استاد: فاطمه منصوری

نیمسال دوم ۱۴۰۲-۱۴۰۳

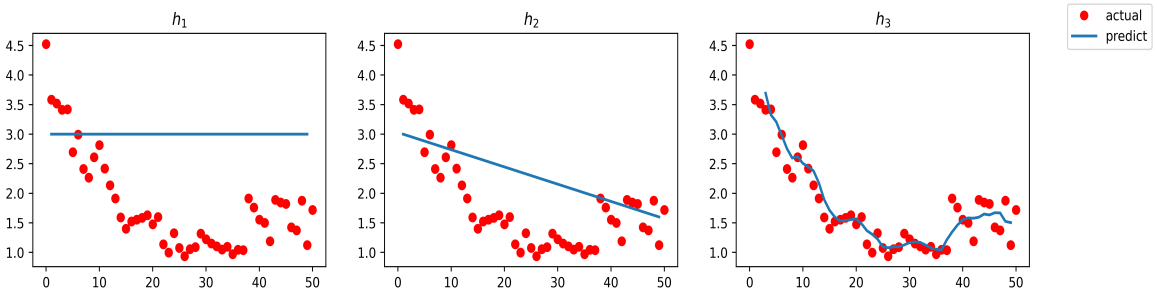
رگرسیون خطی

یادگیری عمیق

گردآورنده: سینا جعفری

۱ تعریف رگرسیون

رگرسیون یک تکنیک یادگیری از نوع نظارت شده است که برچسب آن عددی پیوسته است. همان‌طور که در فصل قبل اشاره شد، یادگیری نظارت شده نوعی از یادگیری است که فرآیند یادگیری در آن با در اختیار داشتن و استفاده از مقدار برچسب صورت می‌گیرد. به عنوان مثال، مجموعه دادگان قیمت خانه‌های کالیفرنیا [1] را درنظر بگیرید. این مجموعه داده شامل ۷ ویژگی است که هدف از طراحی این مجموعه داده، پیش‌بینی متوسط قیمت یک خانه با این ۷ ویژگی است. یا به عنوان مثال، فرض کنید قرار است براساس سن، قد، میزان چربی‌خون و خوابیدن افراد مختلف، مقدار قندخون اشخاص رو پیش‌بینی کنیم. به چنین مسائلی که برچسب آن‌ها عددی پیوسته است، رگرسیون گوییم. درواقع در این روش، قرار است یک خط یا چندجمله‌ای یا ابرصفحه را تخمین بزنیم (شکل ۱).



شکل ۱ : خط پیش‌بینی شده با استفاده سه نوع رگرسیون مختلف که می‌تواند خطی یا غیرخطی باشد.

۲ انواع رگرسیون

رگرسیون را می‌توان از دیدگاه‌های مختلفی دسته‌بندی کرد. یکی از دسته‌بندی‌های مرسوم برای رگرسیون به این صورت است که آن را به دو نوع **خطی** و **غیرخطی** تقسیم می‌کند، که هرکدام را می‌توان به دو نوع دیگر یعنی **تک متغیره** و **چند متغیره** دسته‌بندی کرد. در این فصل قصد داریم رگرسیون خطی و انواع آن را بطور کامل شرح دهیم.

۱.۲ رگرسیون خطی تک متغیره

در فصل اول، تعریف فرضیه [2] را ارائه دادیم. بطور کلی فرضیه به تابعی گفته می‌شود که دو ورودی پارامتر و فضای ویژگی را به فضای برچسب نگاشت می‌کند. اگر یک فرضیه دلخواه را با ϕ نشانگذاری کنیم، می‌توانیم این تابع را به صورت زیر بیان کنیم:

$$\phi : \Theta \times X \longrightarrow \mathbb{R}^C \qquad (۱)$$

[1] https://www.dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html
[2] Hypothesis

که در اینجا، Θ مجموعه پارامترهای مدل، X فضای ویژگی و C تعداد دسته‌های مسئله است که قرار است پیش‌بینی شوند. اگر مسئله از نوع رگرسیون باشد، مقدار $C = ۱$ است. حال می‌خواهیم برای رگرسیون خطی تک متغیره یک فرضیه بنویسیم. از آنجایی که در رگرسیون خطی تک متغیره، تنها یک ویژگی می‌توان در اختیار داشت و قرار است یک خط با توجه به آن ویژگی و برچسب تخمین زده شود، بنابراین فرضیه رگرسیون خطی تک متغیره به فرم معادله یک خط نوشته می‌شود. داریم:

$$h_\theta(x) = \theta_0 + \theta_1 x \qquad (۲)$$

که در اینجا، θ_0 و θ_1 به ترتیب بیانگر عرض از مبدأ و شیب خط تخمین زده و x مقدار ویژگی هستند. طبیعی است که چنین مدلی کاربرد چندانی در دنیای واقعی نخواهد داشت، چراکه تنها از یک ویژگی استفاده می‌کند، که در اکثر مواقع با یک ویژگی نمی‌توان مقدار دقیقی برای برچسب آن مسئله را پیش‌بینی کرد. بنابراین باید چنین فرمی را برای مسائلی که چندین ویژگی دارند نیز بازنویسی کنیم که در بخش بعد آن را توضیح خواهیم داد.

۲.۲ رگرسیون خطی چند متغیره

زمانی که فضای ویژگی مجموعه داده موردنظر، دارای n ویژگی باشد (یعنی فضای ویژگی مجموعه داده، n بُعدی باشد)، رگرسیونی که استفاده می‌شود از نوع چند متغیره است. فرم رابطه ۲ را برای این حالت مشابه رگرسیون خطی تک متغیره، به صورت زیر بازنویسی خواهیم کرد:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \qquad (۳)$$

که در اینجا، $\theta_0 \dots \theta_n$ برابر با پارامترهای مدل و همچنین $x_0 \dots x_n$ به ترتیب برابر با ویژگی اول تا ویژگی nـام است. حال می‌توان، با اضافه کردن یک متغیر ساده مانند x_0 که مقدار آن همیشه برابر با ۱ است، رابطه ۳ را به صورت زیر تغییر داد:

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n \qquad (۴)$$

که در این صورت می‌توانیم رابطه ۴ را به سادگی به فرم ضرب دو بردار تبدیل کنیم. داریم:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \sum_{i=1}^{n} \theta_i x_i = \begin{bmatrix} \theta_0 & \theta_1 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} = \theta^T X \qquad (۵)$$

۳ تابع هزینه

حال باید رابطه ۵ برای مجموعه دادگان آموزشی اعمال و سپس نتایج ارزیابی شود. بنابراین باید یک تابع هزینه تعریف شود تا بتوان این مقادیر پیش‌بینی شده را با نتایج واقعی مقایسه کرد. یکی از معروف‌ترین توابع هزینه، تابع میانگین مربع خطا [3] است که به صورت زیر بیان می‌شود:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 \qquad (۶)$$

که در اینجا، m تعداد نمونه‌های آموزشی و $\hat{y}_i$ نیز بیانگر مقدار پیش‌بینی شده برای نمونه iـام است. می‌توانیم به جای $\hat{y}_i$، مقدار تابع فرضیه را قرار دهیم، بنابراین رابطه ۶ به صورت زیر بازنویسی می‌شود:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2 \qquad (۷)$$

[3] Mean Square Error

اما چگونه می‌توان مقادیر مناسب $\theta_0,\ldots,\theta_n$ که پارامترهای قابل آموزش مدل هستند را انتخاب کرد؟ یکی از الگوریتم‌های معروف برای پیدا کردن مقادیر $\theta_0,\ldots,\theta_n$ ، الگوریتم گرادیان کاهشی [۴] نام دارد که در بخش بعد به آن خواهیم پرداخت.

۴ گرادیان کاهشی

درواقع باید $\theta_0,\ldots,\theta_n$ را طوری انتخاب کرد تا $J(\theta)$ کمینه شود، یعنی بتوانیم پارامترهایی از مدل را پیدا کنیم که کمترین خطا را داشته باشد. بنابراین با یک مسئله بهینه‌سازی روبه‌رو می‌شویم که تابع هدف آن به صورت زیر بیان می‌شود:

$$\min_{\theta_0,\ldots,\theta_n} J(\theta) = \min_{\theta_0,\ldots,\theta_n} \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x_i)-y_i)^2 \qquad (۸)$$

می‌توان نشان داد که حل کردن چنین مسئله‌ای کار چندان راحتی نیست، چراکه با افزایش تعداد پارامترها، فضای جواب مسئله بزرگ‌تر شده و به همان اندازه پیدا کردن پارامترهایی که بتواند مقدار کمینه رابطه ۸ را نتیجه دهد، سخت است. الگوریتم گرادیان کاهشی می‌تواند این مسئله را تا حدودی حل کند. این الگوریتم در ابتدا یک مقدار اولیه برای $\theta_0,\ldots,\theta_n$ انتخاب کرده و سپس در هر گام الگوریتم زیر را برای بروزرسانی پارامترها محاسبه می‌کند.

$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta) \qquad (۹)$$

که در این‌جا، α نرخ‌یادگیری [۵] می‌باشد که آن را یک ابرپارامتر می‌نامیم. این الگوریتم آن‌قدر تکرار می‌شود تا به جواب مسئله همگرا شود. برای آن‌که فهم الگوریتم راحت‌تر شود، الگوریتم گرادیان کاهشی را برای یک مدل رگرسیون چند متغیره پیاده‌سازی می‌کنیم. فرض کنید مقادیر اولیه پارامترها تصادفی باشد، سپس در گام بعد باید عبارت $\frac{\partial}{\partial\theta_j}J(\theta)$ را برای تمامی پارامترها محاسبه کرد و آن‌ها را بروزرسانی کرد. پس داریم:

$$\begin{aligned}\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}(h_\theta(x)-y)^2\\ &= 2.\frac{1}{2}(h_\theta(x)-y).\frac{\partial}{\partial\theta_j}(h_\theta(x)-y)\\ &= (h_\theta(x)-y).\frac{\partial}{\partial\theta_j}(\sum_{i=0}^{d}\theta_i x_i - y)\\ &= (h_\theta(x)-y)x_j\end{aligned}$$

بنابراین گرادیان تابع هزینه نسبت به متغیر θ_j برابر با عبارت فوق خواهد شد. می‌توان این عبارت را در رابطه ۹ جایگذاری کرد، داریم:

$$\theta_j := \theta_j - \alpha(h_\theta(x)-y)x_j \qquad (۱۰)$$

اما این الگوریتم تا چه زمانی باید ادامه پیدا کند؟ اگر فضای مسئله آن‌قدر بزرگ باشد که همگرا شدن به پاسخ بهینه زمان زیادی را از آن خود کند، باید چه پارامتر یا ابرپارامتری را کنترل کرد؟ یکی از مهم‌ترین ابرپارامترهای این الگوریتم، تعداد تکرار [۶] است که بیانگر تعداد دفعات بروزرسانی پارامترها می‌باشد، هرچقدر مقدار این متغیر زیاد باشد، احتمال همگرا شدن بیشتر می‌شود اما اگر فضای مسئله بزرگ و رسیدن به نقطه بهینه سخت باشد، زمان بیشتری صرف پیدا کردن نقطه بهینه می‌شود که این اتفاق خوشایندی نیست.

بنابراین برای آن‌که سرعت همگرا شدن بهبود یابد، می‌توان ابرپارامتری تعریف کرد که وظیفه آن تنظیم سرعت همگرا شدن باشد. نام این ابرپارامتر α می‌باشد که به آن نرخ‌یادگیری گوییم. منظور از نرخ‌یادگیری، سرعت یادگیری مدل یا سرعت همگرا شدن پارامترهای مدل می‌باشد. در انتخاب مقدار این متغیر باید بسیار مراقب بود، چراکه اگر این مقدار خیلی کوچک باشد سرعت یادگیری کاهش پیدا کرده و احتمال همگرا شدن بیشتر می‌شود. اما اگر این مقدار عددی بزرگ باشد، احتمال رسیدن به نقطه بهینه و واگرا شدن مدل زیاد می‌شود،

[4] Gradient Descent
[5] Learning Rate
[6] Number of Iteration

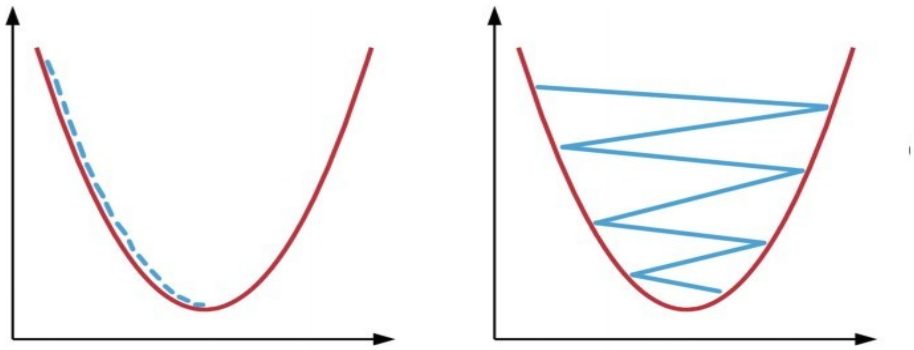
Algorithm ۱: Gradient Descent(x^{train}، y^{train}، N)

Input: x^{train} is equal to the training feature space and y^{train} is equal to the train target، respectively. The value of N is also equal to the number of iteration.

Output: $\Theta = \theta_0, \ldots, \theta_n$ that its equal to model's parameter.

```
۱ begin
۲     Θ ⟵ {r_1, r_2, ..., r_n}                ▷ s.t. r_i selected randomly.
۳     for i = 1 → N do
۴         Θ ⟵ Θ − α ∂/∂Θ J(Θ)                 ▷ s.t. J(Θ) is cost function.
۵     return Θ
```

یعنی در چنین حالتی ممکن است هیچوقت مدل پارامترهای مناسب برای کمینه کردن تابع هزینه را نتواند پیدا کند (شکل ۲). معمولاً مقدار نرخ‌یادگیری عددی بین $۱۰^{-۳}$ تا $۱۰^{-۶}$ می‌باشد، اما از آنجایی که نرخ‌یادگیری یک ابرپارامتر است، نمی‌توان درباره مقدار دقیق و مناسب آن در حالت کلی نظر داد. یکی از راه‌های مناسب برای حل این موضوع، انتخاب نرخ‌یادگیری به صورت انطباقی [7] است، به این صورت که با افزایش تعداد دفعات تکرار الگوریتم، مقدار نرخ‌یادگیری کاهش پیدا کرده و دیگر عددی ثابت درنظر گرفته نمی‌شود. این روش بسیار مناسب بوده و در اغلب سناریوها استفاده می‌شود.



شکل ۲: شمایی از همگرا و واگرا شدن مدل.

[7] Adaptive