

دانشگاه ولی عصر (عج) رفسنجان

نام پروژه : کاربرد آمار در یادگیری ماشین

نام و نام خانوادگی دانشجو : محمد حسین زارعی

استاد راهنما: دکتر عباس مهدوی

پاییز ۱۴۰۴

فهرست مطالب

۱	مقدمه	۲
۲	بررسی عملکرد مدل	۳
۱.۲	اورفیتینگ (overfitting)	۳
۲.۲	اندرفیتینگ (Underfitting)	۴
۳.۲	فیتینگ خوب (Good Fitting)	۵
۳	انواع یادگیری	۶
۱.۳	یادگیری نظارت شده	۶
۲.۳	رگرسیون در یادگیری نظارت شده	۸
۱.۲.۳	رگرسیون خطی ساده	۸
۲.۲.۳	رگرسیون خطی چند متغیره	۱۱
۳.۲.۳	گرادیان (gradeint)	۱۶
۴.۲.۳	رگرسیون لجستیک	۱۸
۵.۲.۳	نظم‌دهی (Regularization)	۲۲
۶.۲.۳	رگرسیون لاسو	۲۳
۷.۲.۳	رگرسیون ریج	۲۴
۳.۳	یادگیری بدون نظارت	۲۵
۱.۳.۳	K-means	۲۶

یادگیری ماشین شاخه‌ای از هوش مصنوعی است که به ماشین‌ها امکان می‌دهد از داده‌ها یاد بگیرند و رفتارهای هوشمندانه‌ای مشابه انسان‌ها بروز دهند، بدون اینکه به صورت صریح برنامه‌ریزی شوند. این فناوری به ماشین‌ها اجازه می‌دهد با استفاده از الگوریتم‌های یادگیری، الگوها و روابط موجود در داده‌ها را شناسایی کرده و بر اساس آن‌ها تصمیم‌گیری کنند یا به محیط اطراف خود تأثیر بگذارند. در یک نگاه ساده، یادگیری ماشین می‌تواند به دو صورت عمل کند: در روش سنتی، برنامه‌نویسان با استفاده از عملیات منطقی و ریاضی، برنامه‌ای مشخص را برای ماشین طراحی می‌کنند. برای مثال، در مدیریت موجودی کارخانه، برنامه‌نویس با دریافت فرمول مقدار اقتصادی سفارش، برنامه نویسی می‌کند. که با تحلیل داده‌های موجودی و پارامترهای محیطی، مقدار بهینه سفارش را محاسبه و اعلام می‌کند. اما در رویکرد مدرن یادگیری ماشین، به جای ارائه برنامه‌های از پیش تعیین‌شده، ماشین با قرار گرفتن در معرض داده‌ها و بهره‌گیری از الگوریتم‌های یادگیری، به تدریج مدل‌های مورد نیاز خود را شکل می‌دهد. این فرایند مشابه یادگیری انسان‌ها، مانند کودکی است که به تدریج زبان را می‌آموزد و توانایی صحبت کردن را کسب می‌کند.

یادگیری ماشین (Machine Learning) در سال‌های اخیر پیشرفت‌های چشمگیری داشته و در حوزه‌های مختلف کاربردهای پیشرفته‌ای پیدا کرده است. در ادامه، به برخی از مثال‌های پیشرفته و نوآورانه یادگیری ماشین اشاره می‌کنم که نشان‌دهنده قدرت و تنوع این فناوری هستند:

۱. تشخیص و درمان پزشکی پیشرفته: - تشخیص سرطان با دقت بالا: الگوریتم‌های یادگیری عمیق (Deep Learning) مانند شبکه‌های کانولوشنی (CNN) برای تحلیل تصاویر پزشکی (CT, MRI، و ماموگرافی) استفاده می‌شوند. برای مثال، مدل‌های AI مانند Google Health's DeepMind در تشخیص سرطان پستان از تصاویر ماموگرافی، گاهی دقت بیشتری نسبت به رادیولوژیست‌های انسانی دارند. - **پیش‌بینی بیماری‌های ژنتیکی** **: ابزارهایی مثل AlphaFold DeepMind ساختار پروتئین‌ها را با دقت بی‌سابقه‌ای پیش‌بینی می‌کنند که در توسعه داروهای جدید برای بیماری‌های پیچیده مانند آلزایمر یا سرطان نقش مهمی دارند.

۲. پردازش زبان طبیعی (NLP): مدل‌های زبانی پیشرفته: مدل‌هایی مانند GPT-4 یا LLaMA با تحلیل حجم عظیمی از داده‌های متنی، توانایی تولید متن‌های شبیه انسان، پاسخ به سؤالات پیچیده، ترجمه زبان‌ها، و حتی نوشتن کد را دارند. این مدل‌ها در چت‌بات‌ها، دستیارهای مجازی، و تحلیل احساسات (Sentiment Analysis) کاربرد دارند. - ترجمه بلادرنگ: سیستم‌هایی مانند Google Translate با استفاده از یادگیری ماشین، ترجمه‌های صوتی و متنی را به صورت بلادرنگ با دقت بالا انجام می‌دهند.

۳. بینایی کامپیوتری: - تشخیص اشیا و چهره: الگوریتم‌های یادگیری عمیق در سیستم‌های امنیتی برای شناسایی چهره‌ها یا تشخیص اشیا در تصاویر و ویدئوها استفاده می‌شوند. مثلاً، سیستم‌های نظارت شهری از این فناوری برای شناسایی رفتارهای مشکوک بهره می‌برند. - خودروهای خودران: شرکت‌هایی مثل Tesla و Waymo از یادگیری ماشین برای پردازش داده‌های حسگرها (LiDAR، دوربین‌ها) استفاده می‌کنند تا خودروها بتوانند موانع، علائم راهنمایی، و مسیرها را شناسایی کرده و رانندگی ایمن انجام دهند.

۴. یادگیری تقویتی در رباتیک: - ربات‌های خودآموز: الگوریتم‌های یادگیری تقویتی (Reinforcement Learning) به ربات‌ها امکان می‌دهند تا از طریق آزمون و خطا وظایف پیچیده‌ای مانند راه رفتن، گرفتن اشیا، یا حتی انجام جراحی را یاد بگیرند. برای مثال، ربات‌های Boston Dynamics از این فناوری برای حرکات پویا و تعادل استفاده می‌کنند. - **بازی‌های استراتژیک** **: مدل‌هایی مثل AlphaGo از DeepMind4 با استفاده از یادگیری تقویتی، استراتژی‌های پیچیده‌ای را در بازی‌های تخته‌ای مانند Go توسعه داده‌اند که حتی قهرمانان انسانی را شکست داده‌اند.

۵. توصیه‌گرهای هوشمند: - سیستم‌های پیشنهاددهنده: پلتفرم‌هایی مانند Netflix، Spotify، و Amazon از الگوریتم‌های یادگیری ماشین (مانند Collaborative Filtering و یادگیری عمیق) برای تحلیل رفتار کاربران و پیشنهاد محتوا یا محصولات شخصی‌سازی شده استفاده می‌کنند. - تبلیغات هدفمند: شرکت‌های تبلیغاتی از یادگیری ماشین برای تحلیل داده‌های کاربران و ارائه تبلیغات متناسب با علایق آن‌ها بهره می‌برند.

۶. پیش‌بینی و تحلیل داده‌های کلان: - پیش‌بینی بازارهای مالی: الگوریتم‌های یادگیری ماشین برای تحلیل روندهای بازار، پیش‌بینی قیمت سهام، و مدیریت ریسک در مؤسسات مالی استفاده می‌شوند. - **مدیریت زنجیره تأمین**^{۳۳}: شرکت‌هایی مانند Walmart از یادگیری ماشین برای بهینه‌سازی موجودی، پیش‌بینی تقاضا، و مدیریت لجستیک استفاده می‌کنند.

۷. کشف دارو و زیست‌فناوری: - طراحی مولکول‌های جدید: AI با شبیه‌سازی تعاملات شیمیایی، ترکیبات دارویی جدید را پیشنهاد می‌دهد. برای مثال، شرکت Insilico Medicine از یادگیری ماشین برای کشف داروهای جدید در زمان کوتاه‌تر استفاده کرده است. - تحلیل داده‌های ژنومی: الگوریتم‌ها برای شناسایی جهش‌های ژنتیکی و پیش‌بینی پاسخ بیماران به درمان‌های خاص استفاده می‌شوند.

۸. کشاورزی هوشمند: - کشاورزی دقیق: یادگیری ماشین با تحلیل داده‌های حسگرها، تصاویر ماهواره‌ای، و داده‌های آب‌وهوایی، به کشاورزان کمک می‌کند تا زمان مناسب کاشت، آبیاری، و برداشت را تعیین کنند. مثلاً، سیستم‌های AI می‌توانند آفات یا بیماری‌های گیاهی را زودهنگام تشخیص دهند. - اتوماسیون مزارع: ربات‌های مجهز به AI برای کاشت، برداشت، و حتی جمع‌آوری داده‌های خاک استفاده می‌شوند.

۹. امنیت سایبری: - تشخیص تهدیدات: الگوریتم‌های یادگیری ماشین برای شناسایی حملات سایبری، بدافزارها، و رفتارهای غیرعادی در شبکه‌ها استفاده می‌شوند. مثلاً، سیستم‌های تشخیص نفوذ (IDS) از AI برای تحلیل ترافیک شبکه بهره می‌برند. - احراز هویت بیومتریک: AI در سیستم‌های تشخیص صدا یا اثر انگشت برای افزایش امنیت کاربرد دارد.

۱۰. هنر و خلاقیت: - تولید محتوا: ابزارهایی مانند DALL-E یا MidJourney با استفاده از یادگیری عمیق، تصاویر، موسیقی، یا حتی داستان‌هایی خلاقانه تولید می‌کنند که مشابه آثار انسانی هستند. - تحلیل سبک‌های هنری: AI می‌تواند آثار هنری را تحلیل کرده و سبک‌های جدید خلق کند یا حتی در مرمت آثار باستانی کمک کند.

ویژگی‌های پیشرفته این کاربردها: - یادگیری عمیق: استفاده از شبکه‌های عصبی عمیق با لایه‌های متعدد برای پردازش داده‌های پیچیده. - تحلیل داده‌های کلان: توانایی پردازش حجم عظیمی از داده‌ها در زمان کوتاه. - تطبیق‌پذیری: قابلیت یادگیری و بهبود مستمر با داده‌های جدید. - تعامل چندحوزه‌ای: ترکیب یادگیری ماشین با حسگرها، رباتیک، و اینترنت اشیا (IoT).

این مثال‌ها تنها بخشی از پتانسیل یادگیری ماشین را نشان می‌دهند. با پیشرفت الگوریتم‌ها و افزایش قدرت محاسباتی، انتظار می‌رود که کاربردهای پیشرفته‌تری در آینده ظهور کنند که زندگی بشر را بیش از پیش متحول کنند.

در یادگیری ماشین، به ویژه در مدل‌های نظارت‌شده مانند رگرسیون خطی، مفاهیم **Overfitting** (اورفیتینگ)، **Underfitting** (اندرفیتینگ) و **Good Fitting** (فیتینگ خوب) نقش کلیدی در ارزیابی و بهبود عملکرد مدل ایفا می‌کنند. این مفاهیم بخشی از مفهوم کلی **Bias-Variance Tradeoff** هستند و نشان‌دهنده تعادل بین پیچیدگی مدل و تعمیم‌پذیری آن می‌باشند. در ادامه، تعاریف رسمی هر کدام را بررسی می‌کنیم.

۲ بررسی عملکرد مدل

۱.۲ اورفیتینگ (overfitting)

Overfitting یکی از چالش‌های اصلی در مدل‌های یادگیری ماشین (به ویژه در یادگیری نظارت‌شده مثل رگرسیون خطی) است. به زبان ساده، **Overfitting** زمانی رخ می‌دهد که مدل بیش از حد به جزئیات و نویزهای داده‌های آموزشی "چسبیده" و الگوهای واقعی را نمی‌تواند به داده‌های جدید تعمیم دهد. یعنی مدل روی داده‌های آموزشی عالی عمل می‌کند (دقت بالا)، اما وقتی داده‌های تست (جدید) می‌آید، عملکردش افت می‌کند مثل دانش‌آموزی که فقط کتاب درسی را حفظ می‌کند، اما سؤال‌های جدید را نمی‌تواند حل کند.

بررسی دقیق تر :

- چه زمانی این اتفاق رخ می دهد؟ مدل پیچیده (مثل رگرسیون با ویژگی های زیاد یا درخت تصمیم عمیق) سعی می کند همه نقاط های آموزشی را "کامل" فیت کند، حتی نویزها (outliers) یا الگوهای تصادفی. این باعث می شود مدل "یاد بگیرد" اما "درک نکند".

- نشانه ها:

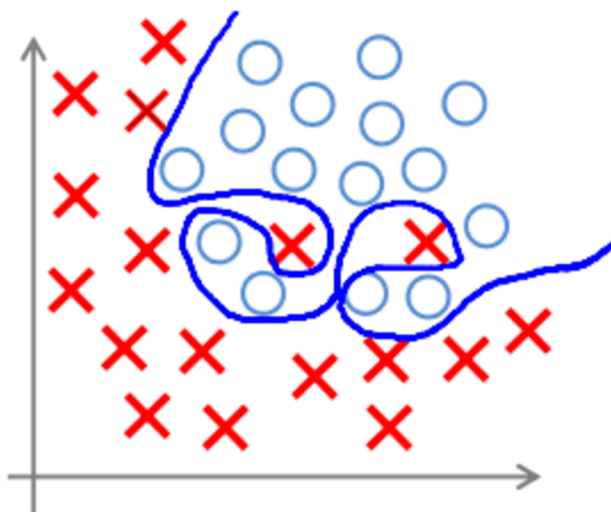
- دقت روی داده های آموزشی (Train Accuracy) بالاست (مثل ۹۹٪)، اما روی داده های تست (Test Accuracy) پایین است (مثل ۷۰٪).

- منحنی یادگیری: خط Train Error کم است، اما Test Error بالاست.

- ارتباط با آمار: در رگرسیون خطی، اگر مدل بیش از حد به داده های آموزشی وابسته شود، MSE روی Train کم می شود اما روی Test زیاد. آمار این را با مفهوم "واریانس بالا" (High Variance) توضیح می دهد.

مثال عملی فرض کنید می خواهیم قیمت خانه را بر اساس مساحت پیش بینی کنیم: با مدل اورفیت، دقت Train: ۹۸٪، اما برای خانه جدید پیش بینی عجیبی می کند (دقت Test: ۶۰٪).

شکل زیر نمونه ای واضح از Overfitting (بیش برازش) را نشان می دهد. در این مدل، خط رگرسیون (الگو) بیش از حد به داده های آموزشی (نقاط آبی) وابسته شده و منحنی را طوری فیت کرده که جزئیات و نویزهای تصادفی آنها را نیز بازتاب می دهد، اما الگوهای کلی داده های تست (نقاط قرمز) را به درستی شناسایی و تعمیم نمی دهد. این وضعیت، مانند حفظ طوطی وار اطلاعات بدون درک واقعی، باعث می شود مدل نتواند پیش بینی دقیقی برای داده های جدید ارائه کند و عملکرد آن در محیط های واقعی افت کند.



شکل ۱: overfit

۲.۲ اندرفیتینگ (Underfitting)

Underfitting زمانی رخ می دهد که مدل بیش از حد ساده باشد و نتواند الگوهای اصلی داده های آموزشی را هم یاد بگیرد. یعنی مدل نه تنها به داده های جدید تعمیم نمی دهد، بلکه حتی روی داده های آموزشی هم ضعیف است. این مثل دانش آموزی است که درس را اصلاً نخوانده و نه سؤال های کتاب را بلد است، نه سؤال های امتحان. **بررسی دقیق تر**

- چرا اتفاق می افتد؟ مدل پیچیدگی کافی ندارد (مثل رگرسیون خطی ساده برای داده های غیرخطی)، یا ویژگی های ورودی کم/نامناسب هستند.

- نشانه ها:

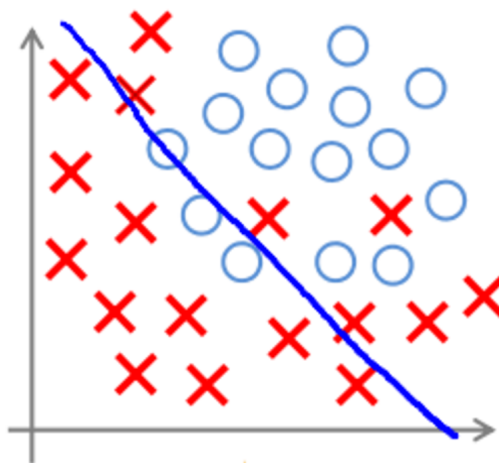
- خطای Train Error و Test Error هر دو بالا (مثل $MSE > 10$).

- دقت پایین روی هر دو مجموعه (Accuracy $< 50\%$).

- ارتباط با آمار: واریانس کم (Low Variance) اما بایاس بالا (High Bias) مدل فرضیات اشتباهی دارد.

مثال عملی: در پیش‌بینی قیمت خانه، اگر مدل فقط مساحت را در نظر بگیرد و عوامل دیگر را نادیده، پیش‌بینی‌ها برای همه داده‌ها ضعیف است (Train Error: 20% , Test Error: 22%).

نمودار زیر نمونه‌ای واضح از Underfitting (اندرفیتینگ یا کم‌برازش) را نشان می‌دهد. در این مدل، خط رگرسیون (الگو) آن‌قدر ساده است که نه تنها الگوهای اصلی داده‌های آموزشی (نقاط آبی) را به خوبی شناسایی نمی‌کند، بلکه داده‌های تست (نقاط قرمز) را نیز به درستی تفکیک و تعمیم نمی‌دهد. این وضعیت، که ناشی از کمبود پیچیدگی مدل است، باعث کاهش شدید دقت کلی (مانند MSE بالا) و تولید پیش‌بینی‌های پرت (outlier) می‌شود. برای حل این مشکل، می‌توان ویژگی‌های بیشتری اضافه کرد یا به مدل‌های پیچیده‌تری (مانند رگرسیون چندمتغیره) مهاجرت نمود. برای جلوگیری از این مشکل، می‌توان از تکنیک‌هایی مانند منظم‌سازی (Regularization) یا اعتبارسنجی متقابل (Cross-Validation) استفاده کرد.



شکل ۲: underfit

۳.۲ فیتینگ خوب (Good Fitting)

Good Fitting حالت ایده‌آل مدل است، جایی که تعادل بین پیچیدگی و تعمیم‌پذیری برقرار است. مدل الگوهای اصلی را خوب یاد می‌گیرد، بدون اینکه به نویز بچسبد. یعنی دقت روی Train و Test مشابه و بالاست مثل دانش‌آموزی که درس را خوب فهمیده و هم تمرین‌ها را حل می‌کند، هم امتحان را.

بررسی دقیق‌تر:

- چرا مهم است؟ مدل واقعی و قابل اعتماد می‌شود، بدون هدررفت منابع.

- نشانه‌ها:

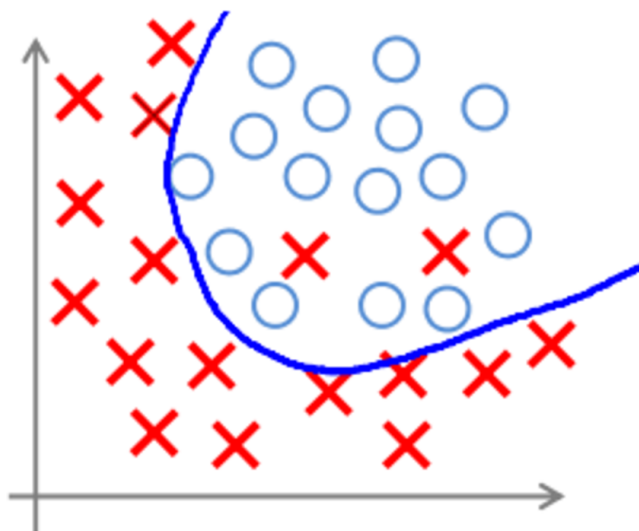
- Train Error و Test Error نزدیک هم و کم.

- R^2 بالا (نزدیک ۱) و تعمیم‌پذیری خوب.

- ارتباط با آمار: بایاس و واریانس هر دو پایین (Bias-Variance Tradeoff) مدل بهینه است.

مثال عملی در پیش‌بینی قیمت خانه، مدل با ویژگی‌های مناسب که Train Error : ۵٪ و Test : ۶٪ پیش‌بینی دقیق برای خانه‌های جدید.

نمودار زیر نمونه‌ای واضح از **Good Fitting** (فیتینگ خوب یا برازش مناسب) را نشان می‌دهد. در این مدل، خط رگرسیون (الگو) با تعادل ایده‌آل، الگوهای اصلی داده‌های آموزشی (نقاط آبی) را به خوبی شناسایی و فیت می‌کند، و همزمان داده‌های تست (نقاط قرمز) را به درستی تفکیک و تعمیم می‌دهد. این وضعیت، که نشان‌دهنده درک واقعی مدل از روابط داده‌ها بدون وابستگی به نویز است، بیانگر آن است که مدل نه تنها "حفظ" کرده، بلکه "یاد گرفته" است با دقت کلی بالا (مانند R^2 نزدیک به ۱) و عملکرد پایدار در پیش‌بینی داده‌های جدید. برای دستیابی به این حالت، نظارت بر **Bias-Variance Tradeoff** و استفاده از تکنیک‌هایی مانند اعتبارسنجی متقابل (**Cross-Validation**) ضروری است.



شکل ۳: overfit

۳ انواع یادگیری

۱.۳ یادگیری نظارت شده

در یادگیری ماشین با دو نوع یادگیری مواجه می‌شویم، که یک نوع آن یادگیری نظارت‌شده (**Supervised Learning**) یکی از شاخه‌های اصلی یادگیری ماشین است که در آن مدل با استفاده از داده‌های برچسب‌دار آموزش می‌بیند. در این روش، داده‌های ورودی (ویژگی‌ها) همراه با خروجی‌های متناظر (برچسب‌ها) به مدل ارائه می‌شوند تا مدل بتواند رابطه بین ورودی‌ها و خروجی‌ها را یاد بگیرد و از آن برای پیش‌بینی خروجی‌های جدید استفاده کند. مفاهیم اصلی:

- داده‌های برچسب‌دار: داده‌های آموزشی شامل دو بخش هستند:

ویژگی‌ها، برچسب

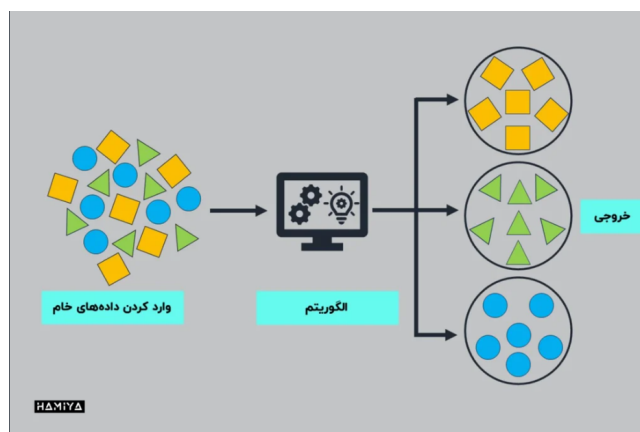
- (**Features**): متغیرهای ورودی که مدل از آن‌ها برای یادگیری استفاده می‌کند (مثل قد، وزن، یا دما).

- (**Labels**) برچسب : خروجی‌های مورد انتظار که مدل باید پیش‌بینی کند (مثل "مثبت" یا "منفی" در طبقه‌بندی، یا یک عدد در رگرسیون). انواع مسائل در یادگیری نظارت‌شده:

- رگرسیون (**Regression**) : پیش‌بینی یک مقدار پیوسته، مثل پیش‌بینی قیمت خانه یا دمای هوا.

- طبقه‌بندی (Classification): پیش‌بینی یک دسته یا کلاس، مثل تشخیص ایمیل به‌عنوان "هرزنامه" یا "غیرهرزنامه". مراحل یادگیری نظارت‌شده:
- جمع‌آوری داده: تهیه مجموعه داده‌ای با ویژگی‌ها و برچسب‌های دقیق.
- آموزش مدل: استفاده از الگوریتم‌هایی مثل رگرسیون خطی، درخت تصمیم، یا شبکه‌های عصبی برای یادگیری رابطه بین ویژگی‌ها و برچسب‌ها.
- ارزیابی مدل: بررسی عملکرد مدل با معیارهایی مثل دقت (Accuracy)، خطای میانگین مربعات (MSE) یا ماتریس درهم‌ریختگی (Confusion Matrix).
- پیش‌بینی: استفاده از مدل آموزش‌دیده برای پیش‌بینی برچسب داده‌های جدید. لگوریتم‌های رایج: رگرسیون خطی (Linear Regression) - رگرسیون لجستیک (Logistic Regression) - ماشین بردار پشتیبان (SVM) - درخت تصمیم (Decision Tree) - جنگل تصادفی (Random Forest) - شبکه‌های عصبی (Neural Networks)

به منظور درک بهتر روش یادگیری نظارت شده می‌توانیم از یک مثال ساده کمک بگیریم. فرض کنید یک مجموعه داده ورودی از تصاویر اشکال مختلف داریم. قصد داریم مدلی را با رویکرد یادگیری نظارت شده با استفاده از این مجموعه داده آموزش دهیم تا در نهایت بتواند با دریافت تصویر جدید، شکل موجود در تصویر را تشخیص دهد.



بدین منظور، برچسب‌های داده‌های آموزشی مدل باید تعیین شوند که برای این مثال، هر یک از برچسب‌ها نام اشکال موجود در تصاویر را مشخص می‌کنند. به عبارتی، در مثال تعریف شده، سه برچسب مربع، مثلث و شش ضلعی وجود دارد. با استفاده از اطلاعات استخراج شده از تصاویر مانند شکل و تعداد اضلاع و به همراه برچسب تصاویر می‌توان مدل یادگیری ماشین را آموزش داد.

پس از اتمام آموزش، تصویر یک شکل جدید را به مدل می‌دهیم و از آن می‌خواهیم که شکل موجود در تصویر را تشخیص دهد. مدل بر اساس ویژگی‌هایی که در مرحله آموزش یاد گرفته است، ویژگی‌های تصویر جدید را بررسی می‌کند و سپس خروجی را ارائه می‌دهد.

یادگیری نظارت شده را که جزو انواع روش‌های ماشین لرنینگ محسوب می‌شود، می‌توان در حل دو نوع مسئله هوش مصنوعی به کار برد که در ادامه به آن‌ها اشاره شده است.

مزایا: - دقت بالا در صورتی که داده‌های برچسب‌دار کافی و باکیفیت وجود داشته باشد. - کاربرد گسترده در مسائل واقعی مثل تشخیص پزشکی، پیش‌بینی مالی، و پردازش زبان طبیعی. چالش‌ها: - نیاز به داده‌های برچسب‌دار که ممکن است جمع‌آوری آن‌ها زمان‌بر و پرهزینه باشد. - حساسیت به داده‌های پرت

۲.۳ رگرسیون در یادگیری نظارت شده

رگرسیون یکی از تکنیک‌های اصلی در یادگیری نظارت شده (Supervised Learning) است که برای پیش‌بینی مقادیر پیوسته (Continuous) استفاده می‌شود. در یادگیری نظارت شده، مدل با استفاده از داده‌های ورودی (ویژگی‌ها) و خروجی‌های متناظر (برچسب‌ها) آموزش می‌بیند تا رابطه‌ای بین ورودی‌ها و خروجی‌ها را یاد بگیرد. در مسائل رگرسیون، خروجی یک مقدار عددی پیوسته است، مانند پیش‌بینی قیمت خانه، دمای هوا، یا میزان فروش. در رگرسیون، متغیر هدف (خروجی) مقدار پیوسته است، برخلاف مسائل طبقه‌بندی که خروجی گسسته (مانند کلاس‌ها) است. هدف: یافتن تابعی که رابطه بین ویژگی‌های ورودی (X) و خروجی (y) را به بهترین شکل مدل کند.

تابع پیش‌بینی: مدل رگرسیون یک تابع $f(X)$ تولید می‌کند که مقادیر پیش‌بینی شده (\hat{y}) را به ازای ورودی‌های جدید تخمین می‌زند. خطا: تفاوت بین مقدار واقعی (y) و مقدار پیش‌بینی شده (\hat{y}) به عنوان خطا محاسبه می‌شود. هدف کمینه کردن این خطا است.

۱.۲.۳ رگرسیون خطی ساده

رگرسیون خطی (Linear Regression) رگرسیون خطی (Linear Regression) یکی از بنیادی‌ترین و پرکاربردترین الگوریتم‌های یادگیری نظارت شده (Supervised Learning) در یادگیری ماشین است که به طور خاص برای پیش‌بینی مقادیر پیوسته (مانند قیمت، دما یا فروش) طراحی شده است. این مدل بر پایه فرض ساده‌ای بنا شده: رابطه بین متغیرهای ورودی (ویژگی‌ها، X) و خروجی (y) به صورت خطی است. برای مثال، اگر بخواهیم قیمت یک خانه را بر اساس دو ویژگی پیوسته مانند مساحت (x_1) و تعداد اتاق‌ها (x_2) پیش‌بینی کنیم، مدل رگرسیون خطی معادله‌ای مانند $\hat{y} = w_0 + w_1x_1 + w_2x_2$ می‌سازد، که w_0 عرض از مبدأ و w_1, w_2 ضرایب شیب هستند. این رویکرد نه تنها ساده و تفسیرپذیر است، بلکه در مسائل واقعی مانند تحلیل املاک (مانند تخمین در پلتفرم‌هایی مثل Zillow) یا پیش‌بینی تقاضای بازار، دقت بالایی ارائه می‌دهد.

در یادگیری نظارت شده، هدف اصلی آموزش یک مدل است که با استفاده از داده‌های آموزشی (مجموعه‌ای از جفت‌های (X, y))، رابطه بین ویژگی‌های ورودی و خروجی را مدل‌سازی کند. داده‌های آموزشی شامل برچسب‌های واقعی (labels) هستند که مدل را هدایت می‌کنند تا الگوها را بیاموزد. رگرسیون خطی این کار را با کمینه‌سازی تابع هزینه (مانند میانگین مربعات خطا، MSE) انجام می‌دهد، تا پیش‌بینی‌ها (\hat{y}) تا حد ممکن به مقادیر واقعی (y) نزدیک باشند. فرض کلیدی مدل، خطی بودن رابطه است یعنی تغییرات خطی در ورودی‌ها، تغییرات متناسب در خروجی ایجاد می‌کند. اگر این فرض برقرار نباشد (مثل روابط غیرخطی)، مدل‌های پیشرفته‌تری مانند رگرسیون چندجمله‌ای یا منظم‌سازی (Ridge/Lasso) پیشنهاد می‌شود. در ادامه، به طور کامل به جنبه‌های مختلف رگرسیون خطی می‌پردازیم: از فرمول‌های ریاضی و روش‌های بهینه‌سازی (مانند گرادینان نزولی) تا چالش‌هایی مانند بیش‌برازش (Overfitting) و راه‌حل‌های آماری آن.

کمینه‌سازی: تابع هزینه معیاری برای سنجش میزان خطای مدل در پیش‌بینی خروجی‌ها نسبت به مقادیر واقعی است. در رگرسیون خطی، تابع هزینه معمولاً میانگین مربعات خطا (Mean Squared Error - MSE) است که تفاوت بین مقادیر واقعی (y_i) و پیش‌بینی شده (\hat{y}_i) را اندازه‌گیری می‌کند. دقت مدل را افزایش دهیم:

کمینه کردن تابع هزینه به معنای کاهش خطای پیش‌بینی است. این کار باعث می‌شود مدل پیش‌بینی‌هایی تولید کند که به مقادیر واقعی نزدیک‌تر باشند. پارامترهای بهینه را پیدا کنید. در رگرسیون خطی، پارامترها (مانند w_0 و w_1) تعیین‌کننده خط پیش‌بینی هستند. کمینه‌سازی تابع هزینه، مقادیر بهینه این پارامترها را پیدا می‌کند تا مدل بهترین تطابق را با داده‌ها داشته باشد.

تعمیم مدل: با کمینه‌سازی تابع هزینه روی داده‌های آموزشی، مدل می‌تواند الگوهای واقعی داده‌ها را یاد بگیرد و روی داده‌های جدید (داده‌های آزمون) عملکرد بهتری داشته باشد، به شرطی که از بیش‌برازش (Overfitting) جلوگیری شود.

معیاری برای مقایسه مدل‌ها: مقدار تابع هزینه می‌تواند به عنوان معیاری برای مقایسه عملکرد مدل‌های مختلف (مثلاً رگرسیون خطی در مقابل رگرسیون چندجمله‌ای) استفاده شود.

پایه‌ای برای الگوریتم‌های پیچیده‌تر: مفهوم کمینه‌سازی تابع هزینه در رگرسیون خطی، پایه‌ای برای یادگیری الگوریتم‌های پیچیده‌تر مانند شبکه‌های عصبی است که از روش‌های مشابه (مثل گرادینان نزولی) برای بهینه‌سازی استفاده می‌کنند.

اثبات کمینه‌سازی تابع هزینه در رگرسیون خطی

$$J(w_0, w_1) = \sum_{i=1}^n (y_i^{(i)} - \hat{y}^{(i)})^2 = \sum_{i=1}^n (y_i^{(i)} - (w_0 + w_1 x_i^{(i)}))^2 \quad (1)$$

که در آن y_i مقدار واقعی، $\hat{y}_i = w_0 + w_1 x_i$ مقدار پیش‌بینی شده می باشد.
کمینه‌سازی تابع هزینه برای کمینه کردن $J(w_0, w_1)$ ، مشتق‌های جزئی نسبت به w_0 و w_1 را محاسبه و صفر می‌کنیم.

مشتق نسبت به w_0

$$\frac{\partial J}{\partial w_0} = \frac{\partial}{\partial w_0} \left[\sum_{i=1}^n (y_i^{(i)} - (w_0 + w_1 x_i^{(i)}))^2 \right]$$

$$-2 \sum_{i=1}^n (y_i^{(i)} - w_0 - w_1 x_i^{(i)}) = 0$$

$$nw_0 + w_1 \sum_{i=1}^n x_i = \sum_{i=1}^n y_i$$

$$w_0 = \frac{\sum_{i=1}^n y_i - w_1 \sum_{i=1}^n x_i}{n} = \bar{y} - w_1 \bar{x}$$

که در آن $\bar{y} = \frac{\sum y_i}{n}$ میانگین داده‌ها هستند و $\bar{x} = \frac{\sum x_i}{n}$

مشتق نسبت به w_1

$$\frac{\partial J}{\partial w_1} = \frac{\partial}{\partial w_1} \left[\sum_{i=1}^n (y_i^{(i)} - (w_0 + w_1 x_i^{(i)}))^2 \right]$$

$$= \sum_{i=1}^n 2(y_i^{(i)} - w_0 - w_1 x_i^{(i)})(-x_i^{(i)}) = \sum_{i=1}^n x_i (y_i^{(i)} - w_0 - w_1 x_i^{(i)})$$

صفر کردن مشتق و جایگذاری $w_0 = \bar{y} - w_1 \bar{x}$:

$$\sum_{i=1}^n x_i^{(i)} (y_i^{(i)} - (\bar{y} - w_1 \bar{x}) - w_1 x_i^{(i)}) = 0$$

$$\sum_{i=1}^n x_i^{(i)} (y_i^{(i)} - \bar{y} - w_1 (x_i^{(i)} - \bar{x})) = 0$$

$$\sum_{i=1}^n x_i^{(i)} (y_i^{(i)} - \bar{y}) - w_1 \sum_{i=1}^n x_i^{(i)} (x_i^{(i)} - \bar{x}) = 0$$

حل برای w_1 :

$$w_1 = \frac{\sum_{i=1}^n x_i^{(i)} (y_i^{(i)} - \bar{y})}{\sum_{i=1}^n x_i^{(i)} (x_i^{(i)} - \bar{x})}$$

برای ساده‌سازی، از فرم معادل استفاده می‌کنیم:

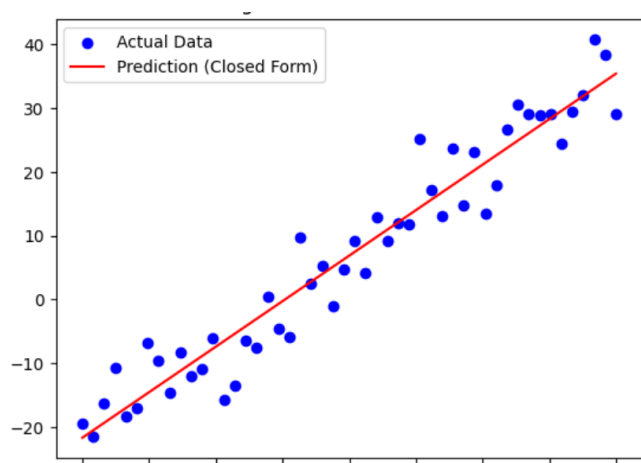
$$w_1 = \frac{\sum_{i=1}^n (x_i^{(i)} - \bar{x})(y_i^{(i)} - \bar{y})}{\sum_{i=1}^n (x_i^{(i)} - \bar{x})^2}$$

برآوردگرهای بهینه بنابراین، برآوردگرهای بهینه برای پارامترها عبارت‌اند از:

$$\hat{w}_1 = \frac{\sum_{i=1}^n (x_i^{(i)} - \bar{x})(y_i^{(i)} - \bar{y})}{\sum_{i=1}^n (x_i^{(i)} - \bar{x})^2} \quad (2)$$

$$\hat{w}_0 = \bar{y} - w_1 \bar{x} \quad (3)$$

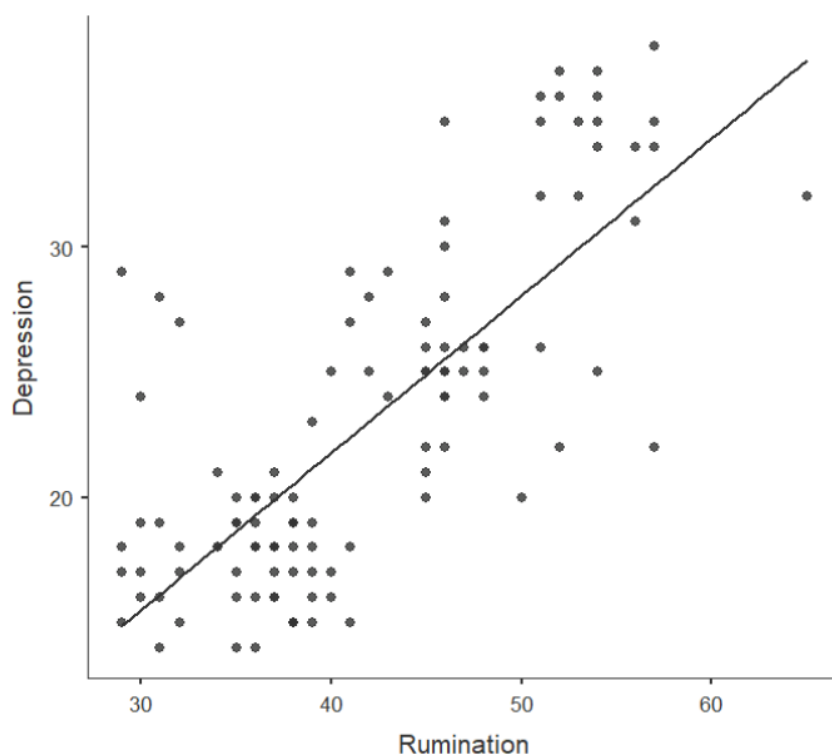
تأثیر کمینه‌سازی بر مدل خطی: نمودار زیر نمونه‌ای واضح از تأثیر مثبت کمینه‌سازی در رگرسیون خطی ساده را نشان می‌دهد. در این مدل، خط پیش‌بینی (قرمز) با استفاده از روش کمینه‌سازی Least Squares، الگوهای اصلی داده‌های واقعی (نقاط آبی) را به خوبی شناسایی و فیت می‌کند. این رویکرد، که معادلات نرمال را حل می‌کند، تعادل مناسبی بین دقت و تعمیم‌پذیری ایجاد کرده و از Overfitting یا Underfitting جلوگیری می‌نماید.



شکل ۴: overfit

نمودار زیر تأثیر کمینه‌سازی را در داده‌های واقعی‌تر (مانند رابطه افسردگی و نشخوار فکری) نشان می‌دهد. در اینجا، خط رگرسیون (سیاه) با استفاده از کمینه‌سازی استاندارد، الگوهای کلی را شناسایی می‌کند، اما به دلیل سادگی مدل، نتوانسته داده‌ها را به طور کامل تفکیک کند. نشانه‌ای از Underfitting، این وضعیت، که در آن MSE هنوز بالاست، بیانگر نیاز به مدل‌های پیشرفته‌تر (مانند رگرسیون چندمتغیره یا منظم‌سازی) است.

Scatterplot



شکل ۵: overfit

۲.۲.۳ رگرسیون خطی چند متغیره

(Multiple Linear Regression) یکی از روش‌های کلیدی در یادگیری نظارت‌شده است که برای پیش‌بینی یک متغیر وابسته (خروجی پیوسته) بر اساس چندین متغیر مستقل (ویژگی‌ها) استفاده می‌شود. این روش زمانی کاربرد دارد که بخواهیم رابطه‌ای خطی بین چندین متغیر ورودی و یک خروجی پیوسته را مدل کنیم. در یادگیری نظارت‌شده، داده‌های آموزشی شامل جفت‌های ورودی (ویژگی‌ها) و خروجی (برچسب‌ها) هستند، و رگرسیون خطی چندگانه با یادگیری این رابطه، پیش‌بینی‌های دقیقی برای داده‌های جدید ارائه می‌دهد. در ادامه، به کاربردهای این روش و فرضیات آن می‌پردازیم.

رگرسیون چندگانه به ما کمک می‌کند تا اثر هر متغیر مستقل بر خروجی را درک کنیم. ضرایب مدل نشان‌دهنده میزان تأثیر هر ویژگی هستند.

مثال: در بازاریابی، می‌توان تأثیر بودجه تبلیغات، نوع رسانه، و فصل فروش بر میزان فروش را بررسی کرد. کاربردهای گسترده در حوزه‌های مختلف:

اقتصاد: پیش‌بینی رشد اقتصادی یا تورم بر اساس متغیرهایی مانند نرخ بهره، تولید ناخالص داخلی، و بیکاری.

پزشکی: پیش‌بینی فشار خون بیمار بر اساس سن، وزن، سطح فعالیت بدنی، و رژیم غذایی.

علوم طبیعی: پیش‌بینی دمای هوا با استفاده از متغیرهایی مانند رطوبت، فشار جو، و سرعت باد.

بازاریابی و تجارت: پیش‌بینی فروش محصول بر اساس قیمت، تبلیغات، و رفتار مشتری.

مهندسی: پیش‌بینی مصرف انرژی یک ساختمان بر اساس اندازه، تعداد ساکنان، و نوع عایق‌بندی. پایه‌ای برای مدل‌های پیچیده‌تر:

رگرسیون خطی چندگانه به دلیل سادگی و تفسیرپذیری، پایه‌ای برای درک مدل‌های پیشرفته‌تر مانند رگرسیون منظم‌شده (ریج، لاسو) یا شبکه‌های عصبی است.

این روش به‌عنوان یک معیار اولیه (Baseline) برای مقایسه با مدل‌های پیچیده‌تر استفاده می‌شود.

تصمیم‌گیری مبتنی بر داده:

با ارائه پیش‌بینی‌های دقیق، رگرسیون چندگانه به تصمیم‌گیری در حوزه‌هایی مانند مدیریت ریسک، برنامه‌ریزی مالی، و بهینه‌سازی منابع کمک می‌کند.

مثال: پیش‌بینی تقاضای محصول برای مدیریت موجودی در زنجیره تأمین.

تحلیل روابط پیچیده:

این روش امکان مدل‌سازی تعاملات بین متغیرهای مختلف را فراهم می‌کند، به‌ویژه زمانی که یک متغیر به‌تنهایی نمی‌تواند خروجی را توضیح دهد.

مثال: پیش‌بینی عملکرد تحصیلی دانش‌آموز بر اساس ساعات مطالعه، کیفیت تدریس، و سطح استرس.

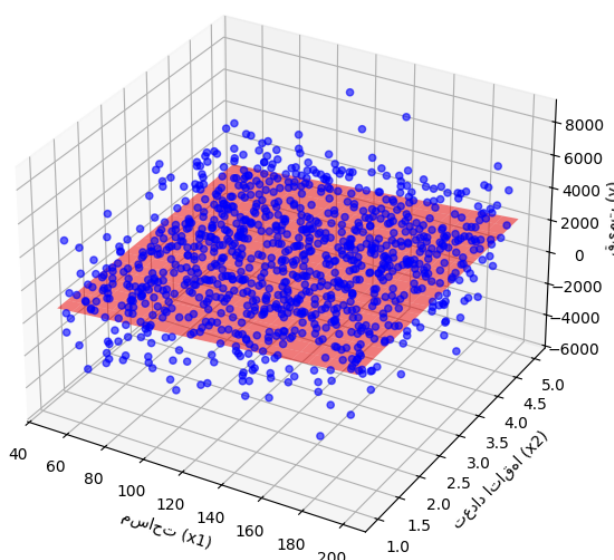
بررسی عملکرد در مدل رگرسیون خطی

نمودار زیر نمونه‌ای از اندرفیتینگ (Underfitting) در رگرسیون خطی چندمتغیره را نشان می‌دهد. در این مدل، سطح مدل (صفحه قرمز) تقریباً صاف است و تغییرات پیچیده دیتای سه‌بعدی را دنبال نمی‌کند. نقاط آبی (داده‌های واقعی) در اطراف این صفحه پراکندگی زیادی دارند و مدل الگوی واقعی داده را یاد نگرفته است. اگر مدل خوب بود، صفحه باید کمی خمیدگی یا شیب متناسب با داده نشان می‌داد، اما اینجا فقط یک صفحه ساده رسم شده است.

نشانه‌های اندرفیتینگ

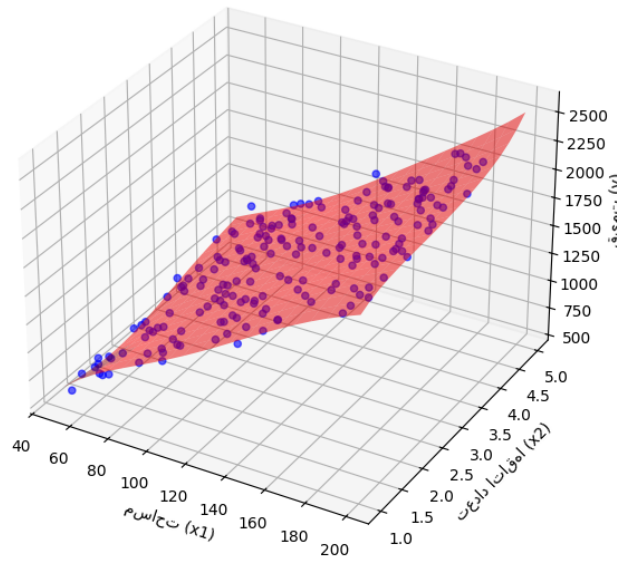
- صفحه قرمز (مدل رگرسیون) تقریباً صاف است و تغییرات پیچیده دیتای سه‌بعدی را دنبال نمی‌کند.
- نقاط آبی در اطراف این صفحه پراکندگی زیادی دارند و مدل الگوی واقعی داده را یاد نگرفته است.
- اگر مدل خوب بود، صفحه باید کمی خمیدگی یا شیب متناسب با داده نشان می‌داد، اما اینجا فقط یک صفحه ساده رسم شده است.

Good Fit (به نوبت): هر یک از نمودارهای سطح نویس‌رگر



شکل ۶: اندرفیتینگ در رگرسیون خطی چندمتغیره: صفحه قرمز صاف (مدل ساده) و پراکندگی نقاط آبی (residuals بزرگ).

ان وضعیت، MSE بالا و R^2 پایین را نتیجه می‌دهد، که با مفهوم High Bias در آمار همخوانی دارد. نمودار زیر نمونه‌ای از overfitting در رگرسیون خطی چندمتغیره را نشان می‌دهد: سطح مدل (قرمز/صورتی) منحنی و پیچیده شده تا به نقاط آبی (train data) بچسبد، اما تعمیم ضعیف است (high variance).



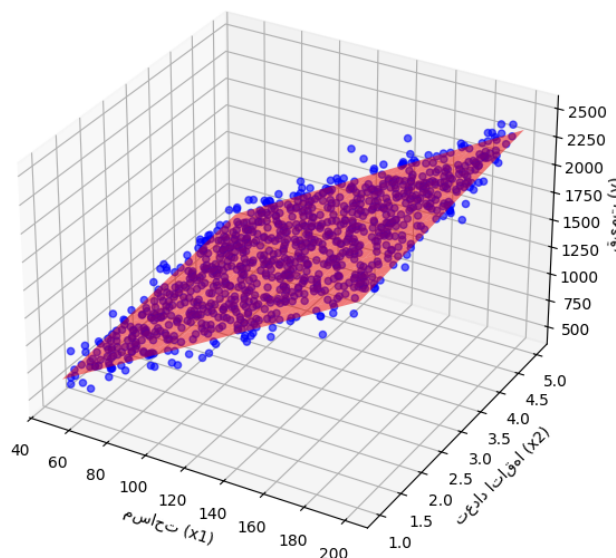
شکل ۷: Overfitting: سطح پیچیده چسبیده به نقاط آبی (Train MSE کم، Test بالا). محورها: x_1 (۴۰-۲۰۰)، x_2 (۱-۵)، y (۵۰۰-۲۵۰۰).

نشانه‌های اورفیتینگ:

- سطح مدل منحنی و پیچیده است و به نویز train data (نقاط آبی) چسبیده، نه الگوهای کلی.
- پراکندگی کم روی train (train MSE ≈ 0)، اما test data دور می‌ماند ($R^2 \text{ test} > 0.7$).
- مدل over-complex (بدون regularization)، تعمیم ضعیف high variance.
- برای حل، regularization (Lasso/Ridge، بخش ۵.۱.۲) سطح را صاف می‌کند.

نمودار زیر نمونه‌ای از Good Fit (برازش مناسب) در رگرسیون خطی چندمتغیره را نشان می‌دهد. سطح مدل (صورتی/قرمز) صاف و متعادل است، و نقاط آبی (داده‌های واقعی) نزدیک سطح قرار گرفته‌اند. R^2 بالا و MSE کم (۱۰۰-۲۰۰)، و R^2 بالا (۹۵.۰٪).

(ه‌ن‌ی‌ه‌ب) Good Fit: ه‌ری‌غ‌ت‌م‌د‌ن‌ج‌ی‌ط‌خ‌ن‌وی‌س‌ر‌گ‌ر



شکل ۸: Good Fit: سطح صاف، نقاط نزدیک (تعادل Bias-Variance Tradeoff). محورها: x_1 (۴۰-۲۰۰)، x_2 (۱-۵)، y (۰-۲۵۰۰).

نشانه‌های Good Fit:

- سطح مدل صاف و خطی است، بدون منحنی (over-complex) یا مسطح (underfit).
- نقاط آبی خیلی نزدیک سطح پراکندگی کم، تعمیم‌پذیری بالا.
- مدل الگوهای اصلی رو یاد گرفته بدون نوز Train/Test Error مشابه.
- برای دستیابی، از regularization (بخش ۵.۱.۲) cross-validation استفاده کنید.

کمینه‌سازی تابع هزینه در رگرسیون خطی چندگانه رگرسیون خطی چندگانه رابطه بین متغیر وابسته (y) و چندین متغیر مستقل (x_1, x_2, \dots, x_n) را به صورت زیر مدل می‌کند:

$$y = w_0 + w_1x_1 + w_2x_2 + \dots + w_nx_n \quad (۴)$$

به صورت ماتریسی: $y = Xw$ ، که در آن:

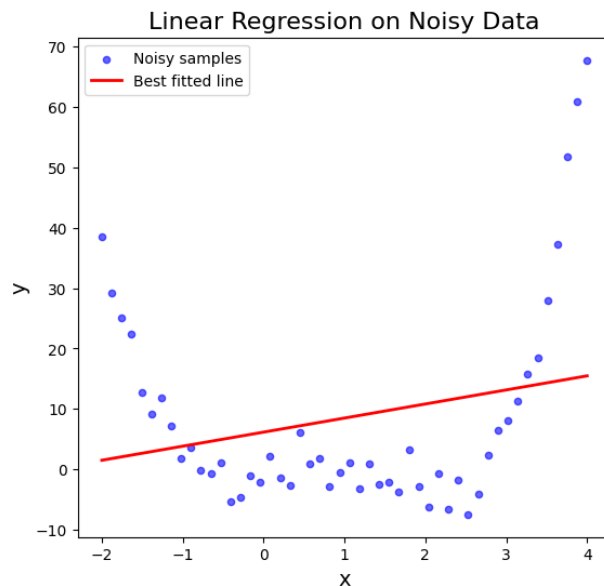
• $X = [1, x_1, x_2, \dots, x_n]$: ماتریس ویژگی‌ها (ستون اول برای w_0 پر از ۱ است).

• $w = [w_0, w_1, \dots, w_n]^T$: بردار ضرایب.

• y : بردار خروجی‌ها.

تابع هزینه تابع هزینه میانگین مربعات خطا (MSE) به صورت زیر تعریف می‌شود:

$$J(w) = \frac{1}{n} \sum_{i=1}^n (y_i - (w_0 + w_1x_{i1} + \dots + w_nx_{in}))^2 \quad (۵)$$



شکل ۹: استفاده از رگرسیون خطی با که باعث نویز شده است.

یا به صورت ماتریسی:

$$J(w) = \frac{1}{n}(y - Xw)^T(y - Xw) \quad (۶)$$

کمینه سازی رگرسیون چند جمله ای برای کمینه سازی $J(w)$ ، مشتق تابع هزینه نسبت به w محاسبه و صفر می شود. تابع هزینه را گسترش می دهیم:

$$J(w) = \frac{1}{n} [(y^T y - 2y^T Xw + w^T X^T Xw)]$$

مشتق نسبت به w :

$$\begin{aligned} \frac{\partial J}{\partial w} &= \frac{\partial}{\partial w} \left[\frac{1}{n} (y^T y - 2y^T Xw + w^T X^T Xw) \right] \\ &= \frac{1}{n} (-2X^T y + 2X^T Xw) = \frac{2}{n} (X^T Xw - X^T y) \end{aligned} \quad (۷)$$

صفر کردن مشتق:

$$X^T Xw = X^T y \quad (۸)$$

برآوردگر w :

$$\hat{w} = (X^T X)^{-1} X^T y \quad (۹)$$

این برآوردگر بهینه، تابع هزینه را کمینه می کند، مشروط بر اینکه $X^T X$ معکوس پذیر باشد (یعنی ویژگی ها هم خطی کامل نداشته باشند).

تعمیم حالت خطی.؟؟ رگرسیون خطی ساده (Simple Linear Regression) برای مدل سازی رابطه بین یک متغیر مستقل (x)

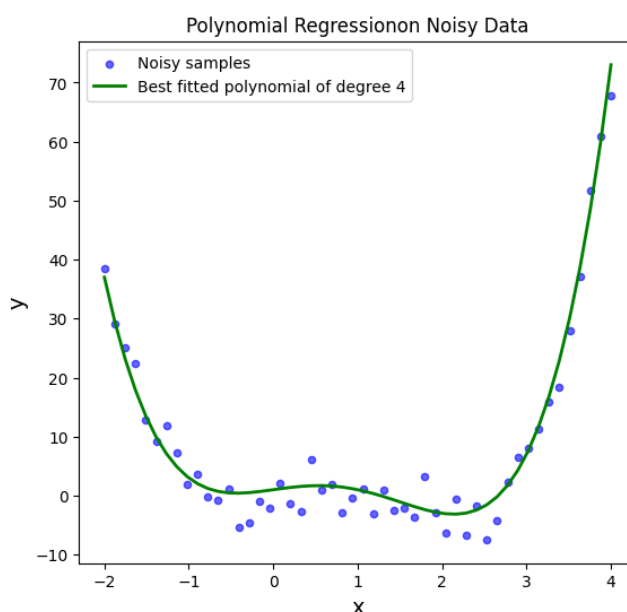
و یک متغیر وابسته (y) به کار می‌رود، اما در بسیاری از مسائل واقعی، خروجی (y) تحت تأثیر چندین متغیر مستقل است. در چنین مواردی، رگرسیون خطی چندگانه (Multiple Linear Regression) به دلیل توانایی در نظر گرفتن چندین متغیر مستقل به طور همزمان، انتخاب مناسب‌تری است. در ادامه، به دلایل اصلی استفاده از رگرسیون چندگانه به جای رگرسیون خطی ساده، استفاده خواهد شد.

همان طوری که در بالا هم مشاهده می‌شود وقتی از رگرسیون خطی روی داده‌های چند گانه می‌شود باعث این می‌شود یک نویز یا خطایی به وجود آید حتی برای آن یک تخمین اولیه را نداشته باشیم. برای رفع این مشکل می‌توانیم از رگرسیون‌های چند گانه با درجه‌هایی مختلفی استفاده کنیم و این نویز را از بین ببریم. به تصویر عدی نگاه کنید.

همان طور که در تصویر بالا مشاهده می‌شود زمانی که درجه را زیاد می‌کنیم باعث می‌شود که نمودار به سمت چند جمله برود در نهایت خطایی مدل ما خیلی کاهش یابد و از overfit جلوگیری شود. نکته‌ای باید مد نظر داشته باشید شما می‌توانید از چند تبدیل‌های مختلفی استفاده کنید و بهترین برارزش را برای داده خود انجام دهید. در ادامه به برخی از تبدیل‌ها اشاره خواهیم کرد.

انواع تبدیل‌ها

- تبدیل لگاریتمی: $y' = \log(y)$ یا $x'_i = \log(x_i)$ برای داده‌های با توزیع کج یا روابط نمایی.
- تبدیل چندجمله‌ای: $x'_i = x_i^2, x_i^3, \dots$ برای مدل‌سازی روابط غیرخطی.
- تبدیل ریشه مربع: $y' = \sqrt{y}$ یا $x'_i = \sqrt{x_i}$ برای کاهش کجی یا اثر مقادیر بزرگ.
- تبدیل معکوس: $y' = \frac{1}{y}$ برای روابط معکوس یا مدیریت داده‌های پرت.



شکل ۱۰: استفاده نمودار غیر خطی .

از جمله تبدیل‌های هستند که می‌توانید برای برزاش داده استفاده کنید از overfit و underfit شدن جلوگیری کنید.

۳.۲.۳ گرادیان (gradient)

گرادیان نزولی (Gradient Descent) یکی از الگوریتم‌های بهینه‌سازی پرکاربرد در یادگیری ماشین است که برای کمینه‌سازی تابع هزینه (Cost Function) در مدل‌هایی مانند رگرسیون خطی، رگرسیون لجستیک، و شبکه‌های عصبی استفاده می‌شود. این روش

به‌ویژه در مواردی که داده‌ها حجم زیادی دارند یا محاسبات تحلیلی (مانند روش حداقل مربعات) هزینه‌بر هستند، بسیار مفید است. در ادامه، توضیح جامعی درباره گرادیان نزولی، نحوه کار، انواع آن، کاربردها، و مزایا و معایب آن ارائه می‌شود. گرادیان نزولی یک الگوریتم تکراری است که برای یافتن مقادیر بهینه پارامترهای مدل مانند (w_0, w_1, \dots) به‌گونه‌ای که تابع هزینه $J(w)$ کمینه شود، استفاده می‌شود. ایده اصلی این است که با حرکت در جهت مخالف گرادیان (مشتق تابع هزینه)، به سمت نقطه کمینه حرکت کنیم. گرادیان: بردار مشتق‌های جزئی تابع هزینه نسبت به پارامترها، که جهت و شدت تغییرات تابع هزینه را نشان می‌دهد. نزولی: حرکت در جهت مخالف گرادیان برای کاهش مقدار تابع هزینه. شروع با مقادیر اولیه: مقادیر اولیه تصادفی یا صفر برای پارامترها (w) انتخاب می‌شود. محاسبه گرادیان: مشتق جزئی تابع هزینه نسبت به هر پارامتر محاسبه می‌شود:

$$\frac{\partial J}{\partial w_j}$$

به‌روزرسانی پارامترها: پارامترها در جهت مخالف گرادیان با گام مشخص (نرخ یادگیری، α) به‌روزرسانی می‌شوند:

$$w_j := w_j - \alpha \frac{\partial J}{\partial w_j}$$

تکرار: مراحل بالا تا زمانی تکرار می‌شود که تابع هزینه به مقدار کمینه برسد یا تغییرات آن ناچیز شود (همگرایی).

انواع گرادیان

گرادیان نزولی بر اساس نحوه استفاده از داده‌ها به سه نوع اصلی تقسیم می‌شود:

1. گرادیان نزولی دسته‌ای (Batch Gradient Descent): گرادیان با استفاده از کل داده‌های آموزشی محاسبه می‌شود. مزایا: همگرایی پایدارتر به سمت کمینه. معایب: برای داده‌های بزرگ، محاسبات سنگین و کند است.
2. گرادیان نزولی تصادفی (Stochastic Gradient Descent - SGD): گرادیان برای هر نمونه داده به‌صورت جداگانه محاسبه و پارامترها به‌روزرسانی می‌شوند. مزایا: سریع‌تر برای داده‌های بزرگ، می‌تواند از کمینه‌های محلی فرار کند. معایب: نوسانات زیاد در تابع هزینه، ممکن است به کمینه دقیق نرسد.
3. گرادیان نزولی مینی‌بچ (Mini-Batch Gradient Descent): داده‌ها به دسته‌های کوچک (mini-Batches) تقسیم می‌شوند و گرادیان برای هر دسته محاسبه می‌شود. مزایا: تعادل بین سرعت و پایداری، پرکاربرد در شبکه‌های عصبی. معایب: نیاز به تنظیم اندازه دسته.

کاربرد گرادیان نزولی در رگرسیون خطی در رگرسیون خطی، تابع هزینه معمولاً میانگین مربعات خطا (MSE) است:

$$J(w) = \frac{1}{n} \sum_{i=1}^n (y_i - (w_0 + w_1 x_{i1} + \dots + w_n x_{in}))^2$$

گرادیان نزولی برای یافتن w_0, w_1, \dots, w_n استفاده می‌شود:

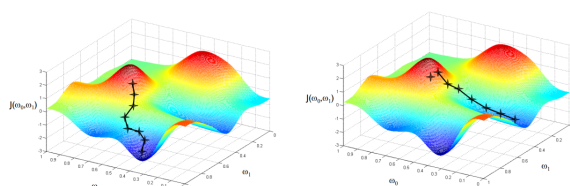
مشتق‌های جزئی:

$$\frac{\partial J}{\partial w_0} = -\frac{2}{n} \sum_{i=1}^n (y_i - \hat{y}_i)$$

$$\frac{\partial J}{\partial w_j} = -\frac{2}{n} \sum_{i=1}^n (y_i - \hat{y}_i) x_{ij}, \quad j = 1, \dots, n$$

$$w_j := w_j + \alpha \frac{2}{n} \sum_{i=1}^n (y_i - \hat{y}_i) x_{ij}$$

cost function در این نمودار یک سطح سه بعدی است که محورهای آن به صورت زیر تعریف شده‌اند:



شکل ۱۱: گرادیان کاهشی

- محور x : θ_0 (پارامتر اول، مثلاً w_0).
- محور y : θ_1 (پارامتر دوم، مثلاً w_1).
- محور هر نمودار یک سطح سه بعدی است که محورهای آن به صورت زیر تعریف شده‌اند:

محور x : θ_0 (پارامتر اول، مثلاً w_0). محور y : θ_1 (پارامتر دوم، مثلاً w_1). محور z : $J(\theta_0, \theta_1)$ (مقدار تابع هزینه).
نمودار سمت چپ: مسیر گرادیان نزولی با نرخ یادگیری (α) بالاتر یا تعداد مراحل کمتر نشان داده شده است. مسیر دارای انحنا و نوسانات بیشتری است و به نظر می‌رسد هنوز به کمینه دقیق نرسیده است.
نمودار سمت راست: مسیر صاف‌تر و مستقیم‌تر است، که نشان‌دهنده نرخ یادگیری مناسب‌تر یا تعداد مراحل بیشتر است. این مسیر به طور واضح به سمت کمینه همگرا شده است.

۴.۲.۳ رگرسیون لجستیک

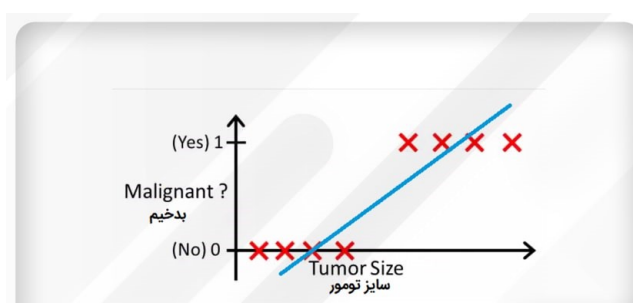
رگرسیون لجستیک یکی از الگوریتم‌های طبقه‌بندی (Classification) است که برای اختصاص داده‌ها به مجموعه‌ای از کلاس‌ها استفاده می‌شود. برخی از نمونه‌های مسائل طبقه‌بندی عبارت‌اند از: طبقه‌بندی ایمیل‌ها به دو دسته‌ی ایمیل‌های اسپم (Spam) یا غیراسپم (Not Spam) یا طبقه‌بندی معاملات آنلاین به دو دسته‌ی کلاهبرداری یا غیرکلاهبرداری یا طبقه‌بندی تومورهای بدخیم یا خوش‌خیم. همان‌طور که تا الان متوجه شدیم، به‌طور کلی در رگرسیون لجستیک خروجی به‌شکل صفر یا ۱ است؛ یعنی برای مثال، یا تورمور بدخیم است (1) یا خوش‌خیم (0).

زمانی که تعداد کلاس‌های خروجی ۲ باشد، به آن طبقه‌بندی باینری (Binary Classification) گفته می‌شود؛ البته تعداد کلاس‌های خروجی می‌تواند بیشتر هم باشد که در این صورت به آن طبقه‌بندی مالتی (Multi Classification) گفته می‌شود.

تا اینجا فهمیدیم که در رگرسیون لجستیک هدف در خروجی اختصاص داده‌ها به یکی از دو کلاس صفر یا ۱ است. در واقع در این تکنیک احتمال این را که داده به کدام کلاس متعلق است در خروجی خواهیم داشت. از آنجا که درباره‌ی احتمال صحبت می‌کنیم، پس قطعاً می‌دانیم خروجی ما باید بین صفر و ۱ باشد. حال لازم است ببینیم در اینجا ما به چه تابعی احتیاج داریم که بتواند داده‌ها را به‌خوبی نمایش دهد و در کلاس درستی طبقه‌بندی کند.

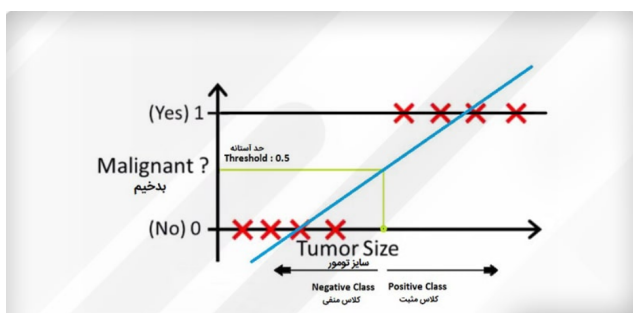
چرا رگرسیون خطی نمی‌شود؟

فرض کنید ما مجموعه داده‌ای داریم که اندازه‌ی تومورها و این امر را که آیا بدخیم هستند یا نه نشان می‌دهد. اگر این داده‌ها را روی نمودار ببریم، از آنجا که مسئله طبقه‌بندی است، تمامی داده‌ها یا صفر هستند یا یک.



شکل ۱۲: گرادیان کاهشی

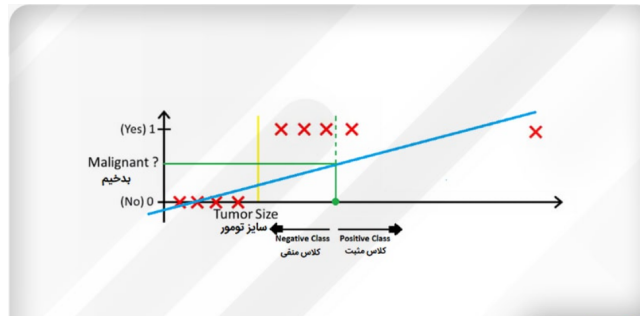
تعیین یک آستانه (Threshold) در محور θ داده‌ها را طوری تقسیم کنیم که همه‌ی داده‌های موجود در سمت راست آستانه‌ی بدخیم (متعلق به کلاس ۱ یا همان مثبت) و در سمت چپ آن همه‌ی داده‌های موجود خوش‌خیم (متعلق به کلاس صفر یا همان منفی) هستند. در اینجا این آستانه 0.5 است.



شکل ۱۳: گرادیان کاهشی

خب، تا اینجا کار رگرسیون خطی خوب جواب داده است و شاید فکر کنیم که اصلاً به رگرسیون لجستیک نیازی نداریم، اما این‌طور نیست.

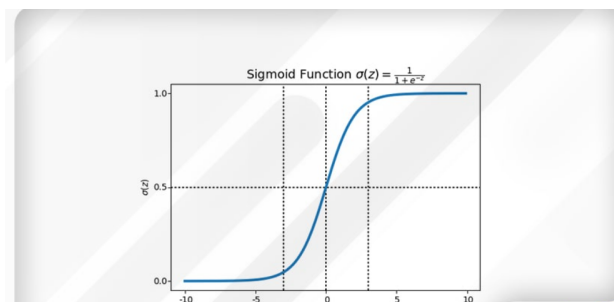
بیا ببینیم در نظر بگیریم یک داده‌ی پرت (Outlier) به این داده‌ها اضافه شود؛ در این صورت خطی که رگرسیون خطی به ما می‌دهد به این شکل خواهد بود:



شکل ۱۴: گزاردیان کاهشی

در اینجا اگر بازه‌ی ۵.۰ را بار دیگر بررسی کنیم، می‌بینیم که داده‌ها به‌درستی تقسیم نمی‌شوند؛ خیلی از کلاس‌های مثبت را به‌اشتباه در دسته‌ی کلاس‌های منفی قرار می‌دهد. درواقع باید بگوییم بار اول رگرسیون خطی به‌صورت اتفاقی داده‌ها را به‌درستی تقسیم کرد؛ پس به‌همین دلیل، ما به رگرسیون لجستیک نیاز داریم.

همان‌طور که می‌دانیم، در رگرسیون خطی ما باید به بهترین معادله‌ی خطی می‌رسیدیم تا داده‌ها را به‌درستی نمایش دهد. این را در نظر بگیریم که در رگرسیون خطی معادله‌ی خطی می‌تواند هر مقداری داشته باشد، درحالی‌که در طبقه‌بندی و رگرسیون لجستیک (Logistic Regression) مقداری که در خروجی داریم احتمال تعلق داده به یکی از دو کلاس صفر یا ۱ است؛ بنابراین مقادیر باید میان صفر و ۱ باشد که این موضوع با معادله خطی امکان‌پذیر نیست.



شکل ۱۵: گزاردیان کاهشی

پس ما به معادله‌ی دیگری احتیاج داریم که آن تابع سیگموئید یا همان لجستیک (Sigmoid/ Logistic Function) است.

تعریف ریاضی می‌توان تابع لجستیک را به این شکل تعریف کرد.

$$\hat{Y} = \sigma(z) = \sigma(x_0w_0 + x_1w_1 + \dots + x_nw_n)$$

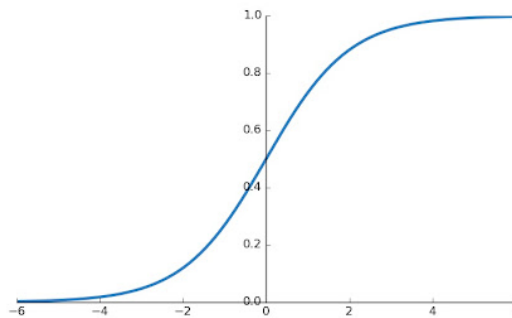
تابع سیگموند به این شکل می‌باشد.

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

و همچنین در مورد مشتق پذیری

$$\begin{aligned}\lim_{z \rightarrow -\infty} \sigma(x) &= \frac{1}{1 + \infty} = 0 \\ \lim_{z \rightarrow \infty} \sigma(x) &= \frac{1}{1 + 0} = 1 \\ \lim_{z \rightarrow 0} \sigma(x) &= \frac{1}{1 + 1} = 0.5\end{aligned}$$

از آن جایی که $-\infty < x < \infty$ پس می توان نوشت $0 < \sigma(z) < 1$ می باشد. و نمودار به شکل زیر می باشد



شکل ۱۶: نمودار تابع لجستیک

یکی دیگر از ویژگی های خوبی که این تابع دارد مشتق پذیر است.

$$\frac{\partial}{\partial z} \left(\frac{1}{1 + e^{-z}} \right) = -\frac{e^{-z}}{(1 + e^{-z})^2} = -\frac{e^{-z}}{(1 + e^{-z})} \frac{1}{1 + e^z} = (\sigma(z))(1 - \sigma(z))$$

در یادگیری ماشین ما نیاز داریم که عددی که یا پیش بینی های که به ما باز گردانده می شوند یک عدد بین صفر و یک باشد که برای همین می توان از تابع لجستیک به شکل نوشت:

$$\begin{aligned}p(y = 1 | x, w) &= \sigma(w^T x) \\ p(y = 0 | x, w) &= (1 - \sigma(w^T x))\end{aligned}$$

به دست آوردن تابع هزینه در رگرسیون لجستیک

$$P(y_i = 1 | x_i; w) = \sigma(w^T x_i) = \frac{1}{1 + e^{-w^T x_i}}$$

برای $y_i \in \{0, 1\}$ توزیع برنولی را داریم:

$$P(y_i | x_i; w) = \sigma(w^T x_i)^{y_i} \cdot (1 - \sigma(w^T x_i))^{1-y_i}$$

با فرض مستقل بودن مشاهدات، تابع درست‌نمایی برابر است با:

$$L(w) = \prod_{i=1}^n \sigma(w^\top x_i)^{y_i} \cdot (1 - \sigma(w^\top x_i))^{1-y_i}$$

برای ساده‌سازی، لگاریتم تابع درست‌نمایی را می‌گیریم:

$$\ell(w) = \log L(w) = \sum_{i=1}^n [y_i \log \sigma(w^\top x_i) + (1 - y_i) \log(1 - \sigma(w^\top x_i))]$$

حال برای پیدا کردن بردار وزن بهینه w ، گرادیان تابع لگاریتم درست‌نمایی را محاسبه می‌کنیم:

$$\nabla_w \ell(w) = \sum_{i=1}^n (y_i - \sigma(w^\top x_i)) x_i$$

این تابع شبیه به رگرسیون نیست که با استفاده از فرم بسته بتوان برای w پیدا کرد. پس با استفاده از گرایان مقدار بهینه را برای پارامتر پیدا خواهیم کرد.

برای بیشینه‌سازی تابع درست‌نمایی، از الگوریتم‌های عددی مانند گرادیان افزایشی (Gradient Ascent) یا گرادیان کاهش (Gradient Descent) استفاده می‌شود. در عمل، ما معمولاً گرادیان کاهش را برای کمینه‌سازی تابع هزینه معادل استفاده می‌کنیم:

$$J(w) = -\ell(w) = -\sum_{i=1}^n [y_i \log \sigma(w^\top x_i) + (1 - y_i) \log(1 - \sigma(w^\top x_i))]$$

مشتق منفی آن نیز همان است:

$$\nabla_w J(w) = -\sum_{i=1}^n (y_i - \sigma(w^\top x_i)) x_i$$

۵.۲.۳ نظم‌دهی (Regularization)

یک تکنیک مهم در ماشین لرنینگ است که باعث کاهش کمینه کردن و بهبود دقت می‌شود، با جلوگیری از بیش‌برازش (Overfitting) (حذف داده‌های پردت و باعث کاهش نویز می‌شود و کمک می‌کند مدل ساده‌تر شود و روی داده‌های جدید به خوبی عمل می‌کند. این عمل فقط با افزودن یک ترم یا جریمه صورت خواهد گرفت.

انواع (Regularization) مدلی رگرسیونی که از تکنیک نظم‌دهی L1 استفاده می‌کند، رگرسیون لاسو (LASSO) نام دارد که مخفف (Least Absolute Shrinkage and Selection Operator) است. در این روش، مقدار قدر مطلق ضرایب (ضریب‌های مدل) به عنوان یک جریمه (penalty) به تابع زیان (L) اضافه می‌شود. این جریمه می‌تواند باعث شود برخی ضرایب کاملاً به صفر برسند، که در نتیجه ویژگی‌های کم‌اهمیت حذف شده و تنها ویژگی‌های مهم باقی می‌مانند.

۶.۲.۳ رگرسیون لاسو

Lasso Regression یک روش رگرسیونی مبتنی بر تکنیک "کوچک‌سازی و انتخاب بر اساس قدر مطلق است که در تحلیل رگرسیون برای انتخاب متغیرها و اعمال نظم‌دهی Regularization استفاده می‌شود. این روش به حذف ویژگی‌های غیرمرتبط داده کمک می‌کند و از بیش‌برازش Overfitting جلوگیری می‌کند. در نتیجه، ویژگی‌هایی که تأثیر ضعیفی دارند به خوبی مشخص می‌شوند، چرا که ضرایب متغیرهای کم‌اهمیت به سمت صفر میل می‌کنند. در حقیقت می‌توان گفت که رگرسیون لاسو یکی از روش‌هایی است که با افزودن یک

جمله‌ی جریمه به تابع هزینه مدل خطی، از بیش‌برازش جلوگیری می‌کند. تابع هزینه در این روش به صورت زیر تعریف می‌شود:

$$J(w) = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i) + \lambda \sum_{i=1}^m |w_i|$$

و همچنین در فرمول بالا $\sum (y_i - \hat{y}_i)^2$ $minRss$

y = مقدار مشاهده شده.

\hat{y} = مقدار پاسخ بیش بینی شده

و در قسمت جمله جریمه $\lambda \sum |w|$ (penalty prem) این تابع مشتق پذیر نمی باشد.

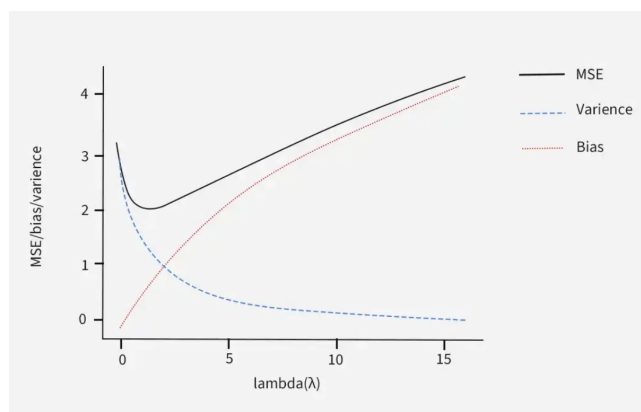
w = ضرایب

λ یک پارامتر تنظیم (uning parameter) است که شدت جریمه (penalty) را کنترل می‌کند. هرچه مقدار λ بیشتر شود، ضرایب بیشتری به سمت صفر رانده می‌شوند.

و می توان به شکل زیر فرمول بالا را بازنویسی کرد:

$$J(w) = \min RSS + \lambda \sum_{i=1}^m |w_i|$$

تنظیم مقدار بهینه **لاندا** در ادامه دو معیار مهم در ماشین لرنینگ را مورد بررسی قرار خواهیم داد.

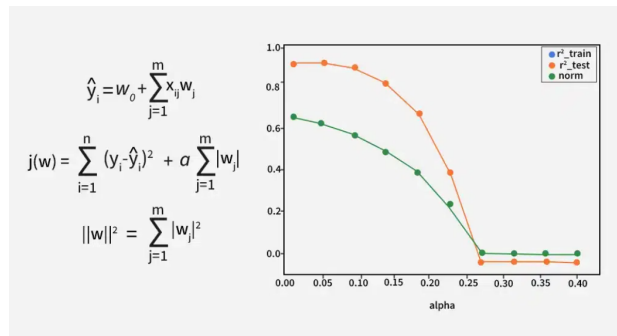


شکل ۱۷: bias and varince in lasso

Bias = وقتی یک مدل بسیار ساده باشد و داده های ما پیچیده باشند نقاط پرت زیاد می شوند در نهایت خطا افزایش می یابد.

variance = در حقیقت وقتی مدل پیچیده باشد می تواند تنها روی داده های آموزش یا داده تست به خوبی فیت شود و در حالی در برگیرنده داده های آموزش نیست در نهایت می توان گفت که واریانس زیاد شده و اورفیت رخ داده است.

panlty = کاربردی که دارد باعث می شود ویژگی های که اهمیت کمی دارند حذف شوند. و این عمل باعث کاهش واریانس می شود و از اورفیت شدن جلوگیری میکند. **افزایش مقدار λ** باعث این می شود مدل بیش از حد ساده باشد و در نهایت باعث این میشود که **underfit** رخ دهد. برای رسیدن به تعادل درست بین **Bias** و **Variance** باید مقدار مناسب λ را با استفاده از **cross-validation** پیدا کرد.



شکل ۱۸: cost function

در نمودار مربوط به Lasso Regression، تابع هزینه به صورت ترکیبی از مجموع مربعات باقیمانده‌ها (Residual Sum of Squares) یا (RSS) و یک جریمه‌ی L1 بر ضرایب β_j تعریف می‌شود. بخش اول یعنی RSS، اختلاف مربعی بین مقادیر واقعی و مقادیر پیش‌بینی‌شده را اندازه‌گیری می‌کند. بخش دوم، یعنی جریمه‌ی L1 (یا L1 penalty)، مقدار قدر مطلق ضرایب را جریمه می‌کند و باعث می‌شود برخی از ضرایب به صفر نزدیک شوند یا دقیقاً صفر شوند. این امر باعث ساده‌تر شدن مدل می‌گردد. شدت اثر این جریمه توسط پارامتر λ (یعنی lambda) کنترل می‌شود.

در نمودار، محور عمودی (محور y) مقدار تابع هزینه را نشان می‌دهد که Lasso Regression تلاش می‌کند آن را کمینه کند، و محور افقی (محور x) مقدار پارامتر λ را نمایش می‌دهد که شدت جریمه‌ی L1 را مشخص می‌کند. منحنی‌ای که از رنگ سبز به نارنجی تغییر می‌کند، نشان‌دهنده‌ی تغییرات مقدار تابع هزینه نسبت به تغییرات λ است. با افزایش λ ، مقدار جریمه بیشتر شده و در نتیجه مقدار تابع هزینه نیز افزایش می‌یابد. این افزایش موجب می‌شود ضرایب بیشتری به سمت صفر رانده شوند و در نتیجه مدل ساده‌تر شود.

۷.۲.۳ رگرسیون ریج

رگرسیون ریج یک regularized است از رگرسیون خطی. وبا اضافه کردن یک تابع جریمه به cost function کمک می‌کند یک به نقطه بهینه برسیم و هم چنین *overfit* کمتری رخ بدهد.

$$J(\theta) = MSE(\theta) + \alpha \sum_i^n \theta_i^2$$

نکته: در رگرسیون ریج به این مورد توجه کنید که ویژگی‌های که اهمیت کمی دارند کوچک می‌شوند یا نزدیک به صفر. نکته: از این ترم $\alpha \sum_{i=1}^n \theta_i^2$ استفاده می‌شود.

نکته: برای این که معیار عمل کرد مدل را تشخیص دهیم. یعنی وقتی که مدل را استفاده از تست دیتا عملکرد آن را مورد سنجش قرار می‌دهیم از این ترم استفاده خواهیم $J(\theta) = MSE(\theta)$ این قسمت برای این که متشقق پذیر است و گرادین آن راحت حساب می‌شود از آن استفاده خواهیم کرد.

توان منظم‌دهی (Regularization strength) با آلفا (α) مشخص می‌شود که یکی از hyperparameter و دو ویژگی دارد: 1. اگر $\alpha = 0$ باشد، هیچ منظم‌دهی اعمال نمی‌شود و مدل به رگرسیون خطی استاندارد (OLS) تبدیل می‌شود.

مفهوم	تعریف	تفاوت کلیدی	ارتباط با هم
Gradient Descent	الگوریتم تکراری برای به روزرسانی θ با مشتق تابع هزینه: $w_j \leftarrow w_j - \alpha \frac{\partial J}{\partial w_j}$	ابزار بهینه سازی (چگونگی آموزش مدل).	می تواند overfitting را ایجاد کند (epochs زیاد)، اما با penalty کنترل می شود.
Overfitting	مدل over-complex به نوبت train می چسبد (high variance).	مشکل عملکرد (عدم تعمیم).	reg- ularization بدون overfitting را تشدید می کند.

جدول ۱: مقایسه گرادیان نزولی و اورفیتینگ.

نقش گرادیان نزولی در رگرسیون

گرادیان نزولی پارامترها (θ) را بهینه می کند تا تابع هزینه کمینه شود.

در رگرسیون خطی (Linear Regression)

- نقش: کمینه سازی MSE: $J(w) = \frac{1}{n} \sum (y_i - \hat{y}_i)^2$. مشتق: $\frac{\partial J}{\partial w_j} = -\frac{2}{n} \sum (y_i - \hat{y}_i)x_{ij}$. کاربرد: تنظیم w برای پیش بینی پیوسته (مثل قیمت خانه). انواع: Batch/SGD برای داده های بزرگ. - مزیت: scalable؛ جلوگیری از overfitting با early stopping.

در رگرسیون لجستیک (Logistic Regression)

- نقش: کمینه سازی Cross-Entropy: $J(w) = -\sum [y_i \log \sigma(w^T x_i) + (1 - y_i) \log(1 - \sigma(w^T x_i))]$. مشتق: $\frac{\partial J}{\partial w_j} = \frac{1}{n} \sum (\sigma(w^T x_i) - y_i)x_{ij}$. کاربرد: طبقه بندی باینری (مثل اسپم). - ضروریه چون فرم بسته ندارد. - مزیت: برای داده های نامتوازن؛ با sigmoid غیرخطی.

کجا بیشتر کاربرد دارند؟

- Gradient Descent: در θ (شبکه های عصبی، ۹۰- Overfitting: در مدل های پیچیده (multiple features)؛ نقش گرادیان: بهینه سازی، اما با regularization (Lasso/Ridge) کنترل می شه.

مدل	کاربرد اصلی گرادیان
Linear Regression	پیش بینی پیوسته، داده های بزرگ (اقتصاد/پزشکی).
Logistic Regression	طبقه بندی باینری (تشخیص بیماری، θ).

جدول ۲: کاربرد گرادیان در رگرسیون.

۳.۳ یادگیری بدون نظارت

یادگیری بدون نظارت (یا یادگیری ماشین بدون نظارت) یکی از تکنیک های یادگیری ماشین (LM) است که از الگوریتم های هوش مصنوعی (AI) برای شناسایی الگوها در مجموعه داده هایی که طبقه بندی یا برچسب گذاری نشده اند، استفاده می کند.

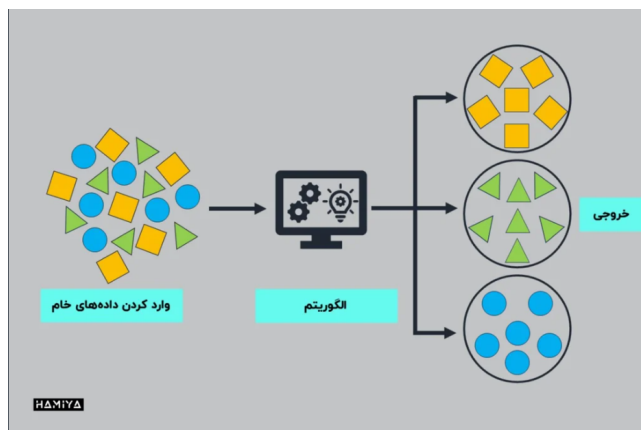
مدل های یادگیری نظارت نشده در هنگام آموزش مجموعه داده ها به نظارت نیاز ندارند و این امر آن ها را به تکنیکی ایده آل در یادگیری

ماشین برای کشف الگوها، گروه بندی ها و تفاوت ها در داده های بدون ساختار تبدیل می کند. این مدل ها برای فرآیندهایی مانند بخش بندی مشتریان، تحلیل اکتشافی داده ها یا تشخیص تصویر بسیار مناسب هستند.

الگوریتم های یادگیری بدون نظارت می توانند نقاط داده موجود در مجموعه داده ها را بدون نیاز به هیچ راهنمایی خارجی برای انجام آن کار، طبقه بندی، برچسب گذاری و گروه بندی کنند. به عبارت دیگر، یادگیری بدون ناظر به یک سیستم اجازه می دهد تا به طور مستقل الگوهای موجود در مجموعه داده ها را شناسایی کند.

در یادگیری بدون نظارت، یک سیستم هوش مصنوعی اطلاعات طبقه بندی نشده را بر اساس شباهت ها و تفاوت ها گروه بندی می کند، حتی اگر هیچ دسته بندی از پیش تعیین شده ای وجود نداشته باشد.

سیستم های هوش مصنوعی که قادر به یادگیری بدون نظارت هستند، اغلب با مدل های یادگیری تولیدکننده مرتبط هستند، اگرچه ممکن است از رویکرد مبتنی بر بازیابی نیز استفاده کنند که بیشتر با یادگیری تحت نظارت مرتبط است. چت بات ها، خودروهای خودران، برنامه های تشخیص چهره، سیستم های خبره و ربات ها از جمله سیستم هایی هستند که از رویکردهای یادگیری با نظارت یا بدون نظارت استفاده می کنند. یادگیری بدون نظارت همچنین به عنوان یادگیری ماشین بدون نظارت شناخته می شود.



شکل ۱۹: unsupervised learning

در آمار، خوشه بندی شامل روش هایی مانند K-Means، خوشه بندی سلسله مراتبی و مدل های مخلوط (مانند مخلوط گاوسی) است که برای بخش بندی مشتری ها بدون داشتن لیبل، تشخیص گروه های پنهان، جداسازی افراد براساس ویژگی ها و یافتن الگوهای مشابه در داده های پزشکی به کار می رود؛ همان مفهومی که در تصویر دیده می شود، و از میان این روش ها، K-Means ساده ترین و پرکاربردترین الگوریتم است

۱.۳.۳ K-means

۱. خوشه بندی (Clustering):

- داده ها را بر اساس شباهت به گروه هایی تقسیم می کند.
- نقاط داده در یک خوشه به یکدیگر شبیه ترند تا به نقاط خوشه های دیگر.
- شباهت بسته به نوع مسئله تعریف می شود (مثلاً رفتار خرید در تقسیم بندی بازار).

۲. کاهش ابعاد (Dimensionality Reduction):

- تعداد ویژگی های داده را کاهش می دهد، اما ویژگی های مهم و اطلاعاتی را حفظ می کند.

۳. تشخیص ناهنجاری (Anomaly Detection):

- شناسایی نقاط داده‌ای که به طور قابل توجهی از حالت عادی منحرف هستند (مثلاً تشخیص تقلب).

۴. مدل سازی مولد (Generative Modeling):

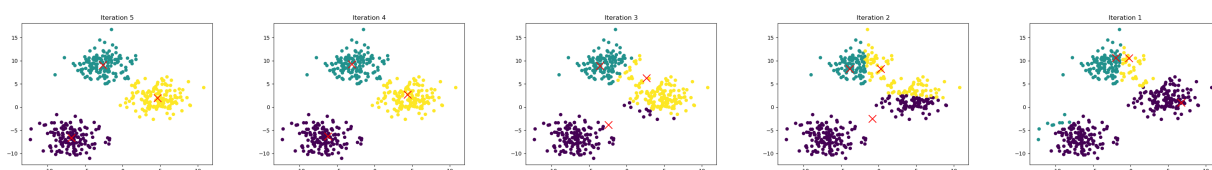
- یادگیری توزیع داده‌ها برای تولید نمونه‌های جدید و مشابه.

کاربردهای خوشه‌بندی

- تقسیم‌بندی مشتریان (Customer Segmentation): در بازاریابی برای گروه‌بندی مشتریان بر اساس رفتار.
- تقسیم‌بندی تصویر و تشخیص اشیا (Image Segmentation and Object Detection): در بینایی کامپیوتری.
- تشخیص ناهنجاری: در امنیت سایبری و امور مالی.
- ژنتیک و بیوانفورماتیک: تحلیل داده‌های زیستی.
- تحلیل شبکه‌های اجتماعی و تشخیص جوامع: شناسایی گروه‌های مشابه در شبکه‌ها.

الگوریتم خوشه‌بندی K-Means

- پرکاربردترین الگوریتم خوشه‌بندی.
- داده‌ها را به K گروه مجزا بر اساس شباهت ویژگی‌ها تقسیم می‌کند.
- با تخصیص تکراری نقاط داده به نزدیک‌ترین مرکز (میانگین گروه) کار می‌کند و سپس مراکز را بر اساس عضویت‌های جدید گروه‌ها بازمحاسبه می‌کند.
- این فرآیند تا زمانی که تخصیص‌ها دیگر تغییر نکنند، تکرار می‌شود.



شکل ۲۰: K-means

- هدف: نقاط داده در یک خوشه باید به یکدیگر شبیه باشند.
- در K-Means، این هدف به صورت تابع هزینه زیر بیان می‌شود:

$$J = \sum_{j=1}^K \sum_{x^{(i)} \in C_j} \|x^{(i)} - \mu_j\|^2$$

- انتخاب f و $\mu = \{\mu_1, \mu_2, \dots, \mu_K\}$ برای کمینه کردن تابع J .

- این مسئله NP-hard است. الگوریتم K-Means یک راه حل هیوریستیک است که تضمین نمی کند حتماً به راه حل بهینه برسد.

- ابتدا هر نمونه به نزدیک ترین مرکز تخصیص داده می شود:

$$f(x) := \arg \min_j \|x - \mu_j\|^2$$

- تخصیص هر نمونه تا زمانی که مرکز نزدیک تری پیدا نشود، ثابت می ماند.
- هر بار که یک نمونه بازتخصیص می شود، مجموع فاصله بین نمونه ها و مراکز آن ها کاهش می یابد.
- تعداد تخصیص های ممکن نمونه به مرکز محدود است.
- الگوریتم زمانی خاتمه می یابد که هیچ نمونه ای مرکز تخصیص یافته خود را تغییر ندهد.
- در مرحله به روزرسانی، با ثابت نگه داشتن $f(x)$ ، تابع J به صورت یک تابع درجه دوم از μ_j (مانند مجموع مربعات خطا) است و با مشتق گیری می توان آن را کمینه کرد:

$$\frac{\partial J}{\partial \mu_j} = 0 \implies \sum_{x^{(i)} \in C_j} -2(x^{(i)} - \mu_j) = 0$$

- این به این معناست که هر μ_j باید به عنوان میانگین خوشه C_j به روزرسانی شود:

$$\mu_j = \frac{\sum_{x^{(i)} \in C_j} x^{(i)}}{|C_j|}$$

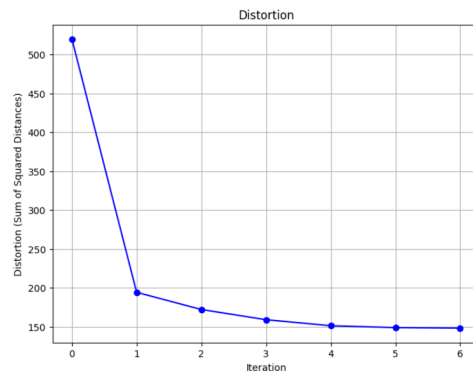
- برای هر خوشه، میانگین نقاط آن خوشه مقدار مجموع فاصله های مربعی را کمینه می کند.
- برای خوشه C_j ، اگر μ'_j مرکز قبلی باشد، داریم:

$$\sum_{x^{(i)} \in C_j} \|x^{(i)} - \mu'_j\|^2 \geq \sum_{x^{(i)} \in C_j} \|x^{(i)} - \mu_j\|^2$$

یعنی انتخاب میانگین جدید هرگز مقدار خطا را افزایش نمی دهد.

- در نتیجه، $J_{\text{new}} \leq J_{\text{old}}$ خواهد بود.
- از آنجایی که تابع هزینه J غیرمنفی است و تعداد تقسیم بندی های ممکن نیز محدود می باشد، دنباله مقادیر J نمی تواند تا بی نهایت کاهش یابد.
- بنابراین الگوریتم K-Means در نهایت همگرا می شود.
- ویژگی های همگرایی الگوریتم K-Means نخستین بار توسط مک کوئین (MacQueen) در سال ۱۹۶۷ بررسی شد.

cost همان طوری که در تصویر مشاهده می شود در نهایت کمتر و کمتر می شود. و به یک نقطه صفر می رسد که بهترین جواب را پیدا خواهد کرد.



شکل ۲۱: نمودار تابع هزینه در k-means