بیشتر باشد نیز عملکرد خوبی از خودش نشان می‌دهد.
- تا زمانی که شرط مستقل بودن برقرار باشد، یک دسته‌بندی‌کننده بیز ساده عملکرد بهتری نسبت به مدل‌های دیگر مانند رگرسیون لجستیک دارد و به حجم آموزش کمی نیاز دارد.
- در حالتی که ورودی‌هایمان دسته‌بندی شده باشند این روش عملکرد بهتری نسبت به حالی دارد که ورودی‌هایمان عدد باشند. برای حالتی که ورودی عدد باشد معمولاً فرض می‌شود که از توزیع نرمال پیروی می‌کنند. (که فرض قوی‌ای است)

علاوه بر مزایایی که این دسته‌بندی‌کننده دارد معایبی نیز دارد، از جمله:

- در صورتی که ورودی‌مان دسته‌بندی شده باشد و در مرحله یادگیری دسته‌ای وجود داشته باشد که دسته‌بندی‌کننده هیچ داده‌ای از آن دسته مشاهده نکرده باشد، دسته‌بندی‌کننده احتمالی برابر صفر برای آن دسته در نظر می‌گیرد و قادر به دسته‌بندی کردن نخواهد بود. برای حل این مشکل می‌توان از تکنیک‌های هموارسازی مانند تخمین‌گر لاپلاس استفاده کرد.
- یکی دیگر از معایب این دسته‌بندی‌کننده این است که دستیابی به شرط مستقل بودن در دنیای واقعی تقریباً غیرممکن است.

کاربردها:

- دسته‌بندی‌کننده متن :دسته‌بندی‌کننده‌های بیز ساده عموماً در دسته‌بندی متن کاربرد دارند و نسبت به روش‌های دیگر درصد موفقیت بیشتری در این زمینه دارند.

- فیلترینگ اسپم :یکی از معروف‌ترین کاربردهای این دسته‌بندی‌کننده فیلترینگ اسپم است. در این روش فیلترینگ از دسته‌بندی‌کننده بیز ساده برای شناسایی ایمیل‌های اسپم استفاده می‌شود. امروزه بسیاری از سرویس‌دهندگان پست‌های الکترونیک از فیلترینگ اسپم بیزی استفاده می‌کنند. این روش در نرم‌افزارهای فیلتر اسپم نیز استفاده می‌شود.
- سامانه توصیه‌گر :دسته‌بندی‌کننده بیز ساده به همراه پالایش گروهی سامانه توصیه‌گری را تشکیل می‌دهد که از تکنیک‌های یادگیری ماشین و داده‌کاوی برای فیلتر کردن اطلاعات دیده نشده و پیش‌بینی نظر یک کاربر در مورد اقلام مختلف استفاده می‌کند.
- تحلیل احساسات :از این دسته‌بندی‌کننده در تحلیل احساسات متون و نظرات مختلف (برای مثال در شبکه‌های اجتماعی) استفاده می‌شود.

الگوریتم Decision Tree

یکی از قدرتمندترین و پرکاربردترین الگوریتم‌های داده کاوی الگوریتم درخت تصمیم است ، که در انجام پروژه داده کاوی ازآن استفاده میشود. این الگوریتم برای کاوش کردن در داده‌ها و کشف دانش در آنها کاربرد دارد. اگر بخواهیم درخت تصمیم را به شکل تمثال توضیح دهم می‌توانیم بگوییم دقیقاً مانند درختی است که در آن نمونه‌ها را دسته بندی می‌کند که از ریشه به سمت پایین رشد کرده و در انتها به گره‌های برگ خواهد رسید. برای روشن شدن بیشتر این سه موضوع را می‌توان اضافه کرد:

- هر گره داخلی یا غیر برگ را با یک ویژگی مشخص می‌کنند در واقع این ویژگی سوالی را در رابطه با مثال ورودی مطرح می‌کند
- برگ‌های این درخت در واقع با یک کلاس و یک درصد جواب مشخص خواهد شد
- در هر گره داخلی به تعداد جوابهای ممکن با این سؤال شاخه وجود دارد که هر کدام آن‌ها با میزان یا مقدار آن جواب مشخص می‌شود
- از آنجایی که یک درخت به طور کلی از ریشه، شاخه‌ها و گره‌ها و برگ‌ها تشکیل شده است. درخت‌های تصمیم نیز همانند این ساختارها در خود دارد. در درخت تصمیم از گره‌ها که با دایره نشان داده می‌شود و شاخه‌ها که با پاره خط‌های اتصال بین گره‌ها مشخص می‌شود برای سادگی در رسم معمولاً از چپ به راست یا از بالا به پایین رسم کرده به طوری که ریشه در بالا قرار بگیرد.

- به گروه اول ریشه گفته می‌شود، در انتهای یک زنجیر که در واقع می‌تواند ریشه، شاخه یا گره باشد را برگ می‌نامیم گره‌ای که برگ نباشد را گره داخلی می گویند.
- از هر گره داخلی می‌توان دو یا چند شاخه منشعب شود. هر گره مربوط به یک خصوصیت معین است و شاخه‌های مرتبط به آن به معنای تازه‌ای از مقادیر است در واقع این بازه‌های مقادیر باید بخش‌های این الگوریتم داده‌ها را به مجموعه‌های مشخص تقسیم کند و هر مجموعه‌ای از مجموعه‌های گفته شده زیر مجموعه‌ای از داده‌های کم و بیش همگن می‌باشد که دارای ویژگی‌های قابل پیش بینی هستند

به طور خلاصه، درخت تصمیم نقشه‌ای از نتایج احتمالی یکسری از انتخاب‌ها یا گزینه‌های مرتبط بهم است به طوری که به یک فرد یا سازمان اجازه می‌دهد تا اقدامات محتمل را از لحاظ هزینه‌ها، احتمالات و مزایا بسنجد. از درخت تصمیم می‌توان یا برای پیشبرد اهداف و برنامه‌های شخصی و غیررسمی یا ترسیم الگوریتمی که بر اساس ریاضیات بهترین گزینه را پیش‌بینی می‌کند، استفاده کرد.

یک درخت تصمیم‌گیری به طور معمول با یک نود اولیه شروع می‌شود که پس از آن پیامدهای احتمالی به صورت شاخه‌هایی از آن منشعب شده و هر کدام از آن پیامدها به نودهای دیگری منجر شده که آن‌ها هم به نوبه خود شاخه‌هایی از احتمالات دیگر را ایجاد می‌کنند که این ساختار شاخه‌شاخه سرانجام به نموداری شبیه به یک درخت مبدل می‌شود. در درخت تصمیم‌گیری سه نوع Node یا گره مختلف وجود دارد که عبارتند از:

- نودهای تصادفی
- نوهای تصمیم گیری
- نودهای پایانی

نود تصادفی، که توسط یک دایره نشان داده می‌شود، نمایانگر احتمال وقوع یکسری نتایج خاص است، نود تصمیم‌گیری، که توسط یک مربع نشان داده می‌شود، تصمیمی که می‌توان اتخاذ کرد را نشان می‌دهد و همچنین نود پایانی نمایانگر پیامد نهایی یک مسیر تصمیم‌گیری خواهد بود.

الگوریتم ساخت Decision Tree :

مجموعه داده ها را با D نمایش میدهیم، یعنی $D = (x_1,y_1) ,\dots, (x_i,y_i) ,\dots, (x_n,y_n)$، به قسمی که

$x_i \in R^d$. درخت تصمیم سعی میکند به صورت بازگشتی داده ها را به قسمی از هم جدا کند که در هر گره متغیرهای مستقل y به هم نزدیک شده همسان شوند.هر گره زیر مجموعه ای از داده هاست که به صورت بازگشتی ساخته شده است. به طور دقیق تر

مجموع داده‌ها را با D نمایش می‌دهیم، یعنی $D = (x_1,y_1) ,\dots, (x_i,y_i) ,\dots, (x_n,y_n)$، به قسمی که

$x_i \in R^d$ و $y_i \in R$. درخت تصمیم‌گیری سعی می‌کند به صورت بازگشتی داده‌ها را به قسمی از هم جدا کند که در هر گره متغیرهای مستقل y به هم نزدیک شده همسان شوند . هر گره زیر مجموعه ای از داده هاست که به صورت بازگشتی ساخته شده‌است. به‌طور دقیقتر در گره m اگر داده ما Q باشد سعی میکنیم یک بعد از متغیرهایی وابسته را به همراه یک آستانه انتخاب کنیم و داده‌ها را برحسب این بعد و آستانه به دو نیم تقسیم کنیم، به قسمی که بطور متوسط در هر دو نیم متغیرهای مستقل یا y خیلی به هم نزدیک و همسان شده باشند. این بعد و آستانه را $\theta = (j , t_m)$ می‌نامیم. دامنه j برابر است با $[1,\dots,d]$ و t_m یک عدد صحیح است. Q برحسب $\theta = (j , t_m)$ به دو بخش $Q_{left}(\theta)$ و $Q_{right}(\theta)$ به شکل پایین تقسیم می‌شود:

$$Q_{left}(\theta) = \{(x_i, y_i) \in Q \mid x_{i,j} \leq t_m\}$$
$$Q_{right}(\theta) = \{(x_i, y_i) \in Q \mid x_{i,j} > t_m\}$$

حال سؤال اینجاست که کدام بعد از متغیرهای وابسته و چه آستانه‌ای را باید انتخاب کرد. به زبان ریاضی باید آن θ یی را انتخاب کرد که ناخالصی داده را کم کند. ناخالصی برحسب نوع مسئله تعریفی متفاوت خواهد داشت، مثلا اگر مسئله یک دسته‌بندی دوگانه است، ناخالصی می‌تواند آنتراپی داده باشد، کمترین ناخالصی زمانی است که هم $Q_{left}(\theta)$ و هم $Q_{right}(\theta)$ از یک دسته داشته باشند، یعنی در هر کدام از این دو گِرِه دو نوع دسته وجود نداشته باشد. برای رگرسیون این ناخالصی می‌تواند واریانس متغیر وابسته باشد. از آنجا که مقدار داده در $Q_{right}(\theta)$ و $Q_{left}(\theta)$ با هم متفاوت است میانگینی وزن‌دار از هر دو ناخالصی را به شکل پایین محاسبه می‌کنیم. در این معادله $N_m = |Q|$، $n_{left} = |Q_{left}(\theta)|$ و $n_{right} = |Q_{right}(\theta)|$

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$$

هدف در اینجا پیدا کردن آن θ یی است که ناخالصی را کمینه کند، یعنی $\theta^* = \text{argmin}_\theta\, G(Q,\theta)$. حال همین کار را به صورت بازگشتی برای $Q_{left}(\theta)$ و $Q_{right}(\theta)$ انجام می‌دهیم. بعضی از گره‌ها را باید به برگ

تبدیل کنیم، معیاری که برای تبدیل یک گره به برگ از آن استفاده می‌کنیم می‌تواند مقداری حداقلی برای N_m (تعداد داده در یک گره) و یا عمق درخت باشد به قسمی که اگر با دو نیم کردن گِره یکی از معیارها عوض شود، گره را به دو نیم نکرده آن را تبدیل به یک برگ میکنیم. معمولاً این دو پارامتر باعث تنظیم مدل (Regularization) می‌شوند . در ابتدای کار گره شامل تمام داده‌ها می‌شود یعنی $Q = D$.

کاربرد Decision Tree در حوزه Data Mining :

از درخت تصمیم می‌توان به منظور ایجاد مدل‌های پیش‌بینی خودکار استفاده کرد که در حوزه یادگیری ماشینی، استخراج داده و آمار کاربردی هستند. این روش که تحت عنوان Decision Tree Learning شناخته می‌شود، به بررسی مشاهدات در مورد یک آیتم به جهت پیش‌بینی مقدارش می‌پردازد و به طور کلی، در چنین درخت تصمیمی، نودها نشان‌دهنده دیتا هستند نه تصمیمات. این نوع درخت‌ها همچنین تحت عنوان Classification Treeنیز شناخته می‌شوند به طوری که هر شاخه در برگیرنده مجموعه‌ای از ویژگی‌ها یا قوانین طبقه‌بندی دیتا است و مرتبط با یک دسته خاص می‌باشد که در انتهای هر شاخه یافت می‌شود.

این دست قوانین که تحت عنوان Decision Rules شناخته می‌شوند قابل بیان به صورت جملات شرطی می‌باشند (مثلاً اگر شرایط ۱ و ۲ و ۳ محقق شوند، با قطعیت می‌توان گفت که X نتیجه‌ای همچون Y برخواهد گرداند.) هر مقدار داده اضافی به مدل کمک می‌کند تا دقیق‌تر پیش‌بینی کند که مسئله مورد نظر به کدام مجموعه از مقادیر متعلق می‌باشد و این در حالی است که از این اطلاعات بعداً می‌توان به عنوان ورودی در یک مدل تصمیم‌گیری بزرگ‌تر استفاده کرد.

درخت‌های تصمیم‌گیری که پیامدهای محتمل پی‌درپی و بی‌نهایت دارند، Regression Tree نامیده می‌شوند. به طور کلی، متدهای کاربردی در این حوزه به صورت زیر دسته‌بندی می‌شوند:

- Bagging :در این متد با نمونه‌سازی مجدد دیتای سورس، چندین درخت ایجاد شده سپس با برداشتی که از آن درختان می‌شود، تصمیم نهایی گرفته شده یا نتیجه نهایی به دست می‌آید.
- Random Forest:در این متد طبقه‌بندی از چندین درخت تشکیل شده که به منظور افزایش نرخ classification طراحی شده‌اند.
- Boosted :درخت‌هایی از این جنس می‌توانند برای رگرسیون مورد استفاده قرار گیرند.

- Rotation Forest :در این متد، همگی درخت‌ها توسط یک به اصطلاح Principal Component Analysis با استفاده از بخشی از داده‌های تصادفی آموزش داده می‌شوند.

درخت تصمیم‌گیری زمانی مطلوب تلقی می‌شود که نشان‌دهنده بیشترین دیتا با حداقل شاخه باشد و این در حالی است که الگوریتم‌هایی که برای ایجاد درخت‌های تصمیم‌گیری مطلوب طراحی شده‌اند شامل CART ، ASSISTANT، CLSو ID31415 می‌شوند. در واقع، هر کدام از این متدها باید تعیین کنند که بهترین راه برای تقسیم داده در هر شاخه کدام است.

مزایای Decision Tree

در میان متخصصین در صنایع مختلف، مدیران و حتی دولوپرها، درخت‌های تصمیم محبوب‌اند چرا که درک آن‌ها آسان بوده و به دیتای خیلی پیچیده و دقیقی احتیاج ندارند، می‌توان در صورت لزوم گزینه‌های جدیدی را به آن‌ها اضافه کرد، در انتخاب و پیدا کردن بهترین گزینه از میان گزینه‌های مختلف کارآمد هستند و همچنین با ابزارهای تصمیم‌گیری دیگر به خوبی سازگاری دارند.

با تمام این‌ها، درخت‌های تصمیم ممکن است گاهی به شدت پیچیده شوند! در چنین مواردی یک به اصطلاح Influence Diagramجمع و جورتر می‌تواند جایگزین بهتری برای درخت تصمیم باشد به طوری که این دست نمودارها توجه را به تصمیمات حساس، اطلاعات ورودی و اهداف محدود می‌کنند.

استفاده از درخت‌های تصمیم‌گیری دریادگیری ماشینی چندین مزیت عمده دارد من‌جمله هزینه یا بهای استفاده از درخت به منظور پیش‌بینی داده با اضافه کردن هر به اصطلاح Data Point کاهش می‌یابد و این در حالی است که از جمله دیگر مزایایش می‌توان به موارد زیر اشاره کرد:

- برای داده‌های طبقه‌بندی شده و عددی به خوبی پاسخگو است.
- می‌تواند مسائل با خروجی‌های متعدد را مدل‌سازی کند.
- می‌توان قابلیت اطمینان به درخت را مورد آزمایش و اندازه‌گیری قرار داد.
- صرف‌نظر از اینکه آیا فرضیات داده منبع را نقض می‌کنند یا خیر، این روش به نظر دقیق می‌رسد.

معایب Decision Tree :

- حین مواجه با داده‌های طبقه‌بندی شده با سطوح مختلف، داده‌های حاصله تحت‌تاثیر ویژگی‌ها یا صفاتی که بیشترین شاخه را دارند قرار می‌گیرد.
- در صورت رویارویی با پیامدهای نامطمئن و تعداد زیادی پیامد بهم مرتبط، محاسبات ممکن است خیلی پیچیده شود.
- ارتباطات بین نودها محدود به AND بوده حال آنکه یک Decision Graph این اجازه را به ما می‌دهند تا نودهایی داشته باشیم که با OR به یکدیگر متصل شده‌اند.

الگوریتم Random Forest

جنگل‌های تصادفی یک روش یادگیری ترکیبی برای دسته‌بندی می‌باشد، که بر اساس ساختاری متشکل از شمار بسیاری درخت تصمیم، بر روی زمان آموزش و خروجی کلاس‌ها (کلاس‌بندی) یا برای پیش‌بینی‌های هر درخت به شکل مجزا، کار می‌کنند. جنگل‌های تصادفی برای درختان تصمیم که در مجموعه آموزشی دچار over fitting می‌شوند، مناسب هستند.

جنگل تصادفی یک الگوریتم یادگیری نظارت شده محسوب می‌شود. همانطور که از نام آن مشهود است، این الگوریتم جنگلی را به طور تصادفی می‌سازد. «جنگل» ساخته شده، در واقع گروهی از «درخت‌های تصمیم « (Decision Trees)است. کار ساخت جنگل با استفاده از درخت‌ها اغلب اوقات به روش «کیسه‌گذاری « (Bagging)انجام می‌شود. ایده اصلی روش کیسه‌گذاری آن است که ترکیبی از مدل‌های یادگیری، نتایج کلی مدل را افزایش می‌دهد. به بیان ساده، جنگل تصادفی چندین درخت تصمیم ساخته و آن‌ها را با یکدیگر ادغام می‌کند تا پیش‌بینی‌های صحیح‌تر و پایدارتری حاصل شوند.

یکی از مزایای جنگل تصادفی قابل استفاده بودن آن، هم برای مسائل دسته‌بندی و هم رگرسیون است که غالب سیستم‌های یادگیری ماشین کنونی را تشکیل می‌دهند. در اینجا، عملکرد جنگل تصادفی برای انجام «دسته‌بندی (Classification) «تشریح خواهد شد، زیرا گاهی دسته‌بندی را به عنوان بلوک سازنده یادگیری ماشین در نظر می‌گیرند. در تصویر زیر، می‌توان دو جنگل تصادفی ساخته شده از دو درخت را مشاهده کرد.

جنگل تصادفی دارای فراپارامترهایی مشابه درخت تصمیم (Bagging Classifier)است. خوشبختانه، نیازی به ترکیب یک درخت تصمیم با یک دسته‌بند کیسه‌گذاری نیست و می‌توان از (Classifier-Class) جنگل تصادفی استفاده کرد. با جنگل تصادفی، و در واقع (Random Forest Regressor) می‌توان به حل مسائل رگرسیون نیز پرداخت.

جنگل تصادفی، تصادفی بودن افزوده‌ای را ضمن رشد درختان به مدل اضافه می‌کند. این الگوریتم، به جای جست‌وجو به دنبال مهم‌ترین ویژگی‌ها هنگام تقسیم کردن یک (Node) ، به دنبال بهترین ویژگی‌ها در میان مجموعه تصادفی از ویژگی‌ها می‌گردد. این امر منجر به تنوع زیاد و در نهایت مدل بهتر می‌شود. بنابراین، در جنگل تصادفی، تنها یک زیر مجموعه از ویژگی‌ها توسط الگوریتم برای تقسیم یک گره در نظر گرفته می‌شود. با استفاده افزوده از آستانه تصادفی برای هر ویژگی به جای جست‌وجو برای بهترین آستانه ممکن، حتی می‌توان درخت‌ها را تصادفی‌تر نیز کرد (مانند کاری که درخت تصمیم نرمال انجام می‌دهد)

دیگر خصوصیت عالی الگوریتم جنگل تصادفی این است که اندازه‌گیری اهمیت نسبی هر ویژگی روی پیش‌بینی در آن آسان است. کتابخانه پایتون (Sklearn) ابزار خوبی را برای این کار فراهم می‌کند. این ابزار، اهمیت یک ویژگی را با نگاه کردن به تعداد گره‌های درخت که از آن ویژگی استفاده می‌کنند، اندازه‌گیری کرده و ناخالصی را در سرتاسر درخت‌های جنگل کاهش می‌دهد. ابزار مذکور، این امتیاز را به صورت خودکار برای هر ویژگی پس از آموزش دادن محاسبه و نتایج را مقیاس می‌کند، بنابراین مجموع همه اهمیت‌ها برابر با ۱ است.

در درخت تصمیم، هر گره داخلی یک تست را روی ویژگی‌ها نمایش می‌دهد (مثلا، سکه در یک پرتاب شیر می‌آید یا خط)، هر شاخه نشانگر خروجی تست و هر گره برگ نشانگر برچسب دسته (کلاس) است (تصمیم پس از محاسبه همه این گره‌ها آن‌ها اتخاذ می‌شود). گره‌ای که هیچ فرزندی ندارد، برگ (Leaf) محسوب می‌شود.

از طریق بررسی اهمیت ویژگی‌ها، کاربر می‌تواند تصمیم بگیرد که کدام ویژگی‌ها را امکان دارد بخواهد با توجه به اینکه به طور کلی و یا به اندازه کافی در فرآیند تصمیم‌گیری نقش ندارند، حذف کند. این مساله حائز اهمیت است زیرا، یک قانون کلیدی در یادگیری ماشین آن است که هرچه ویژگی‌ها بیشتر باشند، احتمال آنکه مدل دچار (OverFitting)یا (UnderFitting) شود وجود دارد.

تفاوت بین درخت تصمیم و جنگل تصادفی

جنگل تصادفی مجموعه‌ای از درخت‌های تصمیم است. اما تفاوت‌هایی میان آن‌ها وجود دارد. اگر یک مجموعه داده ورودی با ویژگی‌ها و برچسب‌های آن به عنوان ورودی به الگوریتم داده شود، برخی از مجموعه قوانین را به گونه‌ای فرموله می‌کند که برای انجام پیش‌بینی مورد استفاده قرار می‌گیرند. برای مثال، اگر کاربر قصد داشته باشد پیش‌بینی کند که «آیا فرد روی یک تبلیغ آنلاین کلیک می‌کند یا نه»، می‌تواند تبلیغاتی که فرد در گذشته روی آن‌ها کلیک کرده و ویژگی‌هایی که تصمیمات او را توصیف می‌کنند گردآوری کند. سپس، با استفاده از آن‌ها می‌تواند پیش‌بینی کند که یک تبلیغ مشخص توسط یک فرد خاص کلیک می‌شود یا خیر. در مقایسه، الگوریتم درخت تصمیم مشاهدات را به صورت تصادفی انتخاب می‌کند، برای ویژگی‌های ساخت چندین درخت تصمیم می‌گیرد و سپس از محاسبه میانگین نتایج استفاده می‌کند. تفاوت دیگر آن است که درخت تصمیم عمیق (Deep) ممکن است دچار (Overfitting) شود. جنگل تصادفی اغلب اوقات با ساخت زیردرخت تصادفی از ویژگی‌ها و ساخت درخت کوچک‌تر با استفاده از این زیردرخت، از بیش‌برازش جلوگیری می‌کند. پس از آن، زیردرخت‌های تصادفی را با یکدیگر ترکیب می‌کند. این راهکار همیشه جوابگو نیست و فرآیند محاسبات را بسته به تعداد جنگل‌های تصادفی که ساخته می‌شوند کندتر می‌کند.

الگوریتم

جنگل تصادفی ، در واقع گروهی از درخت‌های تصمیم (Decision Tree) است. در الگوریتم جنگل تصادفی، به منظور دسته‌بندی یک نمونه جدید بر پایه ویژگی‌های (Features) آن نمونه، هر درخت، کلاسی که نمونه را متعلق به آن می‌داند مشخص می‌کند و در واقع یک رای می‌دهد. جنگل، بر اساس این آرا، دسته‌ای که بیشترین رای را به خود اختصاص داده انتخاب و به عنوان دسته نهایی نمونه داده تعیین می‌کند.

کیسه‌گذاری درختان

مجموعه داده را با D نمایش میدهیم، $D = (x_1,y_1),(x_2,y_2), \ldots, (x_n,y_n)$ و B درخت تصادفی با ایجاد B داده جدید از D ایجاد میکنیم. مدل نهایی با میانگین گرفتن یا رأی‌گیری بین درختان کار می‌کند. جزئیات این الگوریتم در زیر آمده است:

برای $b = 1$ تا B:

- n نمونه با جایگزینی از داده D انتخاب کن و این نمونه‌ها را در مجموعه داده D_b قرار بده. از آنجا که نمونه‌گیری با جایگزینی صورت می‌گیرد یک نمونه ممکن است چندین بار انتخاب شود.
- یک درخت تصادفی به اسم T_b با D_b به روش پایین بساز:

هر دفعه برای پیدا کردن بهترین متغیر ابتدا یک تعداد مشخصی از متغیرها را کاملا به صورت تصادفی انتخاب کن (مثلا m تا، m از قبل به مسئله داده شده‌است، و معمولاً با جذر تعداد متغیرها برابر است) و از میان آن‌ها بهترین متغیر را انتخاب کن.

در مسئله رگرسیون مدل نهائی، میانگین تمامی درخت‌ها است. یعنی :

$$F(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

از طرفی دیگر در مسئله Classification با رأی‌گیری بین درختان به جواب نهائی می‌رسیم.

این نوع ترکیب مدل‌ها جواب بهتری به ما می‌دهد زیرا گوناگونی و تنوع مدل‌ها را افزایش می‌دهد بدون این که بایاس را افزایش دهد! این بدین معناست که زمانی که پیش‌بینی تکی از یک درخت دارای نویز بالایی درون مجموعه دسته آموزش دیده اش باشد، در میانگین بسیاری از درخت‌ها این نویز وجود نخواهد داشت. به شکل ساده آموزش درختان به صورت تکی می‌تواند درخت‌های در ارتباط قوی تری را ارائه دهد. بوت استرپ کردن نمونه، روشی برای یکپارچه‌تر کردن درخت‌ها با نمایش مجموعه داده‌های آموزش دیده گوناگون است.

هایپرپارامترهای مهم

هایپرپارامترها در جنگل تصادفی برای افزایش قدرت پیش‌بینی مدل و یا سریع‌تر کردن آن مورد استفاده قرار می‌گیرند. در ادامه، پیرامون هایپرپارامترهای تابع جنگل تصادفی توکار کتابخانه sklearns صحبت می‌شود.

۱ .افزایش قدرت پیش‌بینی

یک هایپرپارامتر «n_estimators» وجود دارد که در واقع تعداد درختانی است که الگوریتم پیش از دریافت آرای بیشینه یا دریافت میانگین پیش‌بینی‌ها می‌سازد. به طور کلی، تعداد بیشتر درخت‌ها، کارایی را افزایش می‌دهند و پیش‌بینی‌ها را پایدارتر می‌سازند، اما محاسبات را کندتر می‌کنند. دیگر هایپرپارامتر مهمی که پیرامون آن صحبت خواهد شد، «min_sample_leaf»است. این هایپرپارامتر همانطور که از نام آن مشخص است، حداقل تعداد برگ‌هایی که برای تقسیم یک نود خارجی مورد نیاز هستند را مشخص می‌کند.

۲ .افزایش سرعت مدل

هایپرپارامتر «n_jobs» به موتور می‌گوید که اجازه استفاده از چه تعداد پردازنده را دارد. اگر مقدار این هایپرپارامتر برابر با ۱ باشد، می‌تواند تنها از یک پردازنده استفاده کند. مقدار «۱» بدین معنا است که هیچ محدودیتی وجود ندارد «random_state» .خروجی مدل را تکرارپذیر می‌کند. مدل هنگامی که مقدار قطعی برای random_state دارد و اگر هایپرپارامترها و داده‌های آموزش مشابهی به آن داده شود، همیشه نتایج مشابهی را تولید می‌کند.

در نهایت، یک هایپرپارامتر «oob_score» وجود دارد (به آن oob sampling نیز گفته می‌شود)، که روشی برای اعتبارسنجی متقابل(Random Forest) ، جنگل تصادفی است. در این نمونه‌برداری، حدود یک سوم از داده‌ها برای آموزش مدل استفاده نمی‌شوند و برای ارزیابی کارایی آن مورد استفاده قرار می‌گیرند. به این نمونه‌ها (Bag Samples)گفته می‌شود. این راهکار، شباهت زیادی به روش اعتبارسنجی (leave-one-out) دارد، اما تقریبا هیچ بار محاسباتی برای آن وجود ندارد.

یکی از مزایای جنگل تصادفی آن است که هم برای رگرسیون و هم برای دسته‌بندی قابل استفاده است و راهکاری مناسب برای مشاهده اهمیت نسبی که به ویژگی‌های ورودی تخصیص داده می‌شود است. جنگل تصادفی الگوریتمی بسیار مفید و با استفاده آسان محسوب می‌شود، زیرا هایپرپارامترهای پیش‌فرض آن اغلب نتایج پیش‌بینی خوبی را تولید می‌کنند. همچنین، تعداد هایپرپارامترهای آن بالا نیست و درک آن‌ها آسان است.

یکی از بزرگ‌ترین مشکلات در یادگیری ماشین، بیش‌برازش است، اما اغلب اوقات این مساله به آن آسانی که برای دسته‌بند جنگل تصادفی به وقوع می‌پیوندد، اتفاق نمی‌افتد. محدودیت اصلی جنگل تصادفی آن است که تعداد زیاد درخت‌ها می‌توانند الگوریتم را برای پیش‌بینی‌های جهان واقعی کند و غیر موثر کنند.

به طور کلی، آموزش دادن این الگوریتم‌ها سریع انجام می‌شود، اما پیش‌بینی کردن پس از آنکه مدل آموزش دید، اندکی کند به وقوع می‌پیوندد. یک پیش‌بینی صحیح‌تر نیازمند درختان بیشتری است که منجر به کندتر شدن مدل نیز می‌شود. در اغلب کاربردهای جهان واقعی، الگوریتم جنگل تصادفی به اندازه کافی سریع عمل می‌کند، اما امکان دارد شرایط‌هایی نیز وجود داشته باشد که در آن کارایی زمان اجرا حائز اهمیت است و دیگر رویکردها ترجیح داده می‌شوند. البته، جنگل تصادفی یک ابزار مدل‌سازی پیش‌بین و نه یک ابزار توصیفی است. این یعنی، اگر کاربر به دنبال ارائه توصیفی از داده‌های خود است، استفاده از رویکردهای دیگر ترجیح داده می‌شوند.

برخی از زمینه‌های کاربرد

الگوریتم جنگل تصادفی در زمینه‌های گوناگونی مانند بانکداری، بازار بورس، پزشکی و تجارت الکترونیک مورد استفاده قرار می‌گیرد. در بانکداری، از این الگوریتم برای شناسایی مشتریانی که بیشتر از سایرین از خدمات بانکی استفاده می‌کنند و بدهی خود را به موقع باز می‌گردانند استفاده می‌شود. این الگوریتم برای شناسایی مشتریان کلاهبرداری که قصد کلاهبرداری از بانک را دارند نیز مورد بهره‌برداری قرار می‌گیرد.

در امور مالی، از جنگل تصادفی برای شناسایی رفتار بورس در آینده استفاده می‌شود. در حوزه پزشکی، این الگوریتم برای شناسایی ترکیب صحیحی از مولفه‌ها و تحلیل تاریخچه پزشکی بیمار، برای شناسایی بیماری او مورد استفاده قرار می‌گیرد. در نهایت، در تجارت الکترونیک(E-commerce) ، جنگل تصادفی برای شناسایی اینکه مشتریان یک محصول را دوست داشته‌اند یا خیر، استفاده می‌شود.

خلاصه

جنگل تصادفی الگوریتم خوبی برای آموزش در اوایل فرآیند توسعه مدل است، تا چگونگی عملکرد آن بررسی شود و از همین رو ساخت یک جنگل تصادفی «بد»به دلیل سادگی این الگوریتم، عملا سخت‌تر از ساخت یک مدل خوب است. همچنین، در صورت نیاز به توسعه مدل در بازه زمانی کوتاه‌تر، این الگوریتم گزینه خوبی محسوب می‌شود. علاوه بر این، مدل شاخص خوبی از اهمیتی که به ویژگی‌ها می‌دهد را فراهم می‌کند.

الگوریتم SVM

SVMیا ماشین بردار پشتیبان ، یک دسته بند یا مرزی است که با معیار قرار دادن بردارهای پشتیبان ، بهترین دسته بندی و تفکیک بین داده ها را برای ما مشخص می کند.

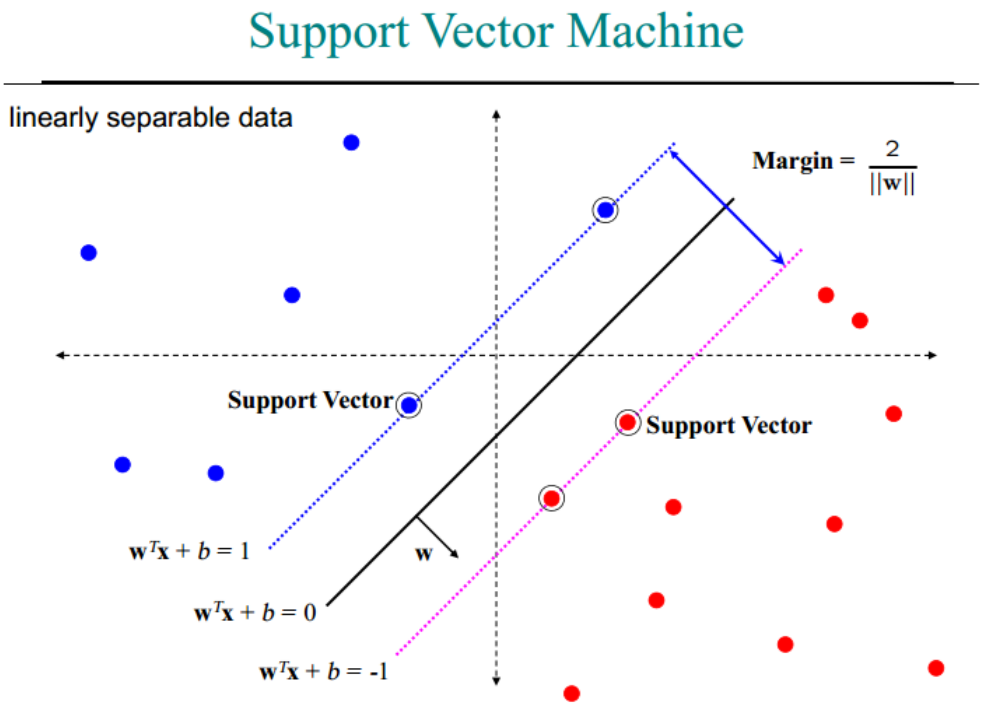
ماشین بردار پشتیبان (SVM) یک الگوریتم نظارت‌شده یادگیری ماشین است که هم برای مسائل طبقه‌بندی و هم مسائل رگرسیون قابل استفاده است؛ با این حال از آن بیشتر در مسائل طبقه‌بندی استفاده می‌شود. در الگوریتم SVM، هر نمونه داده را به عنوان یک نقطه در فضایn بعدی روی نمودار پراکندگی داده‌ها ترسیم کرده(n تعداد ویژگی‌هایی است که یک نمونه داده دارد) و مقدار هر ویژگی مربوط به داده‌ها، یکی از مؤلفه‌های مختصات نقطه روی نمودار را مشخص می‌کند. سپس، با ترسیم یک خط راست، داده‌های مختلف و متمایز از یکدیگر را دسته‌بندی می‌کند. به بیان ساده، بردارهای پشتیبان در واقع مختصات یک مشاهده منفرد هستند. ماشین بردار پشتیبان

مرزی است که به بهترین شکل دسته‌های داده‌ها را از یکدیگر جدا می‌کند.بردارهای پشتیبان به زبان ساده، مجموعه ای از نقاط در فضای n بعدی داده ها هستند که مرز دسته ها را مشخص می کنند و مرزبندی و دسته بندی داده ها براساس آنها انجام می شود و با جابجایی یکی از آنها، خروجی دسته بندی ممکن است تغییر کند. در SVM فقط داده های قرار گرفته در بردارهای پشتیبان مبنای یادگیری ماشین و ساخت مدل قرار می گیرند و این الگوریتم به سایر نقاط داده حساس نیست و هدف آن هم یافتن بهترین مرز در بین داده هاست به گونه ای که بیشترین فاصله ممکن را از تمام دسته ها (بردارهای پشتیبان آنها) داشته باشد.

یک راه ساده برای ساخت یک دسته بند بهینه ، محاسبه فاصله ی مرزهای به دست آمده با بردارهای پشتیبان هر دسته (مرزی ترین نقاط هر دسته یا کلاس) و در نهایت انتخاب مرزیست که از دسته های موجود، مجموعاً بیشترین فاصله را داشته باشد. این عمل تعیین مرز و انتخاب خط بهینه (در حالت کلی ، ابر صفحه مرزی) به راحتی با انجام محاسبات ریاضی نه چندان پیچیده قابل پیاده سازی است.

توزیع غیر خطی داده ها و کاربرد ماشین بردار پشتیبان
اگر داده ها به صورت خطی قابل تفکیک باشند، الگوریتم فوق می تواند بهترین ماشین را برای تفکیک داده ها و تعیین دسته یک رکورد داده، ایجاد کند اما اگر داده ها به صورت خطی توزیع شده باشند ، در این حالت، ما نیاز داریم داده ها را به کمک یک تابع ریاضی (Kernel functions) به یک فضای دیگر ببریم (نگاشت کنیم) که در آن فضا، داده ها تفکیک پذیر باشند و بتوان SVM آنها را به راحتی تعیین کرد.

ماشین بردار پشتیبان یا SVM داده ها را با توجه به دسته های از پیش تعیین شده آنها به یک فضای جدید می برد به گونه ای که داده ها به صورت خطی (یا ابر صفحه) قابل تفکیک و دسته بندی باشند و سپس با یافتن خطوط پشتیبان (صفحات پشتیبان در فضای چند بعدی) ، سعی در یافتن معادله خطی دارد که بیشترین فاصله را بین دو دسته ایجاد می کند. در شکل زیر داده ها در دو دوسته آبی و قرمز نمایش داده شده اند و خطوط نقطه چین ، بردار های پشتیبان متناظر با هر دسته را نمایش می دهند که با دایره های دوخط مشخص شده اند و خط سیاه ممتد نیز همان SVM است . بردار های پشتیبان هم هر کدام یک فرمول مشخصه دارند که خط مرزی هر دسته را توصیف می کند.

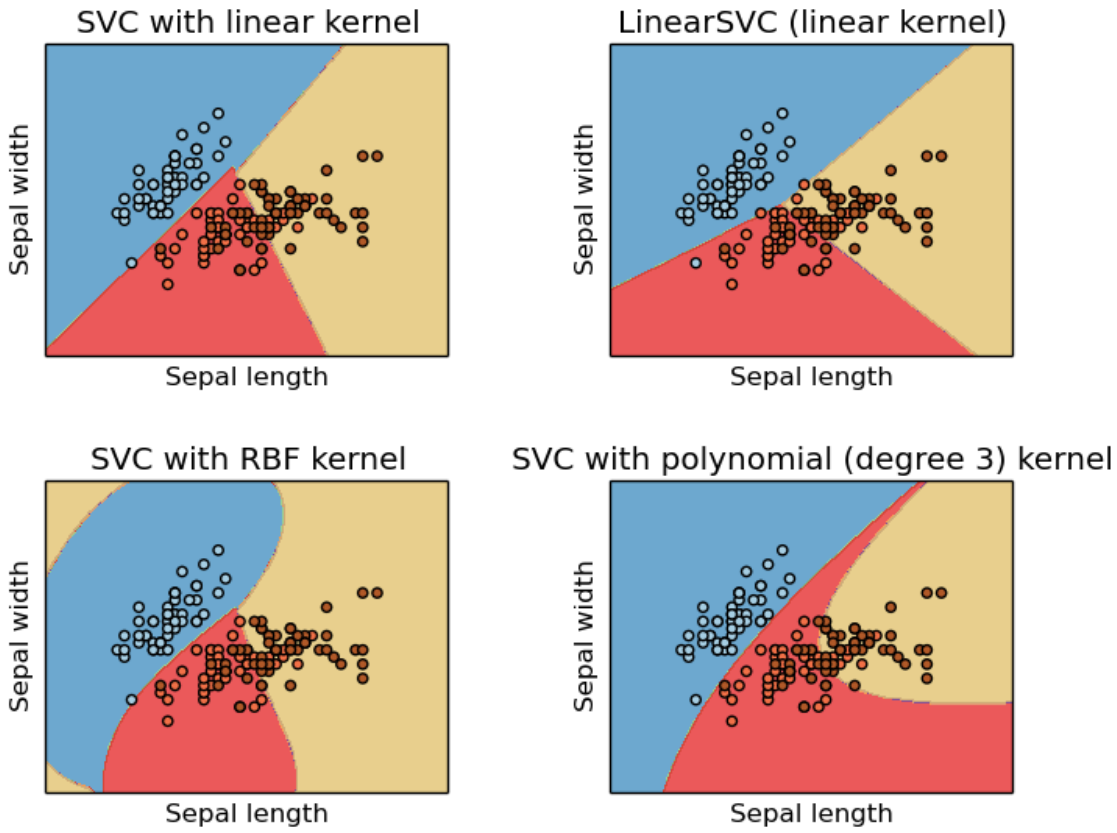


SVMدر پایتون

برای استفاده از ماشین بردار پشتیبان در پایتون، میتوان از کتابخانه یادگیری ماشین پایتون به نام *scikit–learn* استفاده کرد که تمام کرنل ها و توابع نگاشت را به صورت آماده شده دارد .سه تا تابع LinearSVC , NuSVC , SVC وظیفه اصلی دسته بندی را برعهده دارند.

. (SVC = Support Vector Classifier)

نمونه ای از دسته بندی با این توابع :



ماشین بردار پشتیبانی در عمل

برای استفاده از SVM در مورد داده های واقعی ، چندین نکته را باید رعایت کرد تا نتایج قابل قبولی گرفت:

۱. پالایش داده ها (نقاط پرت ، داده های ناموجود و...)
۲. عددی کردن داده ها و نرمال کردن آن ها . داده هایی مانند جنسیت، رشته تحصیلی و ... را به عدد تبدیل میکنیم و سعی میکنیم مقادیر همه صفات بین یک تا منهای یک [۱,-۱] نرمال شوند تا بزرگ یا کوچک بودن مقادیر یک ویژگی داده ها، ماشین را تحت تاثیر قرار ندهد.
۳. کرنل های مختلف را امتحان و به ازای هر کدام، با توجه به مجموعه داده آموزشی که در اختیار داریم و دسته بندی داده های آنها مشخص است، دقت SVM را اندازه گیری میکنیم و در صورت نیاز پارامتر های توابع تبدیل را تغییر میدهیم تا جواب های بهتری بگیریم. این کار را برای کرنل های مختلف هم امتحان میکنیم.

نقاط ضعف ماشین بردار پشتیان

- این نوع الگوریتم ها، محدودیت های ذاتی دارند مثلا هنوز مشخص نشده است که به ازای یک تابع نگاشت ، پارامترها را چگونه باید تعیین کرد.
- ماشینهای مبتنی بر بردار پشتیبان به محاسبات پیچیده و زمان بر نیاز دارند و به دلیل پیچیدگی محاسباتی، حافظه زیادی نیز مصرف می کنند.
- داده های گسسته و غیر عددی هم با این روش سازگار نیستند و باید تبدیل شوند.

با این وجود، SVMها دارای یک شالوده نظری منسجم بوده و جواب های تولید شده توسط آنها ، سراسری و یکتا می باشد. امروزه ماشینهای بردار پشتیبان، به متداول ترین تکنیک های پیش بینی در داده کاوی تبدیل شده اند.

کاربردهای SVM

در این بخش مروری خواهیم داشته به کاربردهای تشخیص الگو با استفاده از SVM. با توجه به هدف کاربردها در هفت دسته کاربرد مختلف برای SVM دسته بندی شده است که در اینجا به آنها اشاره می کنیم. تعدادی از کاربردها که در دسته بندی خاصی گنجانده نشده است در گروه سایر کاربردها، ذکر شده اند.

۱- تشخیص و شناسایی چهره

این دسته یکی از محبوبترین زمینه های در بایومتریک، تایید هویت، کنترل دسترسی و نظارت ویدئویی می باشد. زمینه های تحقیقاتی فعالی در این دسته وجود دارند که برای این کاربردها از روش های مختلفی استفاده می نمایند. به هرحال رسیدن به کارائی مطلوب در این زمینه بسیار مشکل خواهد بود. شناسایی چهره های افرادی که ویژگیهای ظاهری نزدیک به هم دارند بسیار مشکل خواهد بود. همچین حالات مختلف برای یک چهره خاص نظیر آرایش های متفاوت و غیره تشخیص را بسیار مشکل می نماید. همچنین استفاده از عینک و یا ریش و سیبیل میتواند تشخیص را مشکل و پیچیده نماید.

در سال های اخیر تحقیقات زیادی برای استفاده از SVM در کاربردهای تشخیص چهره، شناسایی و بازشناسی چهره و تشخیص حالات چهره صورت گرفته است. هر کدام از روش های فوق از ورودی های، پایگاه داده ها و هسته های مختلفی برای طبقه بندی کننده SVM استفاده کرده اند. اولین بار Osuna از SVM برای تشخیص چهره و تابع هسته چند جمله ای درجه دوم استفاده نمود. در این کاربرد وی از یک تصویر چهره شناسایی ماشین

در مورد چهره به دو دسته شناسایی چهره و تایید چهره تقسیم میشود. Guo از SVM چندکلاسه و یک درخت باینری برای شناسایی چهره استفاده نمود. ویژگی های نرمال شده که توسط روش PCA استخراج شده بودند، به عنوان ورودی SVM بکار گرفته شدند.

۲- تشخیص و شناسایی اشیاء

هدف از تشخیص یا شناسایی اشیا یافتن و تعقیب افراد در حال حرکت و یا بررسی ترافیک برای کاربردهای کنترل ترافیک می باشد. شناسایی اشیا ۳ بعدی نیز یکی از این زمینه ها می باشد. COIL یک پایگاه داده معروف می باشد که شامل ۱۲۲۲ تصویر از ۱۲۲ شیء و از ۱۲ زاویه دید مختلف می باشد. این پایگاه داده در بسیاری از کاربردهای تحقیقاتی در این زمینه مورد استفاده قرار گرفته است. Roobaert تشخیص اشیاء ۳ بعدی را توسط SVM انجام داد و توانایی SVM را در این شناسایی از زاویه های مختلف، بررسی کرد. M.Pontil و A.Verri با آمیختن تصاویر پایگاه دادهی COIL با نویز و شیفت دادن تصاویر، کارائی بالایی را نتیجه گرفتند.

۳- تشخیص دست نوشته و ارقام

در میان کاربردهای مختلفی که بر پایه SVM انجام شده است، تشخیص ارقام دستنویس توسط SVM از تمامی الگوریتم های یادگیری دیگر، کارائی بالاتری داشته است. مشکل عمده در مسئله تشخیص دست نوشته، وجود الگوهای مختلف و متنوع نوشتاری بسیار زیاد می باشد. مدلهای Elastic که برپایه مشاهدات محلی و برنامه نویسی پویا شکل گرفته اند مانند HMM ، برای تحلیل این تنوع کارا و مناسب هستند. Choisy برای ترکیب این توانایی محلی با ویژگی های سراسری، از NSPH-HMM برای مشاهدات محلی و نرمالسازی و از SVM برای مشاهدات سراسری بر روی خروجی نرمال شده توسط NSPH-HMM ، استفاده نمود. کارهای مختلفی در این زمینه انجام شده است که در مجموع برتری این روش را به سایر روش ها در شناسایی دست نوشته ها نشان میدهد.

۴- تشخیص صحبت و گوینده

در مسئله تشخیص صحبت و یا شناسایی گوینده، دو روش بسیار محبوب Discriminative Classifier ها و Generative Model Clasifier ها می باشند. روش های که از Discriminative Classifier ها استفاده می کنند، شامل درخت های تصمیم گیری، شبکه های عصبی و SVM می باشند. معروفترین روش Generative Model Clasifier شامل مدل Hidden Markov(HMM) و مدل ترکیبی Gaussian (GMM) می باشد. Bengio و Wan از SVM برای شناسایی گوینده با پایگاه داده های مختلف، استفاده

نمودند. آنها بر روی داده های وابسته به متن و غیر وابسته به متن کار کرده و روش SVM را با آستانه گذاری کلاسیک برای تصمیم گیری در رد یا پذیرش تایید گوینده، جایگزین نمودند. استفاده های گوناگونی از SVM در این زمینه نیز صورت گرفته است.

۵- بازیابی تصاویر و اطلاعات

Content-based image retrival نتیجه زمنیه تحقیقات مهمی در زمینه کتابخانه های دیجیتال و پایگاه داده های چندرسانه ای می باشد. Guo معیاری جدیدی را تحت عنوان فاصله از مرز، را برای بازیابی بافت تصاویر ابداع نمود که در آن مرز بین دسته ها توسط SVM بدست آمده است. برای بازیابی تصاویر مرتبط بیشتر با تصویر ورودی، طبقه بندی کننده SVM برای جدا کردن تصاویر به دو دسته مرتبط و غیرمرتبط، استفاده شده است. Tian،Drucker و Zhang روش اتوماتیکی را با استفاده از SVM برای بازیابی تصاویر ابداع نمودند که در آن وزن ها با فاصله از ابر صفحه تعیین می شوند و داده ها به دو دسته داده های مثبت و منفی تقسیم می شوند.

۶- پیش بینی

هدف اصلی در بسیاری از روش های پیشگویی غیرخطی، پیش بینی نقطه بعدی در یک سری زمانی می باشد. Tay و Cao روش C-ascending SVM را با کاهش C ارائه دادند. این ایده بر این فرض استوار است که بهتر است که وزن بیشتری را به داده های اخیر بدهیم تا به داده های دورتر. نتایج آنها نشان داد که روش C-ascending SVM کارائی بالاتری را نسبت به SVM استاندارد در پیشگویی سریه ای زمانی اقتصادی، ارائه میدهد. Fan روش SVM را بر مسئله یش بینی مشکلات شرکت ها با توجه به وضعیت اقصادی آنها، منطبق نمود. برای این مسئله انتخاب متغیرهای ورودی (شاخصهای اقتصادی)، روی کارائی سیستم تاثیرگذار می باشد. در مقاله ارائه شده، پیشنهاد داده که از داده های مناسب استفاده شود که فاصله بین دسته های بیشترین مقدار و فاصله بین داده های مشابه را کمترین مقدار قرار می دهند.

۷- سایر کاربردها

کاربردهای بسیار زیاد دیگری برای SVM در زمینه تشخیص الگو وجود دارد. بعنوان مثال Yang از SVM برای دسته بندی بصری بر اساس جنسیت، با استفاده از تصاویر کوچک (۱۲*۱۲) و با کیفیت پایین استفاده نمود. او برای این کار از پایگاه دادهی FERET استفاده نمود و از ۱۱۱۱ تصویر استفاده نمود. پس از تعلیم نشان داده شد که کارائی این روش در مقابل روش های قبلی مانند RBF،FLD و ... بسیار بالاتر می باشد. Gutta از SVM برای دسته بندی حالات چهره بروی پایگاه داده FERET استفاده نمود و به ٪۱۰۰ دقت رسید. Yao

پنج دسته اثر انگشت را فرض نمود و از SVM برای این دسته بندی استفاده نمود و کارائی خوبی را نتیجه گرفت. علاوه بر کاربردهای ذکر شده، بسیاری از کاربردهای دیگر مانند خلاصه کردن داده ها، شناسایی هدف و... وجود دارد که SVM به خوبی از عهده آنها برمی آید. در کاربرد خلاصه سازی داده ها، زیرمجموعه ای کوچک از پایگاه دادهای بزرگ انتخاب می شود و دقت تفکیک کننده که بر روی این داده های کاهش یافته عمل میکند، با نتیجه آموزش روی تمام داده ها، مقایسه می شود.

خلاصه

ماشین بردار پشتیبان یک الگوریتم دسته‌بندی بسیار قدرتمند است. وقتی از آن همراه با الگوریتم های جنگل تصادفی و دیگر ابزارهای یادگیری ماشین استفاده کنیم، این الگوریتم می‌تواند مدلی بسیار قابل توجه برای دسته‌بندی داده‌ها ارائه کند. الگوریتم ماشین بردار پشتیبان هنگامی که قدرت پیش‌بینی بالا مورد نیاز باشد یک گزینه بسیار عالی است. تصویرسازی این الگوریتم‌ها کار دشواری محسوب می‌شود، زیرا فرمول‌بندی پیچیده ای دارند. ماشینهای بردار پشتیبان، الگوریتم های بسیار قدرتمندی در دسته بندی و تفکیک داده ها هستند بخصوص زمانی که با سایر روشهای یادگیری ماشین مانند روش جنگل تصادفی تلفیق شوند. این روش برای جاهایی که با دقت بسیار بالا نیاز به ماشینی داده ها داریم، به شرط اینکه توابع نگاشت را به درستی انتخاب کنیم، بسیار خوب عمل می کند.

سوال سوم)

اعتبار سنجی متقابل(Cross Validation)

اغلب در مدل‌سازی، بخصوص در(Machine Learning) ، احتیاج به برآورد پارامترهای مدل داریم. در صورتی که تعداد پارامترها زیاد باشد، پیچیدگی مدل زیاد شده و ممکن است محاسبات به سادگی قابل انجام نباشند.

اعتبارسنجی متقابل یکی از راه‌هایی است که می‌توان تعداد پارامترها (متغیرهای) مدل را بصورت بهینه تعیین کرد.

برای مثال می‌دانیم که در (Linear Regression) بر اساس یک نمونه تصادفی دو یا چند بعدی از متغیر پاسخ و متغیرهای مستقل، هدف برآورد پارامترهای مدل است. این کار با استفاده از کمینه سازی تابع مربعات خطا صورت می‌گیرد. بنابراین مشخص است که برآورد پارامترهای مدل به شکلی صورت گرفته که برای مشاهدات ثبت شده، میانگین مربعات خطا حداقل ممکن خواهد بود.

با افزایش پارامترهای مدل (افزایش متغیرها مستقل)، کارایی مدل افزایش خواهد یافت، زیرا میانگین مربعات برای نمونه جمع‌آوری شده به هر حال با افزایش متغیرها کاهش خواهد یافت. بنابراین مناسب‌ترین مدل برای نمونه با حداکثر تعداد پارامتر بدست آمده است. برآورد پارامترهای حاصل از این نمونه را به اصطلاح (In-sample Estimation) می‌نامند.

ولی اگر این مدل، برای نمونه دیگری از همان جامعه به کار رود، شاید مناسب محسوب نشود. به نظر می‌رسد افزایش متغیرها، ممکن است به کارایی مدل آسیب برساند. چنین مشکلی را با نام (Overfitting) می‌شناسیم. راه حل برای چنین مسئله‌ای می‌تواند استفاده از اعتبارسنجی متقابل باشد که هدف در آن تعیین تعداد متغیرها (پارامترهای) مناسب مدل است. گاهی به این روش، (Rotation Estimation) یا (Out-of-sample testing) نیز می‌گویند.

اعتبارسنجی متقابل

برای سنجش کارایی یک مدل، معمولا دو روش به کار گرفته می‌شود: ۱- ارزیابی براساس فرضیاتی که باید مدل در آن‌ها صدق کند. ۲- ارزیابی براساس کارایی مدل در پیش‌بینی مقدارهای جدید مشاهده نشده است.

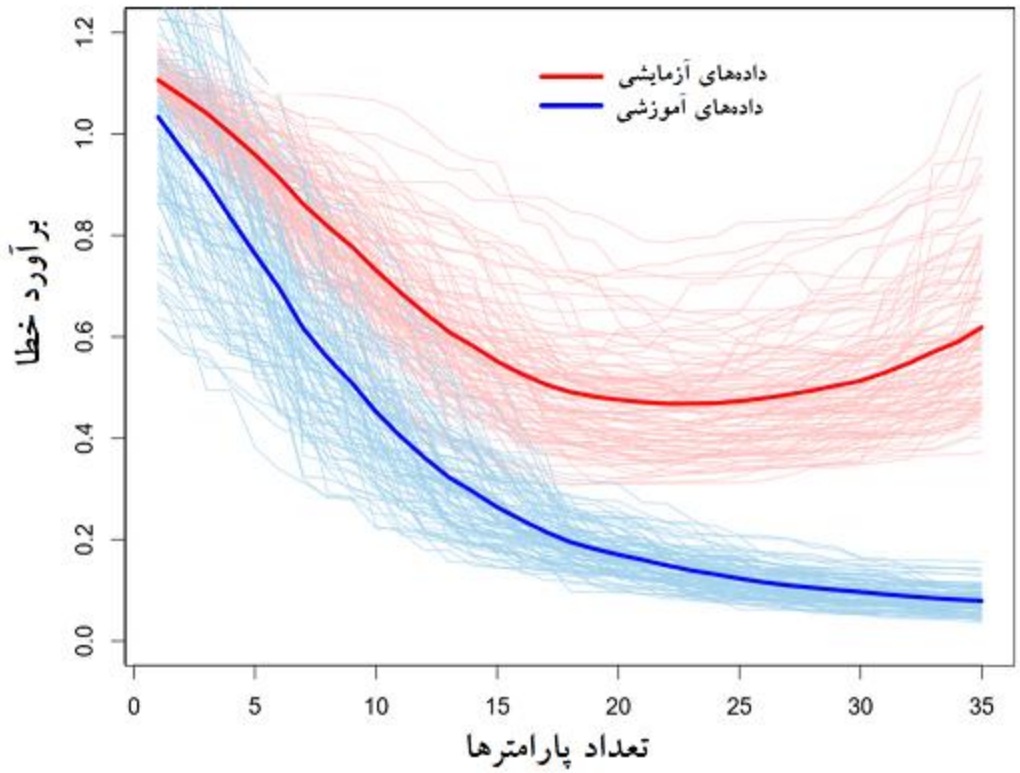
برای بررسی و ارزیابی مدل براساس رویه شماره ۱، تکیه بر داده‌هایی است که مشاهده شده و در ساختن مدل به کار رفته‌اند. برای مثال در رگرسیون خطی، فرض بر این است که باقی‌مانده‌های مدل رگرسیونی باید تصادفی و با واریانس ثابت باشند. همچنین توزیع آن‌ها نیز باید نرمال باشد. بررسی صحت این فرضیات می‌تواند به عنوان معیاری برای سنجش اعتبار مدل محسوب شود. پارامترهای مدل رگرسیونی با کمینه سازی مربعات خطا حاصل

می‌شود. بنابراین انتظار داریم که مدل ساخته شده نسبت به هر مدل دیگری مجموع کمترین مجموع مربعات خطا را نیز داشته باشد.

مشخص است که بررسی موارد بالا براساس داده‌هایی که مدل براساس آن ساخته شده است، میسر است ولی نمی‌توان کارایی مدل را برای داده‌های جدیدی که هنگام مدل‌سازی مشاهده نشده‌اند، سنجید.

اما در اعتبارسنجی متقابل، تکیه بر داده‌هایی است که مشاهده شده‌اند ولی در هنگام ساختن مدل به کار گرفته نمی‌شوند. این داده‌ها به منظور بررسی و سنجش کارایی مدل برای پیش‌بینی داده‌های جدید به کار می‌روند.

به این ترتیب برای اندازه‌گیری کارایی مدل و بهینه بودن آن متوسل به برآورد خطای مدل براساس داده‌هایی می‌شویم که برای اعتبارسنجی متقابل کنار گذاشته شده‌اند. برآورد این خطا را معمولا (Out-of-sample error) می‌نامند.در تصویر زیر، نمودار مربوط به روند تغییرات خطای مدل براساس داده‌های آموزشی و داده‌های آزمایشی براساس پیچیدگی مدل (تعداد پارامترها) ترسیم شده است. خطوط کم‌رنگ آبی و قرمز، خطای مدل برای داده‌های آموزشی و آزمایشی را برحسب تعداد پارامترها نشان می‌دهند. همچنین خط آبی پررنگ، میانگین خطای مدل برای داده‌های آموزشی و خط قرمز پررنگ نیز میانگین خطای مدل برای داده‌های آزمایشی است.



با توجه به این نمودار مشخص است که با افزایش تعداد پارامترهای مدل، میزان خطای مدل براساس داده‌های آموزشی کاهش یافته است. ولی برآورد خطای مدل براساس داده‌های آزمایشی ابتدا نزولی بوده و در زمانی که تعداد پارامترهای مدل (پیچیدگی مدل) از میزانی بیشتر شده (مثلا ۲۰ پارامتر) افزایشی شده است. این اثر دقیقا همان مشکل overfitting است که به کمک اعتبارسنجی متقابل سعی در رفع آن می‌کنیم.

تنظیم پارامترها به کمک اعتبارسنجی متقابل

مشاهداتی از جامعه به صورت یک نمونه تصادفی در دسترس است که قرار است از آن‌ها در مدل‌سازی استفاده شود. هدف در اعتبارسنجی متقابل، دستیابی به مدلی است که تعداد پارامترهای آن بهینه باشد. یعنی پیدا کردن مدلی است که دچارoverfitting نباشد. برای برای دستیابی به این هدف در آموزش ماشین (Machine Learning)داده‌ها را به دو قسمت تفکیک می‌کنند.

- قسمت داده‌های آموزشی :(Training set) از این بخش از داده‌ها به منظور ایجاد مدل و برآورد پارامترهای آن استفاده می‌شود.

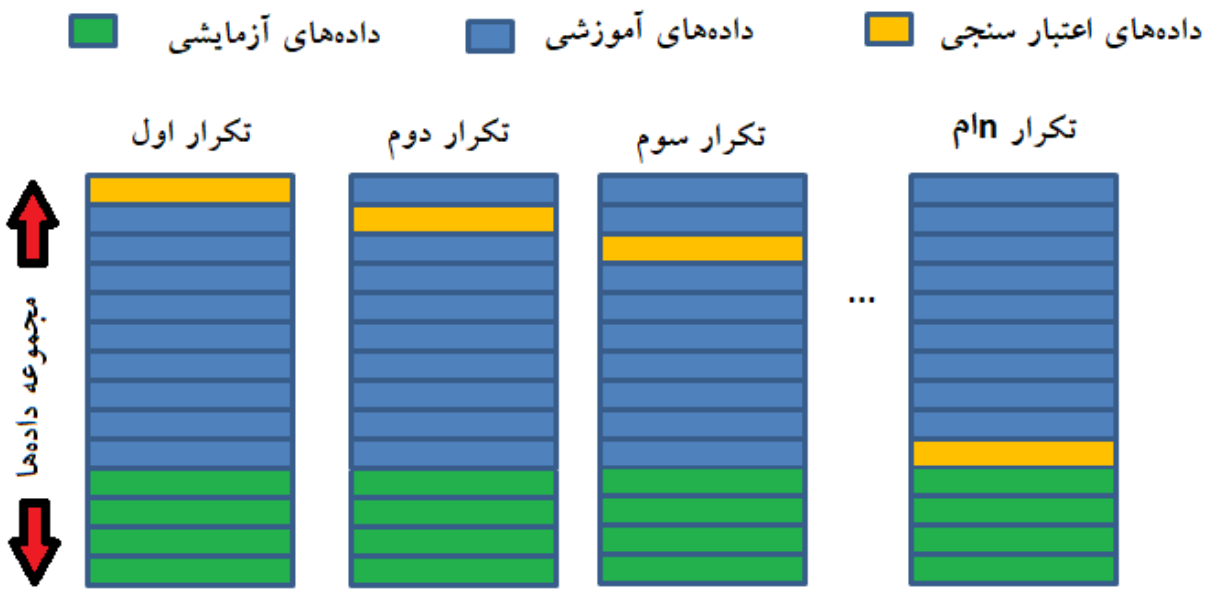
- قسمت داده‌های آزمایشی :(Test set) این قسمت از داده‌ها برای بررسی کارایی مدل استفاده می‌شود. اهمیت این بخش از داده‌ها در این نکته است که این مشاهدات شامل مقدارهای متغیرهای مستقل(x ها) و پاسخی (y) هستند که در مدل به کار نرفته ولی امکان مقایسه مقدار پیش‌بینی شده ($$widehat{y}$$) را با مقدار واقعی به ما می‌دهند.

با توجه به تفکیکی که برای این دو گروه داده در نظر گرفته شد، مدل‌سازی فقط براساس بخش داده‌های آموزشی خواهد بود. ولی در روش اعتبارسنجی متقابل که از این به بعد آن را به اختصار «CV» می‌نامیم، طی یک فرآیند تکرار شونده، قسمت داده‌های آموزشی (Training set) که به منظور مدل‌سازی به کار می رود، خود به دو بخش تفکیک می‌شود. در هر بار تکرار فرآیندCV ، بخشی از داده‌ها برای آموزش و بخشی دیگر برای آزمایش مدل به کار می‌رود. به این ترتیب این فرآیند یک روش بازنمونه‌گیری به منظور برآورد خطای مدل محسوب می‌شود.

نسبت این قسمت‌ها نیز البته جای بحث دارد که در این نوشتار به آن نمی‌پردازیم ولی معمولا ۵۰ درصد کل داده‌ها به منظور آموزش، ۲۵ درصد نیز برای اعتبارسنجی متقابل و مابقی داده‌ها برای آزمایش مدل در نظر گرفته می‌شود.

نکته :باید توجه داشت که داده‌های آزمایشی در فرایند CV ممکن است در تکرار بعدی به عنوان داده‌های آموزشی به کار روند، در نتیجه ماهیت آن‌ها با داده‌هایی که در قسمت قبل به عنوان داده‌های آزمایشی (Test set) معرفی شد، متفاوت است.

مشخص است که داده‌های اعتبارسنجی بخشی از داده‌های آموزشی هستند و داده‌های آزمایشی نیز به عنوان بخشی مجزا از داده‌هایی آموزشی فرض شده‌اند. مراحل تکرار فرآیند CV نیز در تصویر به خوبی دیده می‌شود. نکته دیگری که در تصویر مشخص است، مکمل بودن مجموعه داده‌های آموزشی و اعتبارسنجی است. با انتخاب بخشی از داده‌ها برای انجام فرایندCV ، بقیه داده‌ها برای آموزش به کار گرفته می‌شوند.



در هر مرحله از فرایندCV ، مدل بدست آمده توسط داده‌های آزمایشی برای پیش‌بینی داده‌های CV به کار گرفته و خطا (Error) یا دقت (Accuracy) حاصل از برازش مدل روی داده‌های CV محاسبه می‌شود. معمولا میانگین این خطاها (دقت‌ها) به عنوان خطای (دقت) کلی مدل در نظر گرفته می‌شود. البته بهتر است انحراف معیار خطاها (دقت‌ها) نیز گزارش شود. به این ترتیب با توجه به تعداد پارامترهای مختلف (پیچیدگی مدل)، می‌توان مدل‌های متفاوتی تولید و خطای برآورد آن‌ها را به کمک روش CV اندازه‌گیری کرد. در انتها مدلی را به عنوان مدل مناسب انتخاب خواهیم کرد که دارای کمترین برآورد خطا باشد.

روش‌های مختلف اعتبارسنجی متقابل

براساس شیوه و روش انتخاب مجموعه داده‌های اعتبارسنجی، گونه‌های مختلفی از روش‌های CV معرفی شده‌اند. در ادامه به معرفی بعضی از این گونه‌ها می‌پردازیم.

روش Holdout

در این روش به طور تصادفی، داده‌ها به دو بخش آموزشی و اعتبارسنجی تقسیم می‌شود. پارامترهای مدل توسط داده‌های آموزشی برآورد شده و برآورد خطای مدل نیز براساس داده‌های اعتبارسنجی محاسبه می‌شود.

سادگی محاسبات و عدم تکرار فرآیند CV در این روش از مزیت‌های آن محسوب می‌شود. اگر داده‌های مربوط به بخش آموزش و اعتبارسنجی همگن باشند، این روش مناسب به نظر می‌رسد. ولی از آنجایی که محاسبات خطای مدل براساس فقط یک مجموعه داده، بدست آمده ممکن است برآورد مناسبی برای خطای مدل ارائه نشود.

روش Leave-One-Out

در اینجا به مانند روش بازنمونه‌گیری جک نایف، از مجموعه داده‌های آموزشی یکی مشاهده خارج شده و براساس بقیه مشاهدات، پارامترها برآورد می‌شود. سپس میزان خطای مدل برای مشاهده خارج شده، محاسبه می‌شود. از آنجایی در این روش در هر مرحله از فرآیند CV فقط یکی از مشاهدات خارج می‌شود، تعداد مراحل تکرار فرآیند CVبا تعداد داده‌های آموزشی برابر است. در نتیجه زمان محاسبه خطای مدل کوتاه بوده و به راحتی نیز قابل پیاده‌سازی است. گاهی به اختصار این روش را LOO می‌نامند.

روش Leave-P-Out

اگر در روشLOO ، تعداد مشاهداتی که از مجموعه داده‌های آموزشی خارج می‌شوند برابر با p باشد، آن را روش (Leave-p-Out)یا به اختصار LPO می‌نامند. در نتیجه اگر n تعداد مشاهدات در مجموعه داده‌های آموزشی باشد، تعداد مراحل اجرای فرآیند CV برابر با $${n choose p}$$ خواهد بود.

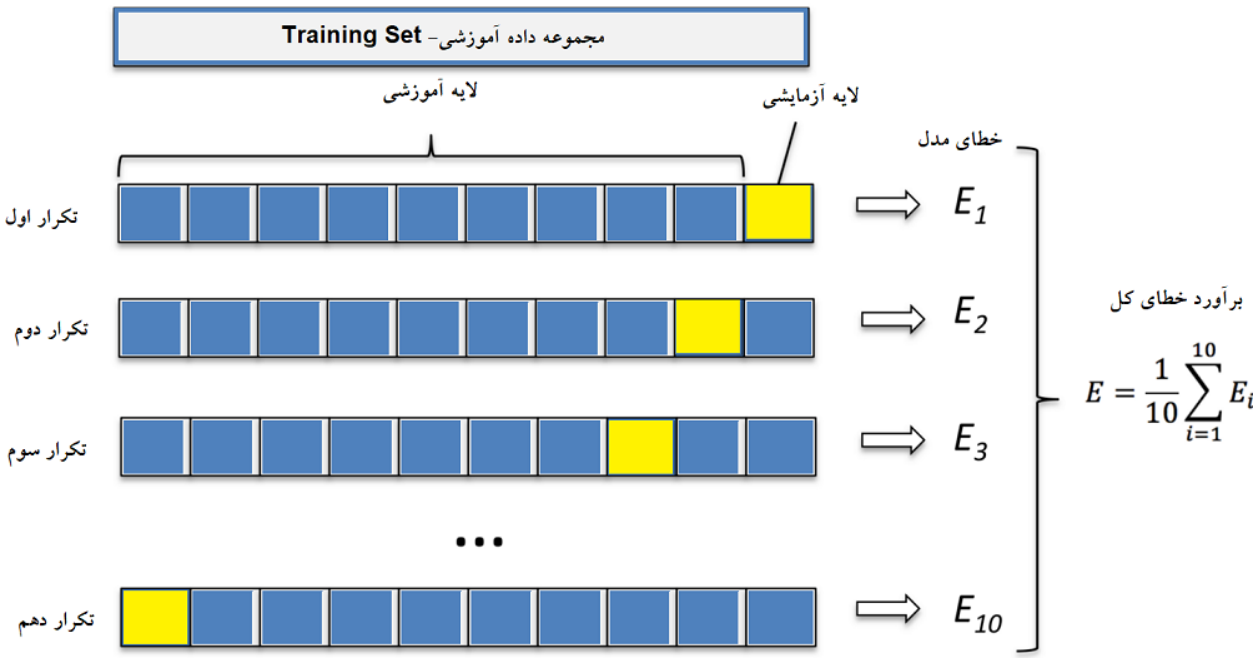
به این ترتیب در هر مرحله از فرآیند، pمشاهده از داده‌های آموزشی، خارج شده و براساس بقیه پارامترها، مدل برآورد می‌شود. سپس خطای مدل برای p مشاهده خارج شده محاسبه می‌شود. در انتها نیز با محاسبه میانگین خطاهای بدست آمده، برآورد خطای مدل حاصل خواهد شد.

برای مثال اگر تعداد مشاهدات برابر با n=100 و p=30 باشد، تعداد مراحل فرآیند CV برابر است با:

$${۱۰۰ choose 30}=3 times 10^{25}$$

روش k-Fold

اگر مجموعه داده‌های آموزشی را به طور تصادفی به k زیرنمونه یا لایه (Fold) با حجم یکسان تفکیک کنیم، می‌توان در هر مرحله از فرایندCV ، تعداد k-1 از این لایه‌ها را به عنوان مجموعه داده آموزشی و یکی را به عنوان مجموعه داده اعتبارسنجی در نظر گرفت. تصویر زیر، مراحل روش k-Fold را به خوبی نشان می‌دهد. مشخص است که با انتخابk=10 ، تعداد تکرارهای فرآیند CV برابر با ۱۰ خواهد بود و دستیابی به مدل مناسب به سرعت امکان‌پذیر می‌شود.



اعتبارسنجی براساس تکرار تصادفی بازنمونه‌گیری

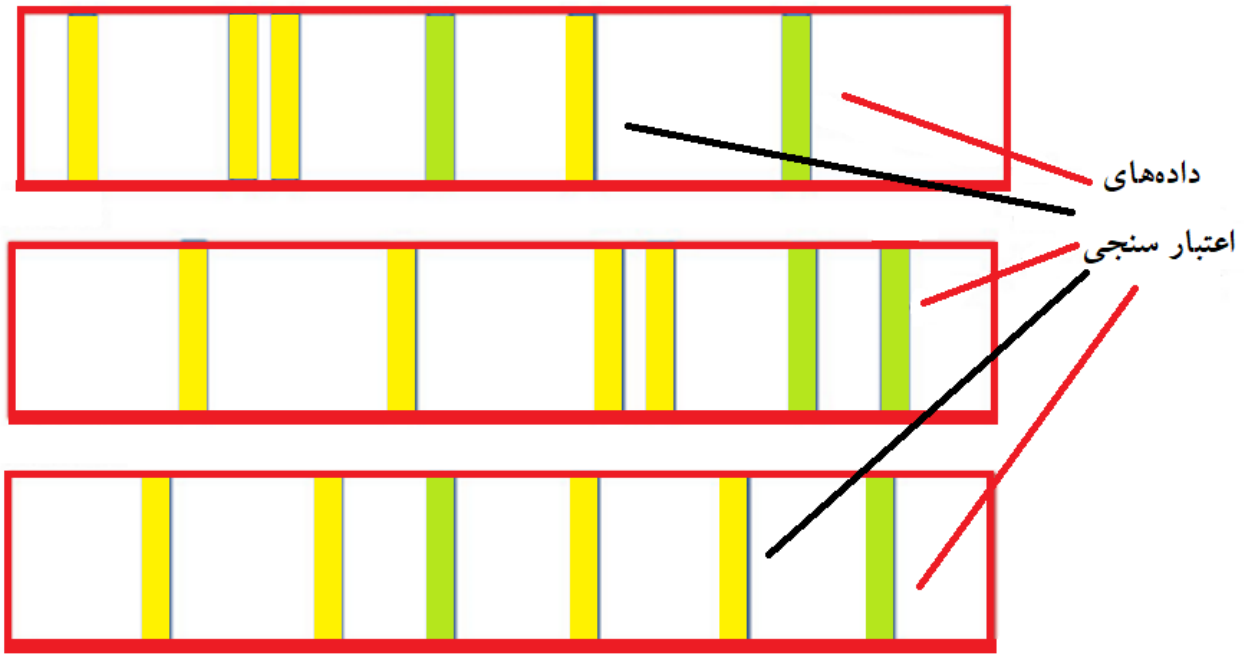
در این روش که گاهی به اعتبارسنجی مونت کارلو (Monte Calro Cross Validation) معروف است، مجموعه داده به طور تصادفی به دو بخش آموزش و اعتبارسنجی تفکیک می‌شود. سپس پارامترهای مدل براساس داده‌های آموزش برآورد شده و خطا یا دقت مدل نیز به کمک داده‌های اعتبارسنجی محاسبه می‌شود. با تکرار تفکیک تصادفی داده‌ها، میانگین خطا یا دقت مدل‌ها، به عنوان معیار انتخاب مدل مناسب در نظر گرفته می‌شود.

با توجه به انتخاب تصادفی داده‌ها، نسبت حجم داده‌های آموزشی و اعتبارسنجی به تعداد تکرارها وابسته نخواهد بود و برخلاف روش k-Fold می‌توان با هر تعداد، فرآیند CV را انجام داد. ولی در عوض به دلیل انتخاب تصادفی

زیرنمونه‌ها، ممکن است بعضی از مشاهدات هرگز در بخش اعتبارسنجی به کار گرفته نشده و بعضی دیگر بیش از یکبار در محاسبات برآورد خطای مدل به کار روند.

اگر تعداد تفکیک‌های تصادفی در این روش به سمت بی‌نهایت میل کند، نتایج حاصل از اعتبارسنجی به سمت نتایج حاصل از روش Leave-P-Out میل می‌کند. همچنین در حالت طبقه‌ای روش مونت کارلو (Stratified Monte Carlo Validation) نمونه‌های تصادفی به شکلی انتخاب می‌شوند که میانگین متغیر پاسخ (مثلا متغیر وابسته در بحث رگرسیون) برای هر زیرنمونه‌ها یکسان باشد. این کار زمانی که متغیر پاسخ به صورت باینری باشد، می‌تواند باعث متعادل شدن نمونه‌ها شود.

برای مثال فرض کنید، تعداد صفرها به طور مشخصی از تعداد ۱ها بیشتر یا کمتر باشد. این حالت زمانی اتفاق می‌افتد که طبقه‌ها از لحاظ تعداد مشاهدات یکسان یا متعادل نباشند. به این ترتیب روش CV طبقه‌ای مونت کارلو، باعث می‌شود که نسبت مشاهدات با مقدار ۰ و ۱ برای متغیر پاسخ، در بین مجموعه داده‌های آموزش و اعتبارسنجی حفظ شود.



روش اعتبار سنجی متقابل مونت کارلو

در تصویر بالا شیوه انتخاب در روش CV مونت کارلو دیده می‌شود. همانطور که دیده می‌شود ممکن است در مراحل انتخاب زیرنمونه‌های تصادفی، گاهی بعضی از مشاهدات انتخاب نشده و گاهی نیز تعداد انتخاب بعضی مشاهدات بیش از یکبار باشد. هر کادر قرمز رنگ نشان دهنده یکبار فرآیند CV است که در آن چندین مشاهده به عنوان زیرنمونه اعتبارسنجی متقابل در نظر گرفته شده است.

مستطیل‌های زرد رنگ مشاهداتی هستند که فقط یکبار در زیرنمونه‌ها دیده شده‌اند و مستطیل‌های سبز رنگ نیز مشاهداتی هستند که دوبار در زیرنمونه‌ها به کار رفته‌اند. مشاهداتی که در قسمت سفید رنگ هستند، داده‌های آموزشی محسوب شده‌اند که ممکن است هرگز در فرآیند CV قرار نگرفته و در برآورد خطای کل نقشی نداشته باشند.

معایب روش اعتبارسنجی متقابل

هرچند این روش، براساس داده‌ها، قادر به برآورد خطای کلی مدل است، ولی بسیار به آن‌ها بخصوص تعداد داده‌ها وابسته است. بنابراین اگر تعداد نمونه یا مشاهدات کاهش یابد، دقت برآورد و یا حتی امکان استفاده از این روش مشکل آفرین خواهد بود. در نتیجه وجود حجم نمونه مناسب و هم‌توزیع بودن داده‌ها در میان مجموعه داده‌های آموزشی، آزمایشی و اعتبارسنجی از اهمیت زیادی در صحت نتایج حاصل از فرآیند CV برخوردار است.

سوال چهارم)

توضیحات مربوط به سوالات این بخش، در فایل سورس کد به صورت متنی ارائه شده است.(برای مشاهده بهتر، فایلی که خروجی html از پیاده سازی است را مطالعه نمایید.)

سوال پنجم)

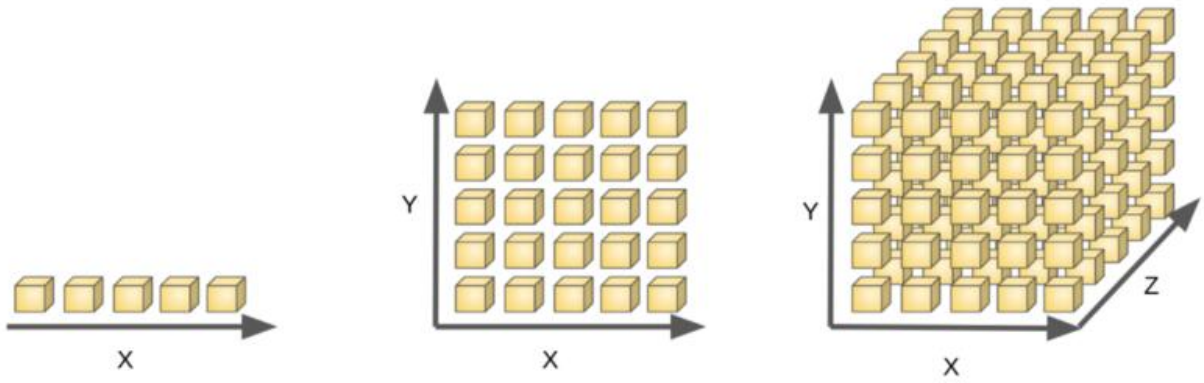
نفرین ابعاد (Curse of Dimensionality) به پدیده‌های گوناگونی اطلاق می‌شود که هنگام تحلیل و ساماندهی داده‌ها در فضاهای با ابعاد بسیار بالا (اغلب با صدها یا هزاران بعد) روی می‌دهند، ولی نه در محیط‌های

با ابعاد بسیار پایین، مانند فضای فیزیکی سه‌بعدی، که در زندگی روزمره احساس می‌کنیم. مضمون مشترک همه این مشکلات آن است که با افزایش ابعاد، حجم فضا آنقدر سریع افزایش می‌یابد که داده‌های موجود پراکنده و تُنُک می‌شوند. این تنکی در هر روشی که مستلزم معنی‌داری آماری است مشکل‌ساز می‌شود. با افزایش ابعاد لازم است داده‌های مورد نیاز برای پشتیبانی از نتیجه هم اغلب به‌طور نمایی افزایش یابند تا نتیجه حاصله از نظر آماری معقول و معتبر باشد. همچنین ساماندهی و جستجوی داده اغلب متکی بر شناسایی ناحیه‌هایی است که در آنجاها اشیاء گروه‌هایی با خواص مشابه تشکیل داده باشند؛ اما در داده‌های با ابعاد بالا، همه اشیاء از بسیاری جهات تُنُک و نامشابه به نظر می‌رسند که این امر از کارایی راهبردهای معمول و متعارف ساماندهی داده‌ها می‌کاهد.

چرا ابعاد زیاد داده نفرین به حساب می آید؟

۱. بار محاسباتی

هرچه ابعاد داده بیش‌تر شود کارکردن با آن نیز دشوارتر می‌گردد، زیرا مانند بسیاری از چالش‌های علوم داده، این نیز به ترکیبات[1] مربوط میشود.



افزایش ابعاد و تاثیر آن بر حجم داده

فرض کنید مجموعه‌ایی از جعبه‌ها را برای یافتن گنج جستجو می‌کنید وقتی $n=1$, فقط ۵ جعبه برای جستجو دارد. به ازای $n=2$, تعداد جعبه‌ها ۲۵ و به ازای $n=3$ تعداد جعبه‌ها ۱۲۵ است. هرچه n بزرگ‌تر شود پیدا کردن گنج از بین دیگر جعبه های خالی سخت تر می‌شود.

[1] Combinatorics

به طور کلی با n بعد که هر کدام m حالت (state) دارند، m ^ n ترکیب ممکن خواهیم داشت. برای داده‌ها با ابعاد زیاد، ما به سادگی نمی توانیم تمام ترکیبات احتمالی را ملاک کار خود قرار دهیم و می‌بایست بخش زیادی از فضای ویژگی را رها کنیم.

۲. افزونگی ابعاد[۲]

داشتن ابعاد زیاد به این معنی نیست که همه ابعاد سودمند هستند.خیلی اوقات ممکن است که الگوی پایه‌ایی یکسانی را به روش های مختلف اندازه‌گیری کنیم.

۳. جنون هندسی(Geometric insanity)

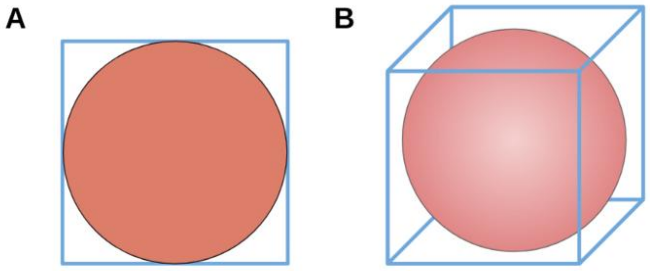
مشکل دیگر ناشی از داده‌های با ابعاد بزرگ مربوط به اثربخشی معیارهای فاصله[۳] مختلف وتکنیک‌های آماری مربوط به آن است. زیرا هندسه در فضای با ابعاد بالا پیچیده میشود. دو مشکل اساسی در هندسه داده‌های با ابعاد بالا ، گریبان گیر ما خواهد بود.

اگر فرض کنیم نقاط داده‌ایی در یک فضای چند بعدی پخش شده باشند، ابعاد بیشتر به معنی فضای بیشتر برای پراکنده شدن داده‌هاست و به این ترتیب پراکندگی داده‌ها از تابع توزیع نرمال پیروی نمی‌کنند و واریانس مناسبی از این توزیع به دست نمی‌آید. پارامتری که قرار بود این واریانس را به ما بدهد، تابع فاصله[۴] نام دارد که اختلاف بین بیش‌ترین فاصله و کم‌ترین فاصله از نقاط داده‌ایی را محاسبه می‌کند.وقتی داده‌ها فضای بزرگتری داشته باشند، این اختلاف بین بیش‌تر داده‌ها مقدار یکسانی می‌شود و به این ترتیب تابع فاصله بی‌استفاده می‌شود زیرا هدف از تعریف این تابع اعمال وجه تمایز بین نقاط داده‌ایی است در حالی که با افزایش ابعاد، این تابع فقط موجب افزونگی[۵] می‌شود. به همین دلیل ماتریسی که شامل محتوای مجموعه داده‌ایی اولیه با ابعاد بالا است، یک ماتریس اسپارس است که بیش‌تر درایه های آن صفر هستند زیرا هر داده به ازای ویژگی‌های خاصی حاوی اطلاعات است و برای مابقی ویژگی‌ها اطلاعاتی ندارد.

دوم تجسم[۶] شکل هندسی با ابعاد بالا غیر ممکن است و امکان درک دیداری برای تحلیل بهتر را از ما میگیرد.

---

[2] Dimensionality redundancy
[3] Distance Metric
[4] Distance Function
[5] Redundancy
[6] Visualization



فضای خالی در مکعب که کره در آن قرار گرفته بیشتر از مربعی است که دایره در آن قرار دارد.

می‌خواهیم کره[۷] را از دید ریاضیات بررسی کنیم.حجم کره با شعاع r و تعداد بعد n از رابطه زیر به‌دست می‌آید:

تابع گاما برای محاسبه حجم ابر کره [۸] در فضای با ابعاد بیش‌تر از سه بعد تعریف می‌شود .

$$\frac{\pi^{n/2} r^n}{\Gamma\left(\frac{n}{2}+1\right)}$$

راه نجات یافتن از این نفرین:

تکنیک‌هایی وجود دارد که به کمک آن‌ها میتوان داده با ابعاد بالا را به داده‌هایی با ابعاد کم‌تر نگاشت کرد. آن چیزی که در این میان اهمیت دارد این است که این تکنیک‌ها ما را به ۳ هدف اصلی برسانند:

- هزینه محاسبات کامپیوتری کاهش پیدا کند چون منابع زمان و حافظه محدود است.
- افزونگی ابعاد [۹] کاهش پیدا کند. یعنی ویژگی‌هایی که تاثیر یکسانی دارند حذف شوند.
- معیارهای فاصله بهبود پیدا کنند تا بتوان واریانس‌های مناسبی برای متمایز کردن داده‌ها از هم پیدا کرد.

تکنیک‌های خطی :

- Multi-Dimensional Scaling
- PCA
- LDA
- Factor Analysis
- Independent Component Analysis

[7] sphere
[8] hyper-sphere

تکنیک‌های غیر خطی :

- ISO map
- Laplacian eigen mapping
- Locally Linear Embedding
- Autoencoders
- t-SNE

سوال ششم)

توضیحات مربوط به سوالات این بخش، در فایل سورس کد به صورت متنی ارائه شده است.(برای مشاهده بهتر، فایلی که خروجی html از پیاده سازی است را مطالعه نمایید.)